

Table of Contents

RPS v4.0.0

Introduction & Overview

Introduction

RPS Terms and Definitions

What's New in 4.0.0

Release Notes

Using the RPS API

RPS Sample Scenario - Tourism

RPS Install Guide

Ports, Protocols, & Security Guide

PostgreSQL

Common Repo Process

Design

RPS Software Design

RPS Configuration Management (DSC) Software Design

Data

RPS Data Persistence (CMDB) Software Design

Configuration Changes to a Two Domain Architecture

Data Validation Schema Definition

Data Validation Integration in RPS

Operations

Certificate Management

RPS Certificate Management Technical Design

Certificate Usage

Certificate Requirements for Linux Clients

Certificate Request Plugin Configuration

Certificate Request Process

Certificate Rolling

DSC Authoring

Authoring RPS DSC Partial Configs

Authoring RPS DSC Resources

DSC Pull Server

Introduction to DSC Pull Server

[RPS Settings for DSC Pull Server](#)

[Runbooks and TaskMaps for DSC Pull Server](#)

[Provisioning](#)

[Create a Host Through RPS](#)

[Create RPS Credentials](#)

[Create a Virtual Machine Template Through RPS](#)

[Create a Hyper-V Virtual Machine Through RPS](#)

[RPS Building iPXE ROMs](#)

[RPS PXE](#)

[Configuring ESXi VMs to Use iPXE](#)

[Using the Provisioning Service](#)

[Task Management Service](#)

[How to Add Runbooks to RPS](#)

[How to Get and Set the Default Runbook Folder](#)

[How to Modify Runbooks in RPS](#)

[How to Remove Runbooks From RPS](#)

[Task Management Service \(TMS\) Settings](#)

[Tasking](#)

[RPS Tasking Guide](#)

[RPS Task Assignment Diagram](#)

[Authoring RPS Runbooks](#)

[Role-Based Access Control \(RBAC\)](#)

[Introduction to RBAC](#)

[How to Add and Remove User Roles](#)

[How to Manage User Roles with PowerShell](#)

[How to Import and Export Users with the Web User Interface](#)

[How to Add and Remove Users with PowerShell](#)

[How to Add and Remove Users with the Web User Interface](#)

[Audit Entries](#)

[Synchronization](#)

[How to Add a Node to an Existing RPS Environment](#)

[How to Self-Register Your Node](#)

[How to Configure RPS Sync Settings](#)

[Logging](#)

[Introduction to Logging in RPS](#)

[How to Control Logging Behavior in RPS](#)

[Viewing RPS Logs](#)

[Writing Log Messages](#)

[Testing](#)

[RPS Testing Strategy](#)

[Additional Resources](#)

[RPS Automation Package Guidelines](#)

[Token Based Software Activation with PowerShell](#)

[Token Based Software Activation with RPS Runbooks](#)

[Token Based Software Activation Using the User Interface](#)

[Token Based Software Activation with MNActivation Tool](#)

[RPS Customization Guide - High Level Overview](#)

[RPS IPSheet Parser](#)

[RPS Instance Definition](#)

[RPS Instance Definition Item](#)

[RPS Instance Definition Node](#)

[How to Configure Logging for the RPS API](#)

[How to Configure RPS-Mapped Parameters](#)

[Creating Dynamic Resource and Target Groups](#)

[How to Import RPS Data Into the CMDB](#)

[How to Export the CMDB](#)

RPS v4.0.0

Last updated on June 28, 2021.

Last Reviewed and Approved on PENDING REVIEW

Welcome to the RPS v4.0.0 Documentation Landing Page

Rapid Provisioning System (**RPS**) is a flexible and powerful automation tool for managing software installation, updates, and configuration.

RPS Vision Statement

To build a robust automation framework that empowers organizations to quickly and securely build, configure, and maintain their systems' desired state over any network.

IMPORTANT

Documentation bundled with RPS v4.0.0 is accurate as of **9/20/2021**.

Updated documentation can be found at: <https://reactr.azurewebsites.us>

RPS Definitions

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

RPS Specific Terms

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Master Key	MasterKey	Master Key is a string value that's used to encrypt Protected Properties in the database. The scope of the Master Key is domain-wide, meaning that all nodes/targets on the domain will share the same Master Key.	1
Package		A Package is a zip archive that contains an RPS manifest file that describes its contents and how to deploy it, and deployable files needed to apply the Package.	1
Package Stream		A Package Stream is a grouping of Packages that can be approved or rejected as a group.	1
Rapid Provisioning System	RPS	"Rapid Provisioning System (RPS) is a generic, flexible automation and cyber compliance framework, based on Azure technology, that enables: Provisioning of compute and non-compute platforms and devices; patching (IAVA and App Updates); Configuration and firmware deployment; Collection and reporting of logs and telemetry. RPS Components consist (or will consist) of the following: API Sync Service CMDB SMA DSC PowerSTIG CDN Provisioning Service User Interface Utilities"	5
Rapid Provisioning System Application Programming Interface	RPS API	An application programming interface exposed directly by a series of PowerShell cmdlets or indirectly by the RPS Web GUI that facilitates interaction with the RPS solution for IT administrators.	5
Rapid Provisioning System Sync Service	RPS Sync Service	The RPS Sync service is used to synchronize RPS automation and reporting data between a distributor (parent) RPS node and a subscriber (child) RPS node. Relationships between a distributor and a subscriber are defined during RPS node registration and are stored in RPS node records in the RPS database.	5
Resource Assignments		Resource Assignments are the link between a single Resource (i.e. Certificates, Credentials, Password Policies) and a single Target Item (i.e. Virtual Machine, Network Interface Card). Resource Assignments are made to provide supplemental data to our task execution engine. So, when a Task needs to be run on a target, it may require a certain Resource in order to complete the execution.	1
Resource Group		A Resource group is a logical collection of resource items	1
Resource Item	ResourceItem	Resource Item's can be thought of as entities used to configure/provision Target Item's. The two go hand-in-hand. Resource Item's typically represent virtual storage, password policies, packages, base images, etc.	1
Task Map	TaskMap	Identifies a set of steps, what order they should be performed in, and what target items those steps apply to.	1

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Target Item	TargetItem	TargetItem: Target Item's typically represent a device/machine, or a virtual representation of that hardware. Some examples of Target Item types include routers, switches, vehicles, virtual machines, etc. These are all items which RPS is intended to configure/provision.	1

Developer Terms

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Agile	N/A	Agile software development describes an approach to software development under which requirements and solutions evolve through the collaborative effort of self-organizing and cross-functional teams and their customer(s)/end user(s). It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change.	2
Agile Release Train	ART	The Agile Release Train (ART) is a long-lived team of Agile teams, which, along with other stakeholders, develops and delivers solutions incrementally, using a series of fixed-length Iterations within a Program Increment (PI) timebox. The ART aligns teams to a common business and technology mission.	3
Applicaton Lifecycle Management	ALM	ALM is a set of pre-defined processes that start somewhere in the business as an idea, a need, a challenge or a risk and then pass through different development phases such as requirements definition, design, development, testing, deployment, release and maintenance spanning across an entire lifecycle of a product. Throughout the ALM process, each of these steps is closely monitored and controlled, followed by proper tracking and documentation of any changes to the application.	2
Applicaton Programming Interface	API	In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components.	2
Cmdlets	N/A	Used from within Azure PowerShell to control Azure resources. A cmdlet is a lightweight Windows PowerShell script that performs a single function.	5
Continuous Deployment	CD	Every change that passes automated tests are deployed to production automatically.	3
Continuous Exploration	CE	Continuous Exploration (CE) is the process of continually exploring the market and user needs, and defining a Vision, Roadmap, and set of Features that address those needs. It's the first element in the four-part Continuous Delivery Pipeline, preceding Continuous Integration (CI) Continuous Deployment (CD), and Release on Demand. Copyright © Scaled Agile, Inc.	3
Change Management	ChgM	(ITIL Service Transition) The process responsible for controlling the lifecycle of all changes, enabling beneficial changes to be made with minimum disruption to IT services.	2
Configuration Item	CI	(ITIL Service Transition) Any component or other service asset that needs to be managed in order to deliver an IT service. Information about each configuration item is recorded in a configuration record within the configuration management system and is maintained throughout its lifecycle by service asset and configuration management. Configuration items are under the control of change management. They typically include IT services, hardware, software, buildings, people and formal documentation such as process documentation and service level agreements.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Continuous Integrations	CI	In software engineering, continuous integration (CI) is the practice of merging all developer working copies to a shared mainline several times a day.	3
Domain Certificate Authority	DCA	This is a component of the Tactical Network Initialization & Configuration (TNIC) provided name convention for each Certificate Authority within the Mission Network PoR.	4
Epic	N/A	An Epic is a container for a Solution development initiative large enough to require analysis, the definition of a Minimum Viable Product (MVP), and financial approval prior to implementation. Implementation occurs over multiple Program Increments (PIs) and follows the Lean startup 'build-measure-learn' cycle.	3
GitHub	N/A	GitHub is a web-based hosting service for version control using git. It is mostly used for computer code. It offers all of the distributed version control and source code management (SCM) functionality of git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project	2
Minimum Viable Product	MVP	A minimum viable product (MVP) is a product with just enough features to satisfy early customers, and to provide feedback for future product development.	3
Program Increment	PI	A Program Increment (PI) is a timebox during which an Agile Release Train (ART) delivers incremental value in the form of working, tested software and systems. PIs are typically 8 – 12 weeks long. The most common pattern for a PI is four development Iterations, followed by one Innovation and Planning (IP) Iteration. Copyright © Scaled Agile, Inc. "	3
Post Implementation Review	PIR	A Post-Implementation Review (PIR) is an assessment and review of the completed working solution. It will be performed after a period of live running, some time after the project is completed. There are three purposes for a Post-Implementation Review: 1. To ascertain the degree of success from the project, in particular, the extent to which it met its objectives, delivered planned levels of benefit, and addressed the specific requirements as originally defined. 2. To examine the efficacy of all elements of the working business solution to see if further improvements can be made to optimise the benefit delivered. 3. To learn lessons from this project, lessons which can be used by the team members and by the organisation to improve future project work and solutions."	2
Risk Management Framework	RMF	"The Risk Management Framework is a United States federal government policy and standards to help secure information systems (computers and networks) developed by National Institute of Standards and Technology. The two main publications that cover the details of RMF are NIST Special Publication 800-37, ""Guide for Applying the Risk Management Framework to Federal Information Systems"", and NIST Special Publication 800-53, ""Security and Privacy Controls for Federal Information Systems and Organizations""."	2
Release Train Engineer	RTE	"The Release Train Engineer (RTE) is a servant leader and coach for the Agile Release Train (ART). The RTE's major responsibilities are to facilitate the ART events and processes and assist the teams in delivering value. RTEs communicate with stakeholders, escalate impediments, help manage risk, and drive relentless improvement. Copyright © Scaled Agile, Inc. "	3
SAFe Agile Team	N/A	"The SAFe Agile Team is a cross-functional group of 5 to 11 people who have the responsibility to define, build, test, and where applicable deploy, some element of solution value—all in a short Iteration timebox. Specifically, the SAFe Agile Team incorporates the Dev Team, Scrum Master, and Product Owner roles. Copyright © Scaled Agile, Inc. "	3

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Scrum	N/A	Scrum is an agile framework for managing work with an emphasis on software development. It is designed for teams of three to nine developers who break their work into actions that can be completed within timeboxed iterations, called Sprints (30 days or less) and track progress and re-plan in 15-minute stand-up meetings, called Daily Scrums.	2
System Demo	N/A	"The System Demo is a significant event that provides an integrated view of new Features for the most recent Iteration delivered by all the teams in the Agile Release Train (ART). Each demo gives ART stakeholders an objective measure of progress during a Program Increment (PI). Copyright © Scaled Agile, Inc. "	3
System of Systems	SOS	System of systems is a collection of task-oriented or dedicated systems that pool their resources and capabilities together to create a new, more complex system which offers more functionality and performance than simply the sum of the constituent systems.	4
Visual Studio	VS	"Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as web sites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code. Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source control systems (like Subversion) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer)."	2

Administrator Terms

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Address Resolution Protocol (For IPv4)	ARP	The Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given network layer address, typically an IPv4 address	4
Assured Compliance Assessment Solution	ACAS	The Assured Compliance Assessment Solution (ACAS) is an integrated software solution that provides automated network vulnerability scanning, configuration assessment, and network discovery. ACAS consists of a suite of products to include the Security Center, Nessus Scanner and the Nessus Network Monitor (formerly the Passive Vulnerability Scanner)	2
Automated Comms Engineering Software	ACES	ACES is real-time, tactical network and planning software. ACES supports the Soldier Radio Waveform (SRW) and the Adaptive Wideband Networking Waveform (ANW2). ACES assists the operator with planning, engineering, delivering and managing radio waveform files via an XML format.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Active Directory	AD	Active Directory (AD) is a directory service that Microsoft developed for Windows domain networks. It is included in most Windows Server operating systems as a set of processes and services. Initially, Active Directory was only in charge of centralized domain management. Starting with Windows Server 2008, however, Active Directory became an umbrella title for a broad range of directory-based identity-related services. A server running Active Directory Domain Services (AD DS) is called a domain controller. It authenticates and authorizes all users and computers in a Windows domain type network—assigning and enforcing security policies for all computers and installing or updating software. For example, when a user logs into a computer that is part of a Windows domain, Active Directory checks the submitted password and determines whether the user is a system administrator or normal user. Also, it allows management and storage of information, provides authentication and authorization mechanisms, and establishes a framework to deploy other related services: Certificate Services, Federated Services, Lightweight Directory Services and Rights Management Services."	2
Azure	N/A	Microsoft Azure (formerly Windows Azure) /'æzər/ is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through a global network of Microsoft-managed data centers. It provides software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS) and supports many different programming languages, tools and frameworks, including both Microsoft-specific and third-party software and systems.	2
Battle Command Sustainment Support Sys.	BCS3	The Battle Command Sustainment & Support System (BCS3) integrates multiple data sources into one program and provides commanders with a visual layout of battlefield logistics	4
Blob Storage	N/A	Azure Blob storage is a service that stores unstructured data in the cloud as objects/blobs. Blob storage can store any type of text or binary data, such as a document, media file, or application installer.	2
Blue Force Tracker	BFT	Blue force tracking is a United States military term for a GPS-enabled capability that provides military commanders and forces with location information about friendly military forces. In NATO military symbology, blue typically denotes friendly forces. The capability provides a common picture of the location of friendly forces and therefore is referred to as the blue force tracker	4
BFT Gateway Node	BGN	Blue Force Tracker Gateway Node	4
Background Intelligent Transfer Service	BITS	BITS (Background Intelligent Transfer Service), managed through Windows PowerShell cmdlets, is a Windows component which facilitates asynchronous, prioritized, and throttled transfer of files between machines using idle network bandwidth.	5
Baseband Processing Unit (HNR)	BPU	A baseband processor (also known as baseband radio processor, BP, or BBP) is a device (a chip or part of a chip) in a network interface that manages all the radio functions (all functions that require an antenna)	4
Cloud computing	N/A	Cloud computing, also on-demand computing, is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand.	2
Configuration Management Database	CMDB	(ITIL Service Transition) A database used to store configuration records throughout their lifecycle. The configuration management system maintains one or more configuration management databases, and each database stores attributes of configuration items, and relationships with other configuration items.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Distributed File System Replication	DFSR	Refers to Distributed File System Replication cmdlets and is executed from within Windows PowerShell	5
Dynamic Host Configuration Protocol	DHCP	The Dynamic Host Configuration Protocol (DHCP) is a network management protocol used on TCP/IP networks whereby a DHCP server dynamically assigns an IP address and other network configuration parameters to each device on a network so they can communicate with other IP networks. A DHCP server enables computers to request IP addresses and networking parameters automatically from the Internet service provider (ISP), reducing the need for a network administrator or a user to manually assign IP addresses to all network devices. In the absence of a DHCP server, a computer or other device on the network needs to be manually assigned an IP address, or to assign itself an APIPA address, which will not enable it from communicating outside its local subnet.	2
Defense Information Systems Network	DISN	The Defense Information System Network has been the United States Department of Defense's enterprise network for providing data, video and voice services for 40 years. The Defense Information System Network is a worldwide-protected telecommunications network that enables the exchange of information in an interoperable and global space, partitioned by security demands, transmission requirements, and geographic needs of targeted end-user communities. The DISN offers a selection of integrated standards-based services to fulfill these connectivity needs. The services provide Defense Information Systems Agency mission partners with capability options to support diverse telecommunication requirements for organizations focused on, but not limited to, the Department of Defense	2
Definitive Media Library	DML	(ITIL Service Transition) One or more locations in which the definitive and authorized versions of all software configuration items are securely stored. The definitive media library may also contain associated configuration items such as licences and documentation. It is a single logical storage area even if there are multiple locations. The definitive media library is controlled by service asset and configuration management and is recorded in the configuration management system.	2
Domain Name Server	DNS	"A name server is a computer application that implements a network service for providing responses to queries against a directory service. It translates an often humanly meaningful, text-based identifier to a system-internal, often numeric identification or addressing component. This service is performed by the server in response to a service protocol request. An example of a name server is the server component of the Domain Name System (DNS), one of the two principal namespaces of the Internet. The most important function of DNS servers is the translation (resolution) of human-memorable domain names and hostnames into the corresponding numeric Internet Protocol (IP) addresses, the second principal name space of the Internet which is used to identify and locate computer systems and resources on the Internet."	2
Desired State Configuration	DSC	DSC gives us a declarative model for system configuration management. What that really means is that we can specify how we want a workstation or server (a 'node') to be configured and we leave it to PowerShell and the Windows Workflow engine to make it happen on those target 'nodes'. We don't have to specify how we want it to happen. The main advantages of DSC are: to simplify your sysadmin task by configuring one or more devices automatically, to be able to configure machines identically with the aim to standardise them, to ensure, at a given time, that the configuration of a machine always be identical to its initial configuration, so as to avoid drift, deployment on demand as a Cloud strategy, or 'en masse', is largely automated and simplified"	2
Domain Services Controller	DSC	This is a component of the Tactical Network Initialization & Configuration (TNIC) provided name convention for each Active Directory Domain Controller within the Mission Network PoR.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Early Life Support	ELS	(ITIL Service Transition) A stage in the service lifecycle that occurs at the end of deployment and before the service is fully accepted into operation. During early life support, the service provider reviews key performance indicators, service levels and monitoring thresholds and may implement improvements to ensure that service targets can be met. The service provider may also provide additional resources for incident and problem management during this time.	2
Emergency Change Advisory Board	ECAB	(ITIL Service Transition) A subgroup of the change advisory board that makes decisions about emergency changes. Membership may be decided at the time a meeting is called, and depends on the nature of the emergency change.	2
Environment Control Unit	ECU	Air Conditioning Unit	4
End User Device	EUD	Typically an NSA Android approved operating system loaded on a commercial device such as a cell phone or tablet, loaded with Army custom software for use by soldiers in the field	4
Incident	N/A	(ITIL Service Operation) An unplanned interruption to an IT service or reduction in the quality of an IT service. Failure of a configuration item that has not yet affected service is also an incident – for example, failure of one disk from a mirror set.	2
Information Assurance	IA	Information assurance (IA) is the practice of assuring information and managing risks related to the use, processing, storage, and transmission of information or data and the systems and processes used for those purposes. Information assurance includes protection of the integrity, availability, authenticity, non-repudiation and confidentiality of user data.	2
Infrastructure as a Service	IaaS	The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components	2
Information Assurance Vulnerability Alert	IAVA	An information assurance vulnerability alert (IAVA) is an announcement of a computer application software or operating system vulnerability notification in the form of alerts, bulletins, and technical advisories identified by DoD-CERT, a division of the United States Cyber Command.	2
Information Assurance Vulnerability Management	IAVM	Process responsible for the management of IAVAs, IAVBs (Information Assurance Vulnerability Bulletins) and their implementation to the baseline.	2
Initial Operational Capability	IOC	Initial operating capability or Initial operational capability (IOC) is the state achieved when a capability is available in its minimum usefully deployable form.	2
Internet of Things	IoT	The interconnection via the Internet of computing devices embedded in everyday objects, enabling them to send and receive data.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Internet Protocol	IP	"The Internet Protocol (IP) is the principal communications protocol in the Internet protocol suite for relaying packets across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet. IP has the task of delivering packets from the source host to the destination host solely based on the IP addresses in the packet headers. For this purpose, IP defines packet structures that encapsulate the data to be delivered. It also defines addressing methods that are used to label the datagram with source and destination information."	2
Known Error Database	KEDB	(ITIL Service Operation) A database containing all known error records. This database is created by problem management and used by incident and problem management. The known error database may be part of the configuration management system, or may be stored elsewhere in the service knowledge management system.	2
Netcentric Waveform	NCW	Given sufficient satellite bandwidth, the NCW connected TCNs and PoPs to the network and provided sufficient data flow while at-the-halt and on-the-move.	2
Network Centric Waveform	NCW	Network Centric Waveform (NCW) capability, is a dynamic robust waveform that optimizes bandwidth and satellite utilization. One of the key attributes of NCW is that it facilitates communication between the at-the-halt WIN-T Increment 1 and the on-the-move Increment 2, increasing interoperability so the two generations of equipment can "talk" seamlessly on the battlefield.	4
Network Operations	NetOps	"NetOps is defined as the operational framework consisting of three essential tasks, Situational Awareness (SA), and Command & Control (C2) that the Commander (CDR) of US Strategic Command (USSTRATCOM), in coordination with DoD and Global NetOps Community, employs to operate, manage and defend the Global Information Grid (GIG) to ensure information superiority for the United States. Tactical Network Operations (NetOps) Management System (TNMS) is a scalable, modular NetOps capability that operates on multiple client or server platforms. The TNMS will facilitate decision making necessary to quickly identify network problems, shift resources, change configurations and coordinate the management of the critical network infrastructure supporting mission command functions."	2
Network Management System	NMS	Application written by Lockheed Martin and General Dynamics (GD) to perform centralized operation, management, and troubleshooting of WIN-T Inc 2 platforms.	4
Network Operations Center	NOC	A network operations center (NOC, pronounced like the word knock), also known as a "network management center", is one or more locations from which network monitoring and control, or network management, is exercised over a computer, telecommunication or satellite network.	2
Network Operations and Security Center	NOSC	Network Operations and Security Center (NOSC). The NOSC supported the unit's network management mission at division and brigade, but needed additional Soldiers and tools at battalion and company.	2
Network Recovery Monitoring	NRM	Linux VM which provides automated command and control of select components on WIN-T vehicles.	4
Network Services Gateway	NSG	Encryption device for JBC-P which enables cross-platform functionality within the JCR / BFT family of systems.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Network Time Protocol	NTP	Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks.	2
Operating Environment Management	OEM	Name for the VM which hosts the NMS application on PoR nodes.	4
Operational Level Agreement	OLA	"(ITIL Continual Service Improvement) (ITIL Service Design) An agreement between an IT service provider and another part of the same organization. It supports the IT service provider's delivery of IT services to customers and defines the goods or services to be provided and the responsibilities of both parties. For example, there could be an operational level agreement: □ Between the IT service provider and a procurement department to obtain hardware in agreed times □ Between the service desk and a support group to provide incident resolution in agreed times. "	2
PowerShell	PS	"Windows PowerShell is a task-based command-line shell and scripting language designed especially for system administration. Built on the .NET Framework, Windows PowerShell helps IT professionals and power users control and automate the administration of the Windows operating system and applications that run on Windows. Built-in Windows PowerShell commands, called cmdlets, let you manage the computers in your enterprise from the command line. Windows PowerShell providers let you access data stores, such as the registry and certificate store, as easily as you access the file system. In addition, Windows PowerShell has an expression parser and a fully developed scripting language."	2
Request for Change	RFC	(ITIL Service Transition) A formal proposal for a change to be made. It includes details of the proposed change, and may be recorded on paper or electronically. The term is often misused to mean a change record, or the change itself.	2
Regional Hub Node	RHN	"Regional Hub Nodes (RHNs) are the largest transport nodes for the Army's tactical network. The five globally-located RHNs enable the Army to deploy forces anywhere in the world in support of contingency operations, disaster relief or national emergency response. The five RHNs are at the upper-most level of the Army's tactical network architecture, and their innovative baseband and satellite communications capabilities enable regionalized reach-back to the Army's global network. They enable the transport of information across the tactical network in and out of theater and around the world. The RHNs operate out of the conflict area and give the Soldier in the field immediate access to secure and non-secure internet and voice communications anywhere on the globe. To provide tactical users with secure, reliable connectivity worldwide, the Army has positioned RHNs in five separate regions: Continental United States (CONUS) East and CONUS West, Central Command, European Command and Pacific Command."	2
Security Technical Implementation Guide	STIG	A Security Technical Implementation Guide (STIG) is a cybersecurity methodology for standardizing security protocols within networks, servers, computers, and logical designs to enhance overall security. These guides, when implemented, enhance security for software, hardware, physical and logical architectures to further reduce vulnerabilities.	2
Tactical Communications Node	TCN	The Tactical Communications Node (TCN) provides the principal backbone element and supports command post operations for the WIN-T Increment 2 network. The TCN provides communication and networking equipment and allows the Soldier the ability to access the network at a variety of security levels. While at-the-halt, the TCN is equipped with a 10 meter, extendable mast to improve line-of-sight connectivity and larger satellite assemblage for high throughput.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Tactical Server Infrastructure	TSI	Tactical Server Infrastructure (TSI) replaces the current disparate server hardware by merging all operational and intelligence functions onto one common set of servers. TSI enables more server capability and delivers a consistent approach for installation and configuration, creating efficiencies in fielding, training and sustainment.	2
Virtual Machine	VM	"In computing, a virtual machine (VM) is an emulation of a computer system. Virtual machines are based on computer architectures and provide functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination. There are different kinds of virtual machines, each with different functions: System virtual machines (also termed full virtualization VMs) provide a substitute for a real machine. They provide functionality needed to execute entire operating systems. A hypervisor uses native execution to share and manage hardware, allowing for multiple environments which are isolated from one another, yet exist on the same physical machine. Modern hypervisors use hardware-assisted virtualization, virtualization-specific hardware, primarily from the host CPUs. Process virtual machines are designed to execute computer programs in a platform-independent environment."	2
Virtual Machine (MDA)	MDA VM	Virtual Machine hosted within an MDA. These VMs are system virtual machines (also termed full virtualization VMs) provide a substitute for a real machine. They provide functionality needed to execute entire operating systems. A hypervisor uses native execution to share and manage hardware, allowing for multiple environments which are isolated from one another, yet exist on the same physical machine. Modern hypervisors use hardware-assisted virtualization, virtualization-specific hardware, primarily from the host CPUs.	4

General Terms

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Advanced Field Artillery Targeting Data System	AFATDS	The Advanced Field Artillery Tactical Data System (AFATDS) is the Fire Support Command and Control (C2) system employed by the U.S. Army and U.S. Marine Corps units to provide automated support for planning, coordinating, controlling and executing fires and effects. AFATDS prioritizes targets received from various sensors and performs attack analysis using situational data combined with commander's guidance. The result is timely, accurate and coordinated fire support options to engage targets using Army, Marine, Navy and Air Force weapon systems. The system provides complete flexibility to manage attacks on preplanned and time-sensitive targets. AFATDS supports weapon systems such as mortars, field artillery cannons, rockets, close air support, attack helicopters, and Naval Surface Fire Support (NSFS) systems. AFATDS also acts as a fire support ""server"" to LAN-based and Tactical Internet-based clients, including the AFATDS Effects Management Tool (EMT), and the USMC Command and Control Personal Computer (C2PC) EMT. AFATDS is used in all U.S. Army echelons from weapons platoon to corps and in the Marine Corps from firing battery to Marine Expeditionary Forces. AFATDS is installed aboard the U.S. Navy LHA/LHD Class big deck amphibious ships to support Expeditionary Strike Groups (ESGs) for amphibious operations	4
Army Battle Command Systems	ABCS	The Army Battle Command System (ABCS) is a digital Command, Control, Communications, Computers and Intelligence (C4I) system for the US Army. It includes a mix of fixed/semi-fixed and mobile networks. It is also designed for interoperability with US and Coalition C4I systems.	4
Anti Jam	AJ	Used or intended to inhibit or prevent electronic jamming	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Adaptive Networking Wideband Waveform	ANW2	"The ANW2 provides a data link between the TOC and vehicles, with up to 30 radios in a subnet. It provides simultaneous IP data and voice with integral situational awareness. ANW2 is a self-healing, robust, IP-driven waveform that maximizes redundancy and limits single point failures and networks"	4
Army National Guard	ARNG	The Army National Guard (ARNG), in conjunction with the Air National Guard, is a militia force and a federal military reserve force of the United States. They are simultaneously part of two different organizations, the Army National Guard of the several states, territories and the District of Columbia (also referred to as the Militia of the United States), and the Army National Guard of the United States. The Army National Guard is divided into subordinate units stationed in each of the 50 states, three territories, and the District of Columbia, and operates under their respective governors.	2
At the Halt	ATH	Not in motion	2
Brigade Combat Team	BCT	The brigade combat team (BCT) is the basic deployable unit of maneuver in the US Army. A brigade combat team consists of one combat arms branch maneuver brigade, and its assigned support and fire units. "The Brigade is normally commanded by a Colonel (O-6) although in some cases a Brigadier General (O-7) may assume command." A brigade combat team contains combat support and combat service support units necessary to sustain its operations away from its parent division. BCTs contain organic artillery support, formerly received from the division artillery (DIVARTY).	2
Battlefield Surveillance Brigade	BFSB	"The battlefield surveillance brigade (BFSB) was a United States Army surveillance/reconnaissance formation introduced from 2006–2015. The United States Army planned for the creation and transformation of nine intelligence brigades to a 'battlefield surveillance' role in 2007. The first battlefield surveillance brigade was deployed the same year conducting Surveillance, Reconnaissance and Intelligence operations. However, gathering information is only a part of the challenge it faces. Along with the structural changes and intelligence capabilities, the sustainment capabilities of the brigade also changed. The United States Army is currently reorganizing these BFSB formations into expeditionary military intelligence brigades. These brigades were designed to be self-sufficient Army modular forces."	2
Beyond Line of Sight	BLOS	Beyond Line-Of-Sight (BLOS) is a related term often used in the military to describe radio communications capabilities that link personnel or systems too distant or too fully obscured by terrain for LOS communications.	2
Command and Control	C2	Command and Control (C2) systems enable information superiority on the battlefield. They provide the commander with the information to make effective decisions and they provide the warfighter the capability to access the information necessary to complete their mission.	2
Command and Control Registry	C2R	The Command and Control Registry (C2R) is the Address Book used by today's Army. It dynamically coordinates and collaborates command and control naming, addressing, network and operations data across many different types of military systems deployed globally. It is utilized by our Army's systems as the on-line repository for addressing information such as email addresses, military addresses and network information	4
Command Control Communications Tactical	C3T	Program Executive Office Command, Control, Communications-Tactical (PEO C3T) develops, acquires, fields and supports the Army's tactical network, a critical priority that brings information dominance to current and future Soldiers. The mobile tactical network delivered by PEO C3T provides capability giving commanders a resilient, redundant, easy-to-use and mobile interoperable tactical network.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance	C4ISR	The Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance Evaluation Directorate (C4ISRED) focuses on systems in the areas of mission command and integration, network, information and enterprise, databases and software, Intelligence, Surveillance and Reconnaissance, and intelligence electronic warfare.	2
Command, Control, Communications, Computers, Combat Systems, Intelligence, Surveillance, and Reconnaissance.	C5ISR	C4ISR has recently changed to C5ISR. C5ISR stands for Command, Control, Communications, Computers, Combat Systems, Intelligence, Surveillance, and Reconnaissance.	1
Change Advisory Board	CAB	(ITIL Service Transition) A group of people that support the assessment, prioritization, authorization and scheduling of changes. A change advisory board is usually made up of representatives from: all areas within the IT service provider; the business; and third parties such as suppliers.	2
Combat Aviation Brigades	CABs	A Combat aviation brigade (CAB) is a multi-functional brigade-sized unit in the United States Army that fields military helicopters, offering a combination of attack/reconnaissance helicopters (AH-64 Apache), medium-lift helicopters (UH-60 Black Hawk), heavy-lift helicopters (CH-47 Chinook), and MEDEVAC capability.	2
Common Access Card	CAC	The Common Access Card, also commonly referred to as the CAC or CAC card, is a smart card about the size of a credit card. It is the standard identification for Active Duty United States Defense personnel, to include the Selected Reserve and National Guard, United States Department of Defense (DoD) civilian employees, United States Coast Guard (USCG) civilian employees and eligible DoD and USCG contractor personnel. It is also the principal card used to enable physical access to buildings and controlled spaces, and it provides access to defense computer networks and systems. It also serves as an identification card under the Geneva Conventions (esp. the Third Geneva Convention). In combination with a personal identification number, a CAC satisfies the requirement for two-factor authentication: something the user knows combined with something the user has. The CAC also satisfies the requirements for digital signature and data encryption technologies: authentication, integrity and non-repudiation.	2
Capital Expense	CapEx	Capital expenditure or capital expense (capex) is the money a company spends to buy, maintain, or improve its fixed assets, such as buildings, vehicles, equipment, or land. It is considered a capital expenditure when the asset is newly purchased or when money is used towards extending the useful life of an existing asset, such as repairing the roof.	2
Cryptographic Ignition Key	CIK	The CIK is a small device which can be loaded with a 128-bit sequence which is different for each user. When the device is removed from the machine, that sequence is automatically added (mod 2) to the unique key in the machine, thus leaving it stored in encrypted form. When it is reattached, the unique key in the machine is decrypted, and it is now ready to operate in the normal way. The analogy with an automobile ignition key is close, thus the name. If one loses lose that key, they are still ok. unless the finder (or thief) can match it with their machine. One gets a new CIK, effectively changing the lock in the cipher machine, and gets back in business.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Configuration Management System	CMS	(ITIL Service Transition) A set of tools, data and information that is used to support service asset and configuration management. The CMS is part of an overall service knowledge management system and includes tools for collecting, storing, managing, updating, analysing and presenting data about all configuration items and their relationships. The CMS may also include information about incidents, problems, known errors, changes and releases. The CMS is maintained by service asset and configuration management and is used by all IT service management processes. See also configuration management database.	2
Commanding Officer	CO	The commanding officer or, if the incumbent is a general officer, commanding general, is the officer in command of a military unit. The commanding officer has ultimate authority over the unit, and is usually given wide latitude to run the unit as they see fit, within the bounds of military law. In this respect, commanding officers have significant responsibilities, duties, and powers.	2
Company	Co	A company is a military unit, typically consisting of 80–150 soldiers and usually commanded by a major or a captain. Most companies are formed of three to six platoons, although the exact number may vary by country, unit type, and structure.	2
Colorless (Enclave)	CO	The reason it is called "colorless" is that the Army often places color codes on certain security enclaves, with secret typically being designated as red and unclassified as black. Unlike previous enclaves, in the colorless core all of the data is encrypted, so no one can tell whether the information is secret or unclassified; the "color" cannot be identified. Unclassified information is just as hard to obtain as secret.	4
Control Objectives for Information and Related Technology	COBIT	COBIT (Control Objectives for Information and Related Technologies) is a good-practice framework created by international professional association ISACA for information technology (IT) management and IT governance. COBIT provides an implementable "set of controls over information technology and organizes them around a logical framework of IT-related processes and enablers.	2
Common Operating Environment	COE	COE provides standards to unite existing programs and new technologies on a common software foundation, simplifying development, integration, training and sustainment.	2
Communications Security	COMSEC	Communications security is the discipline of preventing unauthorized interceptors from accessing telecommunications in an intelligible form, while still delivering content to the intended recipients. In the North Atlantic Treaty Organization culture, including United States Department of Defense culture, it is often referred to by the abbreviation COMSEC. The field includes cryptographic security, transmission security, emissions security and physical security of COMSEC equipment and associated keying material. COMSEC is used to protect both classified and unclassified traffic on military communications networks, including voice, video, and data. It is used for both analog and digital applications, and both wired and wireless links."	2
Concept of Operations	CONOPS	"A Concept of Operations (CONOPS) is a verbal or graphic statement of a commander's assumptions or intent in regard to an operation or series of operations as defined by Joint Publication 1-02 DoD Dictionary of Military and Associated Terms. It's designed to give an overall picture of an operation. In Acquisitions, a CONOPS is used to examine current and new and/or proposed capabilities required to solve a current or emerging problem. It describes how a system will be used from the viewpoints of its various stakeholders. This provides a bridge between the often vague capabilities that a project begins with and the specific technical requirements needed to make is successful. A CONOPS is a useful tool that helps the user community write/refine their Initial Capabilities Documents (ICD), System Requirements Document (SRD) and Capabilities Development Documents (CDD)."	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Continental United States	CONUS	The contiguous United States consists of the 48 adjoining U.S. states plus Washington, D.C. (federal district), on the continent of North America. The term excludes the non-contiguous states of Alaska and Hawaii, and all off-shore insular areas.	2
Communities of Practice	CoPs	Communities of Practice (CoPs) are organized groups of people who have a common interest in a specific technical or business domain. They collaborate regularly to share information, improve their skills, and actively work on advancing the general knowledge of the domain.	3
Contracting Officer's Representative	COR	A Contracting Officer's Representative (COR) is an individual authorized in writing by the contracting officer to perform specific technical or administrative contract functions. The COR must receive a written designation of their authority to act on behalf of the contracting officer. The COR is not authorized to make any commitments or changes that will affect price, quality, quantity, delivery, or any other term or condition of the contract.	2
Commercial Off the Shelf	COTS	In the context of the U.S. government, the Federal Acquisition Regulation (FAR) has defined "COTS" as a formal term for commercial items, including services, available in the commercial marketplace that can be bought and used under government contract. For example, Microsoft is a COTS software provider. Goods and construction materials may qualify as COTS but bulk cargo does not. Services associated with the commercial items may also qualify as COTS, including installation services, training services, and cloud services.	2
Command Post of the Future	CPOF	The United States Army's Command Post of the Future (CPOF) is a C2 software system that allows commanders to maintain topsight over the battlefield; collaborate with superiors, peers and subordinates over live data; and communicate their intent. Originally a DARPA technology demonstration, in 2006 CPOF became an Army Program of Record. It is integrated with the Army's Maneuver Control System and other products.	4
Capability Set 16	CS-16	U.S. Army tactical networks are gradually evolving through a series of upgrades called "capability sets" that seek to apply lessons learned on the battlefield.	4
Critical Success Factor	CSF	Something that must happen if an IT service, process, plan, project or other activity is to succeed. Key performance indicators are used to measure the achievement of each critical success factor. For example, a critical success factor of 'protect IT services when making changes' could be measured by key performance indicators such as 'percentage reduction of unsuccessful changes', 'percentage reduction in changes causing incidents' etc.	2
Continual Service Improvement	CSI	(ITIL Continual Service Improvement) A stage in the lifecycle of a service. Continual service improvement ensures that services are aligned with changing business needs by identifying and implementing improvements to IT services that support business processes. The performance of the IT service provider is continually measured and improvements are made to processes, IT services and IT infrastructure in order to increase efficiency, effectiveness and cost effectiveness. Continual service improvement includes the seven-step improvement process. Although this process is associated with continual service improvement, most processes have activities that take place across multiple stages of the service lifecycle. See also Plan-Do-Check-Act.	2
Cipher Text	CT	In cryptography, ciphertext or cyphertext is the result of encryption performed on plaintext using an algorithm, called a cipher. Ciphertext is also known as encrypted or encoded information because it contains a form of the original plaintext that is unreadable by a human or computer without the proper cipher to decrypt it. Decryption, the inverse of encryption, is the process of turning ciphertext into readable plaintext. Ciphertext is not to be confused with codetext because the latter is a result of a code, not a cipher.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Defense Advanced GPS Receiver	DAGR	The Defense Advanced GPS Receiver (DAGR; colloquially, "dagger") is a handheld GPS receiver used by the United States Department of Defense and select foreign military services. It is a military-grade, dual-frequency receiver, and has the security hardware necessary to decode the encrypted P(Y)-code GPS signals.	4
Databridge (CPOF)	DB	"Single point of interface between CPOF and other ABCS systems such as FBCB2/JCR, JBC-P, BCS3, TAIS, DCGS-A, etc. -Supports two-way exchange of data between the various ABCS systems and CPOF. -Runs Microsoft IIS which is used to import images into CPOF system via Drag and Drop or the Snagit application. -Provides A means to import and export graphics in various formats."	4
Distributed Computer Environment	DCE	*Vendor independent distributed computing environment *Not an OS or an application *An integrated set of services and tools that can be installed as a coherent environment on top of an existing OS *Serves as a platform for building and running distributed apps	4
Division	Div	"A division is a large military unit or formation, usually consisting of between 10,000 and 20,000 soldiers. Infantry divisions during the World Wars ranged between 8,000 and 30,000 in nominal strength. In most armies, a division is composed of several regiments or brigades; in turn, several divisions typically make up a corps. Historically, the division has been the default combined arms unit capable of independent operations. Smaller combined arms units, such as the American Regimental combat team (RCT) during World War II, were used when conditions favored them. In recent times, Modern Western militaries have begun adopting the smaller Brigade combat team (similar to the RCT) as the default combined arms unit, with the Division they belong to being less important."	2
Encrypted Data Group (JBC-P Tranceiver)	EDG	"The Joint Battle Command - Platform (JBC-P) program is the cornerstone of joint forces Command and Control (C2) Situational Awareness (SA) and communications. JBC-P provides secure Blue Force Tracking capability in Platforms and Command Posts, providing soldiers and commanders a map-based Common Operating Picture of the battlefield, as a result, reducing fratricide. The JBC-P program fields hardware (vehicle platform computer systems, satellite transceivers, encryption devices, and ancillary equipment), software capabilities, and will continue to leverage the Army's previous equipment investments by installing the new JBC-P software on new hardware as well as existing Force XXI Battle Command Brigade and Below (FBCB2) computer systems. JBC-P serves a primary role as the basis of the Mounted Computing Environment (MCE), one of six (6) environments within the Common Operating Environment (COE) framework. The COE is a standardized set of computing technologies that enable secure and interoperable applications to be rapidly developed and executed across a variety of computing environments. The MCE leverages JBC-P hardware and software to consolidate and integrate multiple warfighting systems in the Platform (Mounted) environment. This integrated MCE, with its open standards, enhanced interoperability, and simplified end-user interface, will speed delivery of the new Mission Command applications to the warfighter while improving the effectiveness and value of current systems."	4
Frequency Division Multiple Access	FDMA	Frequency division multiple access (FDMA) is a channel access method used in multiple-access protocols as a channelization protocol. FDMA gives users an individual allocation of one or several frequency bands, or channels. It is particularly commonplace in satellite communication. FDMA, like other multiple access systems, coordinates access between multiple users.	4
Fires Brigade	FiB	A Fires Brigade (FiB) was a military unit of the United States Army revolved around field artillery. With recent structural changes, Fires Brigades were either inactivated and reflagged as Division Artilleries (DIVARTY) or reorganized and redesignated as Field Artillery Brigades.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Final Operational Capability or Full Operational Capability	FOC	For the United States Department of Defense military acquisition FOC is defined as "in general attained when all units and/or organizations in the force structure scheduled to receive a system have received it and have the ability to employ and maintain it.	2
Field Service Representative	FSR	Contractor personnel tasked with software and hardware maintenance activities at the customers deployed location.	2
Fiscal Year	FY	A fiscal year (or financial year, or sometimes budget year) is a period used for calculating annual ("yearly") financial statements in businesses and other organizations all over the world.	2
Gigabits per second	Gbps	GigaBits or GigaBytes per Second) One billion bits or bytes per second. Gbps is a measurement of peripheral data transfer or network transmission speed. The correct abbreviation is b for bits and B for bytes; however, b and B are often interchanged.	2
Gigahertz (1000 MHz)	GHz	The hertz (symbol: Hz) is the derived unit of frequency in the International System of Units (SI) and is defined as one cycle per second. It is named for Heinrich Rudolf Hertz, the first person to provide conclusive proof of the existence of electromagnetic waves. Hertz are commonly expressed in multiples: 1 GHz = 1*10^9 H.	4
Global Information Grid	GIG	The Global Information Grid (GIG) is an all-encompassing communications project of the United States Department of Defense. It is defined as a "globally interconnected, end-to-end set of information capabilities for collecting, processing, storing, disseminating, and managing information on demand to warfighters, policy makers, and support personnel."	2
Global Positioning System	GPS	"The Global Positioning System (GPS), is a satellite-based radionavigation system owned by the United States government and operated by the United States Air Force. It is a global navigation satellite system that provides geolocation and time information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. Navigation: Soldiers use GPS to find objectives, even in the dark or in unfamiliar territory, and to coordinate troop and supply movement. Target tracking: Various military weapons systems use GPS to track potential ground and air targets before flagging them as hostile.[citation needed] These weapon systems pass target coordinates to precision-guided munitions to allow them to engage targets accurately. Military aircraft, particularly in air-to-ground roles, use GPS to find targets. Missile and projectile guidance: GPS allows accurate targeting of various military weapons including ICBMs, cruise missiles, precision-guided munitions and artillery shells. Embedded GPS receivers able to withstand accelerations of 12,000 g or about 118 km/s ² have been developed for use in 155-millimeter (6.1 in) howitzer shells.[96] Search and rescue. Reconnaissance: Patrol movement can be managed more closely."	4
Graphic Bearing Indicator	GBI	The Nett Warrior system will display a Soldier's GPS position and azimuth on a map using a graphic bearing indicator. It also shows buddy icons of other Nett Warrior users on the network, and any information regarding map locations that has been shared with other users.	4
Highband Networking Radio	HNR	HNR utilizes directive beam technology to achieve a high-throughput mesh network over long distances with enhanced spectrum efficiency. The radio has a number of advanced, multi-layer security features that enable the passing of SCI-level (Sensitive Compartmented Information) data. These include Advanced Encryption Standard (AES) capability, tamper-proof labels, High Assurance Internet Protocol Encryption (HAiPE) compatibility, a directional-beam antenna for Low Probability of Intercept (LPI), and a secure transit case accessory with SIPRNET/NIPRNET network services. The HNR mesh network is a "colorless core" capable of transporting secure and unsecure data over the same network.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Highband Networking Waveform	HNW	A terrestrial time division multiple access (TDMA) line-of-sight radio frequency waveform which provides lower latency and greater throughput for MN PoR vehicles.	4
Highband Radio Frequency Unit	HRFU	A radio unit designed to utilize HNW to pass data over the WAN. Contains multiple feed horns to facilitate on the move connectivity; semi-omnidirectional capable.	4
Integrated Bridge	IB	The central point of control for vehicle systems (electrical power, cameras, FBCB2) on the MAT-V platform. Controls power to all MN components.	4
Intelligence Community	IC	The United States Intelligence Community (IC) is a federation of 16 separate United States government agencies that work separately and together to conduct intelligence activities to support the foreign policy and national security of the United States. Member organizations of the IC include intelligence agencies, military intelligence, and civilian intelligence and analysis offices within federal executive departments. The IC is overseen by the Office of the Director of National Intelligence (ODNI), which itself is headed by the Director of National Intelligence (DNI), who reports to the President of the United States.	2
Identity and Access Management	IdAM	Identity management, also known as identity and access management (IAM) is, in computer security, the security and business discipline that "enables the right individuals to access the right resources at the right times and for the right reasons".	2
Incident Management	IM	(ITIL Service Operation) The process responsible for managing the lifecycle of all incidents. Incident management ensures that normal service operation is restored as quickly as possible and the business impact is minimized	2
Intelligence and Security Command	INSCOM	The United States Army Intelligence and Security Command (INSCOM) is a direct reporting unit that conducts intelligence, security, and information operations for U.S. Army commanders and national decision makers. INSCOM is headquartered at Fort Belvoir, Virginia.	2
Information Technology Information Library	ITIL	A set of best-practice publications for IT service management. Owned by the Cabinet Office (part of HM Government), ITIL gives guidance on the provision of quality IT services and the processes, functions and other capabilities needed to support them. The ITIL framework is based on a service lifecycle and consists of five lifecycle stages (service strategy, service design, service transition, service operation and continual service improvement), each of which has its own supporting publication. There is also a set of complementary ITIL publications providing guidance specific to industry sectors, organization types, operating models and technology architectures. See www.itil-officialsite.com for more information	2
Information Technology Service Management	ITSM	IT service management (ITSM) refers to the entirety of activities – directed by policies, organized and structured in processes and supporting procedures – that are performed by an organization to design, plan, deliver, operate and control information technology (IT) services offered to customers. It is thus concerned with the implementation of IT services that meet customers' needs, and it is performed by the IT service provider through an appropriate mix of people, process and information technology.	2
Joint Automated CEOI System	JACS	Automated Communications Engineering Software/Joint Automated Communications Electronics Operating Instructions (CEOI) System (ACES-JACS) that helps with COMSEC keying, information key tags and signal operating instructions development.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Joint Battle Command - Platform	JBC-P	JBC-P is the Army's next-generation friendly force tracking system, equipping Soldiers with a faster satellite network, secure data encryption and advanced logistics. JBC-P is the Army's next generation friendly force tracking system, equipping Soldiers with a faster satellite network, secure data encryption and advanced logistics. JBC-P includes an intuitive interface with features like touch-to-zoom maps and drag-and-drop icons. JBC-P will be interoperable with the Nett Warrior handheld device, delivering situational awareness capabilities to dismounted Soldiers. JBC-P incorporates the common hardware solution known as the Mounted Family of Computer Systems (MFoCS), standardized tactical computers that are scalable and tailorable to the mission and vehicle. Ranging in options from a detachable tablet to a fully-loaded, vehicle-mounted workstation, MFoCS runs not only JBC-P but can also run other software applications, reducing size, weight and power demands. JBC-P builds on the situational awareness capability known as Force XXI Battle Command Brigade and Below/Blue Force Tracking (FBCB2/BFT), which is integrated on more than 120,000 platforms and is fielded or authorized to every brigade combat team in the Army. "	4
Joint Command, Control, Communications, Computers, Intelligence, Surveillance	JC4ISR	Joint command, control, communications, computers, intelligence, surveillance and reconnaissance	2
Joint Capabilities Release	JCR	JCR is a software upgrade. JCR includes computers, global positioning equipment and communication systems that work in tandem to provide near-real-time information to combat leaders at the tactical level so units are better able to synchronize operations and avoid friendly fire incidents. Soldiers inside vehicles can plot the location of enemy, friendly and neutral objects and exchange command and control messages. They can also alert nearby friendly units of improvised explosive devices or enemy locations.	4
Joint Enterprise Network Manager	JENM	"The Joint Enterprise Network Manager (JENM) is a consolidated software application that plans, loads, manages and secures/defends mid and lower-tier software defined radios and associated waveforms, including: the Soldier Radio Waveform (SRW), Wideband Networking Waveform (WNW), the Mobile User Objective System (MUOS), as well as the Single Channel Ground and Airborne Radio System (SINGARS) and some Satellite Communications. JENM can plan and configure an entire network of disparate networking radios and waveforms, ensuring interoperability across and between echelons. Its new enterprise Over-the-Air Management (eOTAM) capability reduces manpower hours to reconfigure, manage, control a tactical radio network, by performing the tasks rapidly over-the-air. The JENM eOTAM capability reduces the need for Signal Soldiers to travel from location to location, allowing them to manage and configure their radio networks from remote locations, such as the battalion tactical operations cell. Additional improvements to JENM include a more intuitive graphical user interface, simplification in planning tactical networks, network monitoring and troubleshooting, and capability with more software defined radios and respective waveforms."	4
Joint Gateway Node	JGN	The JGN allows WIN-T to connect to a variety of external networks.	2
Joint Tactical Network operations Toolkit	JTNT	Joint Tactical Networking Environment Network Operations Toolkit provides a means to load and configure the Army's family of software-defined radios. Before J-TNT, there were nearly 50 tools for signal Soldiers to plan, manage, monitor and control the Lower Tactical Network Environment (LTNE).	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Joint Tactical Radio System	JTRS	Joint Tactical Radio System (JTRS) is the Department of Defense family of common software-defined programmable radios that will form the foundation of radio frequency information transmission for Joint Vision 2020. JTRS radios are intended to interoperate with existing radio systems and provide the warfighter with additional communications capability to access maps and other visual data, communicate via voice and video and obtain information directly from battlefield sensors. JTRS will provide internet protocol (IP)-based capability to the warfighter and will replace all existing tactical radios based on the Services' migration plans. The JTRS program is built around an open Software Communications Architecture (SCA), allowing common software waveform applications to be implemented across the family of radios to provide joint-service, allied, and coalition interoperability. JTRS is a key enabler that will provide dynamic connectivity throughout the battle space to operate within the network centric operational environment. Activities also include studies and analysis to support both current program planning and execution and future program planning	4
Kilobits per second	Kbps	Kilobit per second is currently defined as 1,000 bits per second.	2
Key Encryption Key	KEK	"DEK: Data Encryption Key KEK: Key Encryption Key Master Key: Generally will describe one of the two above keys. Depending on the scheme in which it is implemented. This type of encryption scheme is often used for secure storage. Microsoft Windows is known to use this type of encryption scheme to protect user credentials and other types of data that are secured for a user. Microsoft generates a Key Encryption Key using the user's password. This KEK is then used to encrypt what they call the Master Key. The Master Key is really a Data Encryption Key. It will be used to encrypt any data that is put in the user's protected storage. Key management for Full Disk Encryption will also work the same way. The FDE software will randomly generate a DEK, then use the user's password/keyfile/smart card to create a KEK in order to encrypt the DEK. This mechanism allows the user to change their password without having to decrypt and re-encrypt the entire volume. Instead, the DEK is just re-encrypted with the new KEK."	4
Kilohertz (1000 Hz)	kHz	The hertz (symbol: Hz) is the derived unit of frequency in the International System of Units (SI) and is defined as one cycle per second. It is named for Heinrich Rudolf Hertz, the first person to provide conclusive proof of the existence of electromagnetic waves. Hertz are commonly expressed in multiples: 1 KHz = 1×10^3 Hz	4
Key Material Identifier (TACLANE)	KMID	Communications Security (COMSEC) term for an identifying marker embedded within a electric key wrapper. Most commonly used with a firefly vector set (FFVS). This lets us identify the exact 'serial number' of an encryption set installed on a device. No two devices can use material with the same KMID.	4
Contracting Officer	KO	A Contracting Officer (CO or KO) is a person who can bind the Federal Government of the United States to a contract that is greater than the Micro-Purchase threshold. This is limited to the scope of authority delegated to the Contracting Officer by the head of the agency.	2
Key Performance Indicator	KPI	(ITIL Continual Service Improvement) (ITIL Service Design) A metric that is used to help manage an IT service, process, plan, project or other activity. Key performance indicators are used to measure the achievement of critical success factors. Many metrics may be measured, but only the most important of these are defined as key performance indicators and used to actively manage and report on the process, IT service or activity. They should be selected to ensure that efficiency, effectiveness and cost effectiveness are all managed.	2
Key Tag Binary (Key Tag File)	KTB	Similar to a KMID but for simple key material (not FFVS).	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Keyboard Video Mouse	KVM	A KVM switch (with KVM being an abbreviation for "keyboard, video and mouse") is a hardware device that allows a user to control multiple computers from one or more sets of keyboards, video monitors, and mice. Although multiple computers are connected to the KVM, typically a smaller number of computers can be controlled at any given time. Modern devices have also added the ability to share other peripherals like USB devices and audio.	4
Local Area Network	LAN	A local area network (LAN) is a group of computers and associated devices that share a common communications line or wireless link to a server. Typically, a LAN encompasses computers and peripherals connected to a server within a distinct geographic area such as an office or a commercial establishment.	2
Line of Service	LOS	(ITIL Service Strategy) A core service or service package that has multiple service options. A line of service is managed by a service owner and each service option is designed to support a particular market segment.	2
Line of Sight	LOS	"The line between the target and the aiming reference. 2. The straight line between two points. This line is in the plane of the great circle, but does not follow the curvature of the earth."	2
Low Probability of Detection	LPD	The result of measures used to hide or disguise intentional electromagnetic transmissions.	2
Low Probability of Interception	LPI	Result of measures to prevent the intercept of intentional electromagnetic transmissions	2
Lower Tactical Internet	LTI	Lower Tactical Internet is radio based internet with it's related hardware, software and tools	4
Media Access Control (Ethernet Address)	MAC	A media access control address of a device is a unique identifier assigned to a network interface controller for communications at the data link layer of a network segment. MAC addresses are used as a network address for most IEEE 802 network technologies, including Ethernet and Wi-Fi. In this context, MAC addresses are used in the medium access control protocol sublayer	4
Mobile Access Router (HNR)	MAR	Embedded Cisco router in the Harris Baseband Processing Unit (BPU)	4
Megabits per second	Mbps	Megabits per second (Mbps) are a unit of measurement for bandwidth and throughput on a network. Each megabit is equal to 1 million bits. Mbps belongs to a family of metrics used to measure the capacity and speed of data transfer.	2
Megabytes per second	MBps	Megabytes per second (MBps) is a measure used to describe data transfer rates between devices. One megabyte is technically equal to 1,048,576 bytes, but in networking it refers to 1 million bytes. MBps should not be confused with the abbreviation Mbps, which refers to megabits per second.	2
Mission Command	MC	Command post and platform that enable mission execution by commanders and leaders at all levels to be more effective, agile and decisive	4
Modular Communications Node - Basic	MCN-B	Modular Communications Node – Basic (MCN-B). The MCN-B allowed the unit to extend subscriber services from an adjacent TCN.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Multi-Domain Atlas	MDA	"The Multi-Domain Atlas (MDA) is a rugged, dismountable vehicle computing platform consisting of an intelligent docking station computer and a dismountable tablet computer. MDA Features: Supports up to four classification domains simultaneously; runs virtual machines to host independent security enclaves. Dock capable of running multiple instances of Windows® and Linux® operating systems simultaneously. Tablet capable of running a single instance of the Windows® or Linux® operating system. Built-in KVM capability provides single display user interface to tablet and dock computers. A system with detachable handheld display (tablet) with up to 50' range; wired or Type-1 encrypted wireless connection to the dock. "	4
Microsoft Deployment Toolkit	MDT	The purpose of MDT is to help automate the deployment of Windows operating systems and applications to desktop, portable, and server computers in the environment. At a high level, MDT automates the deployment process by configuring the unattended Setup files for Windows and packaging the necessary files into a consolidated image file that you then deploy to reference and target computers.	2
Maneuver Enhancement Brigade	MEB	A maneuver enhancement brigade (MEB) is a self-contained, modular, and multifunctional support brigade of the United States Army customized to meet whatever mission it receives. A MEB's primary purpose is to plug into operational formations commanded by corps or division commanders, to support brigade combat teams once deployed, and to conduct tactical level tasks and support.	2
Megahertz (1000 kHz)	MHz	The hertz (symbol: Hz) is the derived unit of frequency in the International System of Units (SI) and is defined as one cycle per second. It is named for Heinrich Rudolf Hertz, the first person to provide conclusive proof of the existence of electromagnetic waves. Hertz are commonly expressed in multiples: MHz = 1*10^6 Hz	4
Military Intelligence	MI	Military intelligence. Military intelligence is a military discipline that uses information collection and analysis approaches to provide guidance and direction to commanders in support of their decisions.	2
Military Standard	MIL-STD	A United States defense standard, often called a military standard, "MIL-STD", "MIL-SPEC", or (informally) "MilSpecs", is used to help achieve standardization objectives by the U.S. Department of Defense.	2
Mean Time Between Failures	MTBF	(ITIL Service Design) A metric for measuring and reporting reliability. MTBF is the average time that an IT service or other configuration item can perform its agreed function without interruption. This is measured from when the configuration item starts working, until it next fails.	2
Mean Time Between Service Incidents	MTBSI	(ITIL Service Design) A metric used for measuring and reporting reliability. It is the mean time from when a system or IT service fails, until it next fails. MTBSI is equal to MTBF plus MTRS.	2
Microsoft Test Manager	MTM	Microsoft Test Manager (MTM) is used to help test the application that has been built. MTM stores test plans and results on Team Foundation Server (TFS). Part of Visual Studio Enterprise and Visual Studio Test Professional.	2
Mean Time to Restore Service	MTRS	The average time taken to restore an IT service or other configuration item after a failure. MTRS is measured from when the configuration item fails until it is fully restored and delivering its normal functionality. See also maintainability; mean time to repair.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Mean Time to Repair	MTTR	The average time taken to repair an IT service or other configuration item after a failure. MTTR is measured from when the configuration item fails until it is repaired. MTTR does not include the time required to recover or restore. It is sometimes incorrectly used instead of mean time to restore service.	2
Metric	N/A	Something that is measured and reported to help manage a process, IT service or activity.	2
Non-Secure Internet Protocol Router	NIPR	See NIPRNET	2
Non-classified but Sensitive Internet Protocol Router Network	NIPRNET	The Non-classified Internet Protocol (IP) Router Network (NIPRNet) is a private IP network used to exchange unclassified information, including information subject to controls on distribution, among the private network's users. The NIPRNet also provides its users access to the Internet.	2
Not Mission Capable	NMC	Condition indicating that systems and equipment are not capable of performing any of their assigned missions.	4
NettWarrior	NW	The Nett Warrior is an integrated dismounted situational awareness (SA) and mission command (MC) system for use during combat operations. Designed as a tool for leaders, NW provides unparalleled SA and MC capabilities to the dismounted leader, permitting faster and more accurate decisions during the tactical fight. With advanced navigation and information sharing capabilities, leaders are able to avoid fratricide and are more effective and more lethal in the execution of their combat missions. The NW program delivers a SA and MC system, which has the ability to graphically display the location of an individual leader's location on a digital geo-referenced map image. Additional Soldier, platform and unit locations are also displayed on the digital user interface. NW is connected through a radio that will send and receive information from one NW to another, thus connecting the dismounted leader to the network. These radios will also connect the equipped leader to higher echelon data and information products to assist in decision making and situational understanding. Soldier position location information will be added to the network via interoperability with the Army's Rifleman Radio capability. All of this will allow the leader to easily see, understand, and interact in the method that best suits the user and the particular mission. NW will employ a system-of-systems approach, optimizing and integrating capabilities while reducing the Soldier's combat load and logistical footprint.	4
Outside Continental United States	OCONUS	Outside the continental limits of the United States	2
Operational Expense	OpEx	The cost resulting from running the IT services, which often involves repeating payments – for example, staff costs, hardware maintenance and electricity (also known as current expenditure or revenue expenditure). See also capital expenditure.	2
Open Shortest Path First	OSPF	Open Shortest Path First (OSPF) is a routing protocol for Internet Protocol (IP) networks. It uses a link state routing (LSR) algorithm and falls into the group of interior gateway protocols (IGPs), operating within a single autonomous system (AS).	4
On the Move	OTM	Military elements in physical motion	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Platform as a Service	PaaS	Platform as a Service (PaaS) or application platform as a Service (aPaaS) or platform base service is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app.	2
Personal Communications Device	PCD	Personally owned device such as a laptop, cell phone, or tablet	2
Plan, Do, Check, Act	PDCA	(ITIL Continual Service Improvement) A four-stage cycle for process management, attributed to Edward Deming. Plan-Do-Check-Act is also called the Deming Cycle. Plan – design or revise processes that support the IT services; Do – implement the plan and manage the processes; Check – measure the processes and IT services, compare with objectives and produce reports; Act – plan and implement changes to improve the processes.	2
Platform Encryption Device (KGV-72)	PED	The KGV-72 Type-1 Programmable Encryption Device features a modular architecture with the programmability and scalability to accommodate a wide range of link and Internet Protocol (IP) in-line network encryption applications. Developed as part of the Force XXI Battle Command, Brigade-and-Below (FBCB2) Type-1 Encryption Device (T1ED) Program, the KGV-72 is a high-grade security solution that is compatible with existing and future Blue Force Tracking (BFT) terminals and transceivers. Its flexible, software-upgradeable design supports both legacy FBCB2 L-band/BLOS link communications and evolving IP standards. The KGV-72 is secured with the programmable National Security Agency (NSA)-certified Sierra II™ encryption module, which meets all requirements of the NSA's Cryptographic Modernization initiative. Programming of Suite-A and Suite-B algorithms is also supported, allowing the KGV-72 to be used for a wide range of in-line network applications. The KGV-72 operates seamlessly with installed FBCB2 remote computers and provides high-grade traffic data encryption.	4
Program Executive Office	PEO	A program executive office may be responsible for a specific program (e.g., the Joint Strike Fighter), or for an entire portfolio of similar programs (e.g., the Air Force PEO for space, who is responsible for all acquisition programs at the Air Force Space Command Space and Missile Systems Center, or the Navy PEO for aircraft carriers).	2
Performance Enhancing Proxy (WIN-T)	PEP	Performance-enhancing proxies (PEPs) are network agents designed to improve the end-to-end performance of some communications protocol. PEP standards are defined in RFC 3135 (PEPs intended to mitigate link-related degradations) and RFC 3449 (TCP performance implications of network path asymmetry).	4
Position Location Information	PLI	Data in a Joint Variable Message Format (JVMF) which provides the ability to determine an accurate location on the Earth.	4
Project Management Office	PMO	A project management office, abbreviated to PMO, is a group or department within a business, agency or enterprise that defines and maintains standards for project management within the organization.	2
Point of Presence	PoP	The Point of Presence (PoP) is installed on select combat platforms at division, brigade and battalion echelons (to include select vehicles), enabling mobile mission command by providing on-the-move network connectivity, both line-of-sight and beyond-line-of-sight. A point of presence (PoP) is an artificial demarcation point or interface point between communicating entities. An Internet point of presence typically houses servers, routers, network switches, multiplexers, and other network interface equipment.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Program of Record	PoR	The term is used to describe a program that is funded (approved) across the FYDP, (Future Year Defense Program) through the POM (Program Objective Memorandum). When this happens, the program becomes a "line item record" in the budget -- hence the term "program of record"	2
Personal Radio Communications	PRC	The AN/PRC-155 Manpack radio is the first fielded two-channel, software defined radio capable of network-centric connectivity and legacy interoperability, supporting advanced (SRW, MUOS) and current-force waveforms (including SINGARS and SATCOM). In addition, the AIM II embedded programmable security allows the dismounted warfighter to seamlessly join any combat net with confidence. Added functionality, like internal voice and data bridging between networks, makes this the most powerful piece of tactical equipment in the soldier's communications arsenal – joining local networks to beyond line of sight networks.	4
Quality Assurance	QA	(ITIL Service Transition) The process responsible for ensuring that the quality of a service, process or other service asset will provide its intended value. Quality assurance is also used to refer to a function or team that performs quality assurance. This process is not described in detail within the core ITIL publications. See also service validation and testing.	2
QOS Edge Device	QED	In general, edge devices are normally routers that provide authenticated access (most commonly PPPoA and PPPoE) to faster, more efficient backbone and core networks. The trend is to make the edge device smart and the core device(s) "dumb and fast", so edge routers often include Quality of Service (QoS) and multi-service functions to manage different types of traffic. Consequently, core networks are often designed with switches that use routing protocols such as Open Shortest Path First (OSPF) or Multiprotocol Label Switching (MPLS) for reliability and scalability, allowing edge routers to have redundant links to the core network. Links between core networks are different, for example Border Gateway Protocol (BGP) routers often used for peering exchanges.	4
Quality Management System	QMS	(ITIL Continual Service Improvement) The framework of policy, processes, functions, standards, guidelines and tools that ensures an organization is of a suitable quality to reliably meet business objectives or service levels. See also ISO 9000.	2
Quality Of Service	QOS	"Quality of service (QoS) is the description or measurement of the overall performance of a service, such as a telephony or computer network or a cloud computing service, particularly the performance seen by the users of the network. To quantitatively measure quality of service, several related aspects of the network service are often considered, such as packet loss, bit rate, throughput, transmission delay, availability, jitter, etc. In the field of computer networking and other packet-switched telecommunication networks, quality of service refers to traffic prioritization and resource reservation control mechanisms rather than the achieved service quality. Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow. Quality of service is particularly important for the transport of traffic with special requirements. In particular, developers have introduced Voice over IP technology to allow computer networks to become as useful as telephone networks for audio conversations, as well as supporting new applications with even stricter network performance requirements.	4
Responsible, Accountable, Consulted, Informed	RACI	(ITIL Service Design) A model used to help define roles and responsibilities. RACI stands for responsible, accountable, consulted and informed.	2
Root Cause Analysis	RCA	(ITIL Service Operation) An activity that identifies the root cause of an incident or problem. Root cause analysis typically concentrates on IT infrastructure failures. See also service failure analysis.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Radio Frequency	RF	Radio frequency (RF) refers to alternating (AC) electric current or radio waves, oscillating in the frequency range used in radio, extending from around twenty thousand times per second (20 kHz) to around three hundred billion times per second (300 GHz), roughly between the upper limit of audio frequencies and the lower limit of infrared frequencies.	4
Sustainment System Mission Command	S2MC	On 6/20/2014, Sustainment System Mission Command (S2MC), the organization that developed Battle Command Sustainment Support System (BCS3) and associated logistics command and control products, was retired. All current and future development efforts will be web-based, placing the software and hardware systems into sustainment.	4
Situational Awareness	SA	On the ground during combat, situational awareness is the ability to see what's in the vicinity and anticipate what's not — knowledge that can mean the difference between surviving or being killed in action. Situational awareness is also the integrated web of networks, servers, storage devices, and analysis and management software that ingests data, makes it available for analysis, and then shares it anytime and anywhere, up and down the chain of command.	4
Software as a Service	SaaS	SaaS provides a complete software solution that you purchase on a pay-as-you-go basis from a cloud service provider. You rent the use of an app for your organization, and your users connect to it over the Internet, usually with a web browser. All of the underlying infrastructure, middleware, app software, and app data are located in the service provider's data center. The service provider manages the hardware and software, and with the appropriate service agreement, will ensure the availability and the security of the app and your data as well. SaaS allows your organization to get quickly up and running with an app at minimal upfront cost	2
Service Acceptance Criteria	SAC	(ITIL Service Transition) A set of criteria used to ensure that an IT service meets its functionality and quality requirements and that the IT service provider is ready to operate the new IT service when it has been deployed. See also acceptance.	2
Service Access and Configuration Management	SACM	(ITIL Service Transition) The process responsible for ensuring that the assets required to deliver services are properly controlled, and that accurate and reliable information about those assets is available when and where it is needed. This information includes details of how the assets have been configured and the relationships between assets. See also configuration management system.	2
Scaled Agile Framework	SAFe	"SAFe synchronizes alignment, collaboration, and delivery for multiple Agile teams. Scalable and configurable, SAFe allows each organization to adapt it to its own business needs. It supports smaller-scale solutions employing 50 – 125 practitioners, as well as complex systems that require thousands of people. Copyright © Scaled Agile, Inc. "	3
Satellite Communications	SATCOM	Product Manager Satellite Communications (PdM SATCOM) rapidly designs, acquires, fields and supports fully integrated, easy to operate and cost effective tactical SATCOM and services that meet Joint network communications requirements around the world. As part of the Army's holistic One Tactical Network, most of these terminals transmit voice, video and data over the Warfighter Information Network-Tactical (WIN-T) backbone. With terminals ranging in size from a softside carry-on to a small house, the Army's global network of SATCOM capability provides interoperable high-speed, high-capacity connectivity, so Soldiers can communicate across vast distances and in austere locations and terrains, virtually anytime, anywhere.	2
Signal Battalion	SB	Provides Command, Control, Communications and Computer (C4) support to it's parent Signal Brigade.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Software Communications Architecture	SCA	The Software Communications Architecture (SCA) is an open architecture framework that defines a standard way for radios to instantiate, configure, and manage waveform applications running on their platform. The SCA separates waveform software from the underlying hardware platform, facilitating waveform software portability and re-use to avoid costs of redeveloping waveforms.	2
Service Design Package	SDP	(ITIL Service Design) Document(s) defining all aspects of an IT service and its requirements through each stage of its lifecycle. A service design package is produced for each new IT service, major change or IT service retirement.	2
Smart Display Unit	SDU	A ruggedized video display unit mounted in a military vehicle. Programmable buttons along the bezel facilitate different functionality between running applications.	4
Secure Internet Protocol Router	SIPR	See SIPRNET	2
Secure Internet Protocol Network	SIPRNET	The Secret Internet Protocol Router Network (SIPRNet) is "a system of interconnected computer networks used by the U.S. Department of Defense and the U.S. Department of State to transmit classified information (up to and including information classified SECRET) by packet switching over the 'completely secure' environment".	2
Simple Key Loader (AN/PYQ-10)	SKL	The AN/PYQ-10 Simple Key Loader (SKL) is a ruggedized, portable, hand-held fill device, for securely receiving, storing, and transferring data between compatible cryptographic and communications equipment.	4
Service Level Agreement	SLA	(ITIL Continual Service Improvement) (ITIL Service Design) An agreement between an IT service provider and a customer. A service level agreement describes the IT service, documents service level targets, and specifies the responsibilities of the IT service provider and the customer. A single agreement may cover multiple IT services or multiple customers. See also operational level agreement.	2
Service Level Management	SLM	(ITIL Service Design) The process responsible for negotiating achievable service level agreements and ensuring that these are met. It is responsible for ensuring that all IT service management processes, operational level agreements and underpinning contracts are appropriate for the agreed service level targets. Service level management monitors and reports on service levels, holds regular service reviews with customers, and identifies required improvements.	2
Service Level Requirement	SLR	(ITIL Continual Service Improvement) (ITIL Service Design) A customer requirement for an aspect of an IT service. Service level requirements are based on business objectives and used to negotiate agreed service level targets.	2
Service Level Target	SLT	(ITIL Continual Service Improvement) (ITIL Service Design) A commitment that is documented in a service level agreement. Service level targets are based on service level requirements, and are needed to ensure that the IT service is able to meet business objectives. They should be SMART, and are usually based on key performance indicators.	2
Specific, Measurable, Achievable, Relevant and Time-bound	SMART	(ITIL Continual Service Improvement) (ITIL Service Design) An acronym for helping to remember that targets in service level agreements and project plans should be specific, measurable, achievable, relevant and time-bound.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Service Management Office	SMO	The Service Management Office (SMO) is a relatively new concept which is directly analogous to the Project Management Office (PMO). Like the PMO, the SMO provides a centre of excellence within the organization to drive efficiency and effectiveness.	2
Soldier Network Extension	SNE	The Soldier Network Extension (SNE), delivers the Army's mobile tactical network backbone to the company level, and is evolving from a vehicle used by the company commander to an information hotspot allowing other Soldiers to plug in, make phone calls and send and receive data from anywhere on the battlefield. The Soldier Network Extension (SNE) is installed on select vehicles to provide on-the-move network communications to extend the network. Using its on-the-move satellite communication systems, the SNE can also be used to heal and extend remote tactical radio networks.	2
Special Operations Command	SOCOM	The United States Special Operations Command (USSOCOM or SOCOM) is the Unified Combatant Command charged with overseeing the various Special Operations Component Commands of the Army, Marine Corps, Navy, and Air Force of the United States Armed Forces. The command is part of the Department of Defense and is the only Unified Combatant Command legislated into being by the U.S. Congress. USSOCOM is headquartered at MacDill Air Force Base in Tampa, Florida.	2
Signal Operating Instructions	SOI	Signal operating instructions or Communications-Electronics Operation Instructions are U.S. military terms for a type of combat order issued for the technical control and coordination of communications within a command. They include current and up-to-date information covering radio call signs and frequencies, a telephone directory, code-words, and visual and sound signals. A designated battalion signal officer prepares the battalion SOI in conformance with the SOI of higher headquarters. Units maintained 2 copies of the SOI: a training version and a "go-to-war" version. During operations, SOI are changed daily. Since the fielding of the SINCGARS system, however, the paper SOI has generally faded from Army use. Electronic SOI are now generated, distributed and loaded along with cryptographic keys.	4
Standard Operating Procedure	SOP	A standard operating procedure is a set of instructions covering those features of operations which lend themselves to a definite or standardized procedure without loss of effectiveness	2
Soldier Radio Waveform	SRW	The SRW provides networked wideband communications that enable simultaneous, integrated combat net radio voice, data and video capabilities. Designed as a mobile ad hoc waveform, the SRW functions as a "node" or "router" within a radio network and transmits vital information across large distances and over elevated terrain, such as mountains. The SRW is used by individual Soldiers, small units and very small sensors such as unattended ground or air vehicles, and it enables communication without a "fixed" infrastructure such a cell tower or satellite network.	4
Satellite Transportable Terminal	STT	The Satellite Transportable Terminal (STT) is a highly transportable and mobile satellite system, which operates in conjunction with the JNN and BnCPN, designed to establish secure voice, video and data communications virtually anytime and anywhere.	2
Sustainment Brigades	Sust Bdes	The sustainment brigade is designed to provide mission command for combat support and combat service support units. It can be adjusted in size to support anywhere from one to ten brigade combat teams (BCTs). A sustainment brigade has a joint capability that allows the Army to better manage the flow of logistics into the area of operations (AO) and provides support to other services for common logistics like fuel, common ammo, medical supplies, repair parts of wheeled vehicles, and so forth. A sustainment brigade is designed to operate independently in a theater of operations, in conjunction with other sustainment brigades under the command of a sustainment command (expeditionary), or directly under a theater sustainment command. When in theater, a sustainment command (expeditionary) will report to the theater sustainment command.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
TACLANE Configuration Tool	TCT	Manufacturer provided software application which automated the configuration of TACLANE devices.	4
Time Division Multiple Access	TDMA	Time-division multiple access (TDMA) is a channel access method for shared-medium networks. It allows several users to share the same frequency channel by dividing the signal into different time slots. The users transmit in rapid succession, one after the other, each using its own time slot. This allows multiple stations to share the same transmission medium (e.g. radio frequency channel) while using only a part of its channel capacity. TDMA is used in the digital 2G cellular systems such as Global System for Mobile Communications (GSM), IS-136, Personal Digital Cellular (PDC) and iDEN, and in the Digital Enhanced Cordless Telecommunications (DECT) standard for portable phones. It is also used extensively in satellite systems, combat-net radio systems, and passive optical network (PON) networks for upstream traffic from premises to the operator.	4
Transmission Encryption Key	TEK	Traffic encryption key (TEK)/data encryption key (DEK) - a symmetric key that is used to encrypt messages. TEKs are typically changed frequently, in some systems daily and in others for every message. See session key. DEK is used to specify any data form type (in communication payloads or anywhere else).	4
Team Foundation Server	TFS	Team Foundation Server (commonly abbreviated to TFS) is a Microsoft product which provides source code management (either via Team Foundation Version Control or Git), reporting, requirements management, project management (for both agile software development and waterfall teams), automated builds, lab management, testing and release management capabilities.	2
Trivial File Transfer Protocol	TFTP	Trivial File Transfer Protocol (TFTP) is a simple lockstep File Transfer Protocol which allows a client to get a file from or put a file onto a remote host. One of its primary uses is in the early stages of nodes booting from a local area network. TFTP has been used for this application because it is very simple to implement.	4
Time Of Day	TOD	A time-of-day (ToD) port on the front panel of the router allows you to connect external timing signal sources. The external timing input port is labeled TOD. (From RFC 868: This protocol provides a site-independent, machine readable date and time. The Time service sends back to the originating source the time in seconds since midnight on January first 1900.)	4
Transmission Key Encryption Key	TRKEK	COMSEC key used to encrypt over-the-air-rekey (OTAR) messages.	4
Tactical Relay Tower (HNR)	TRT	WIN-T Inc 2 (PoR MN) coonfiguration item designed for range extension of HNW radio networks.	4
Transmission Security Key	TSK	Seed for a pseudorandom number generator that is used to control a radio in frequency hopping or direct-sequence spread spectrum modes	4
Ultra High Frequency	UHF	Ultra high frequency (UHF) is the ITU designation for radio frequencies in the range between 300 megahertz (MHz) and 3 gigahertz (GHz), also known as the decimetre band as the wavelengths range from one meter to one decimeter. Radio waves with frequencies above the UHF band fall into the SHF (super-high frequency) or microwave frequency range.	4

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
User Interface	UI	The user interface (UI), in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.	2
Unit Reference Number	URN	Specific numerical identifier for devices, unit, and persons on the battlefield. This URN is generated and used within Mission Command applications to distinguish certain resources.	4
Upper Tactical Internet	UTI	WIN-T	4
Vehicular Amplifier Adapter	VAA	It provides additional power amplification for clear communication over greater ranges.	4
Very High Frequency	VHF	Very high frequency is the ITU designation for the range of radio frequency electromagnetic waves from 30 to 300 megahertz, with corresponding wavelengths of ten to one meter.	4
Voice over IP	VoIP	Voice over Internet Protocol (also voice over IP, VoIP or IP telephony) is a methodology and group of technologies for the delivery of voice communications and multimedia sessions over Internet Protocol (IP) networks, such as the Internet. The terms Internet telephony, broadband telephony, and broadband phone service specifically refer to the provisioning of communications services (voice, fax, SMS, voice-messaging) over the public Internet, rather than via the public switched telephone network (PSTN).	2
Vehicular Radio Communications	VRC	Army/Navy designation for terrestrial RF communications equipment.	4
Vehicle Wireless Package	VWP	The Vehicle Wireless Package (VWP) is a communications package for non-WIN-T Command and Control (C2) vehicles. The VWP B-Kit provides remote connectivity to a TCN via a Local Access Waveform for command and control vehicles during at-the-halt and on-the-move operations. It is a small form factor Local Area Network extension of the TCN's satellite and terrestrial line-of-sight network systems.	2
Vehicle Wireless Platform	VWP	Discontinued WIN-T configuration item.	4
Wide Area Network	WAN	A wide area network (WAN) is a telecommunications network or computer network that extends over a large geographical distance. Wide area networks are often established with leased telecommunication circuits.	2
Warfighter Information Network-Tactical	WIN-T	WIN-T is the Army's tactical communications network backbone that enables mission command and secure reliable voice, video and data communications anytime, anywhere. Leveraging both satellite and line-of-sight capabilities for optimum efficiency, effectiveness and operational flexibility, the WIN-T network provides the data "pipe" that other communication and mission command systems need to connect into in order to operate. With WIN-T, Commanders and Soldiers can leverage mission command applications at any location, from traditional command posts, to network-equipped vehicles crossing the battlefield, even from the belly of C17 aircraft en route to an objective.	2

TERM	ACRONYM IF EXISTENT	DEFINITION	VERSION
Work in Process	WIP	A status that means activities have started but are not yet complete. It is commonly used as a status for incidents, problems, changes etc.	2
Warfighter Initialization Tool	WIT	<p>"Now, the Army is bringing increased flexibility to the way troops initialize their mission command systems by putting the power in the hands of the communications officer, or S6, to make real-time changes, on the fly. Once data products are delivered to the unit, the S6 can make the alterations to reflect exactly what systems are on the ground, instead of sending revisions back and ordering new data products. The S6 will now be able to integrate new equipment, modify or add roles, and have those changes take effect almost immediately compared to the old way of doing business. This capability has already been delivered to select units across the Army fielded with Capability Sets 13-15. The Army will take that effort one step further this summer with the planned fielding of the next generation of initialization tools, which will enable Soldiers at individual workstations to initialize their own system, much like turning on a commercial computer for the first time.</p> <p>"</p>	4

What's new in 4.0.0

Last updated on March 1, 2021.

Last Reviewed and Approved on PENDING REVIEW

Rapid Provisioning System Release Notes

Last updated on August 11, 2021.

Last Reviewed and Approved on PENDING REVIEW

Unreleased Changes

[Unreleased]

Added

- Added PostgreSQL as RPS CMDB. This replaces Microsoft SQL Server.
- Added Task Management Service (TMS). This replaces SMA.
- Added WindowsMsp as an accepted type to patch object manifest.
- Added the ability to find Target Items, Resource Items, Resource Groups, Target Groups by filter (property) where it contains a given string instead of only exact matches
- Added filtering capabilities to the Maintenance Window and History views under Package Streams.
- New capability in Sync for node to register itself with its parent once online. i.e., NOSC registering itself with Master Node.
- Added DSC Pull Server Plugin.
- Added new certificate that is used for DSC Pull Server Client Authentication.

• **New and Updated RPS Web UI Features**

- Package Stream History tab updated to display Superseded Package Assignment telemetry
- Package Stream Approvals tab updated to remove Package deployment status
- Package Stream History tab has been redesigned to organize data by: Package Stream -> Node -> Target -> Package
- Package Stream History tab updated to include the ability to filter by Node, Package Stream, or Package
- Packages page added (RPS Menu > Distribution > Packages)
- Package Streams menu item renamed:
 - Previous: RPS Menu > Distribution > Packaging
 - Current: RPS Menu > Distribution > Package Streams
- Add new UI to create new Package Streams
- Add new UI to upload RPS Packages to existing Package Streams
- Added new telemetry to track when a Package fails to deploy to a Target and the Target is running an incompatible Patch System version
 - New telemetry for individual assignments:
 - IsAboveMaxPatchSystemVersion
 - IsBelowMinPatchSystemVersion
 - Updated roll-up telemetry logic to account for the above new telemetry. They will be counted as errors when calculating the roll-up status.
- Item UI Page will now display all Target Items with and without Parent Ids.
- Added Certificate Management page used for rolling RPS managed certificates.

• **New PowerShell Cmdlets**

- Add-RpsPatch: Accepts a collection of Patches with the `-Patch` parameter to make the child items of a Patch Stream that has not yet been approved.
- Approve-RpsPatchStream: new cmdlet for Approving a Patch Stream.
- Deny-RpsPatchStream: new cmdlet for Rejecting a Patch Stream.
- Disable-RpsMonitorUser: Disables monitoring on a user.
- Disable-RpsPatch: Disables Patch as well as patches that depend on this patch.

- Enable-RpsMonitorUser: Enables monitoring on a user.
- Enable-RpsPatch: Enables Patch as well as dependent patches.
- Export-RpsSecurityData: Used to export RPS RBAC Users and Role Assignments.
- Get-RpsAuditEntry: Gets all audit entries or ones by specified criteria.
- Get-RpsInstalledSeedDataVersion: Gets the current installed seed data version from the CMDB.
- Get-RpsRole: Gets RPS RBAC Role object.
- Get-RpsRoleAssignment: Gets the role assignment for a user, role, or both.
- Get-RpsSeedDataVersion: Gets the version of the seed data in the Rps.SeedData.dll.
- Get-RpsUser: Gets an RPS RBAC user.
- Import-RpsSecurityData: Used to import RPS RBAC Users and Role Assignments.
- Import-RpsSeedData: Imports all seed data in the Rps.SeedData.dll into the CMDB.
- Protect-RpsMasterKey: Encrypts the master key with the certificate described by the given thumbprint.
- Remove-RpsRoleAssignment: Removes the role assignment for a specified user.
- Remove-RpsUser: Removes an RPS RBAC user.
- Test-RpsPatchManifest: Validation of a Patch Manifest now allows for miscellaneous/extra elements to be added to a manifest and not checked against the XSD.

● New Web Application Plugins

- Certificate Manager: Rps.CertificateManager.RpsPlugin is a REST endpoint used to invoke the certificate request process. For additional details, see [RPS Certificate Management Technical Design](#).
- Certificate Request: Rps.CertificateRequest.RpsPlugin is a REST endpoint used to communicate with a specified Certificate Authority. Specific use case includes sending certificate signing requests to the Certificate Authority and retrieve certificates that have been issued from requests made by RPS. For additional details, see [RPS Certificate Management Technical Design](#).

Deprecated

Deprecated the following obsolete PowerShell cmdlets (will be removed in a future version):

CMDLET DEPRECATED	REPLACED BY
Get-RpsPatchManifest	Get-RpsPatchManifest
Get-RpsPatchStream	Get-RpsPatchStream
Get-RpsPatch	Get-RpsPatch
New-RpsPatchStream	New-RpsPatchStream
New-RpsPatch	New-RpsPatch
Remove-RpsPackageStream	Remove-RpsPatchStream
Remove-RpsPatch	Remove-RpsPatch

Removed

- References to Sync Service Account.
- Removed Microsoft SQL Server as RPS CMDB. It is replaced by PostgreSQL.
- Removed Master-Controller runbook.
- Removed SMA. It is replaced by Task Management Service (TMS).
- Removed the "SetDefaultProperties" method functionality from the ResourceGroup object.
- Removed the "SetDefaultProperties" method functionality from the ResourceItem object.
- Removed the following obsolete PowerShell cmdlets:

CMDLET REMOVED	REPLACED BY
ConvertTo-RpsResourceGroupXml	Export-RpsData
ConvertTo-RpsResourceItemXml	Export-RpsData
ConvertTo-RpsTargetGroupXml	Export-RpsData
ConvertTo-RpsTargetItemXml	Export-RpsData
ConvertTo-RpsTaskAssignmentXml	Export-RpsData
ConvertTo-RpsTaskItemXml	Export-RpsData
ConvertTo-RpsTaskMapXml	Export-RpsData
Find-RpsLogItem	Get-RpsLogItem
Find-RpsNode	Get-RpsNode
Find-RpsResourceAssignment	Get-RpsResourceAssignment
Find-RpsResourceGroup	Get-RpsResourceGroup
Find-RpsResourceItem	Get-RpsResourceItem
Find-RpsTargetGroup	Get-RpsTargetGroup
Find-RpsTargetItem	Get-RpsTargetItem
Find-RpsTaskAssignment	Get-RpsTaskAssignment
Find-RpsTaskAssignmentUserAction	Get-RpsTaskAssignmentUserAction
Find-RpsTaskItem	Get-RpsTaskItem
Find-RpsTaskMap	Get-RpsTaskMap
Find-RpsTaskMapDefDependency	Get-RpsTaskMapDefDependency
Find-RpsTaskMapDefFilter	Get-RpsTaskMapDefFilter
Find-RpsTaskMapDefinition	Get-RpsTaskMapDefinition
Find-RpsTaskMapStep	Get-RpsTaskMapStep
Find-RpsTaskMapStepDependency	Get-RpsTaskMapStepDependency
Find-RpsTaskMapStepFilter	Get-RpsTaskMapStepFilter
Import-RpsResourceGroup	Import-RpsData

CMDLET REMOVED	REPLACED BY
Import-RpsResourceItem	Import-RpsData
Import-RpsTargetGroup	Import-RpsData
Import-RpsTargetItem	Import-RpsData
Import-RpsTaskAssignments	Import-RpsData
Import-RpsTaskItem	Import-RpsData
Import-RpsTaskMap	Import-RpsData
Invoke-RpsEvaluateTaskAssignmentStatus	Invoke-RpsEvaluateTaskAssignment
Invoke-RpsResetTaskAssignmentStatus	Invoke-RpsResetTaskAssignment
New-RpsTaskMapDefDependency	New-RpsTaskMapStepDependency
New-RpsTaskMapDefFilter	New-RpsTaskMapStepFilter
New-RpsTaskMapDefinition	New-RpsTaskMapStep
New-RpsTaskMapStructure	New-RpsTaskMap
Remove-RpsResourceAssignmentStatus	Remove-RpsResourceAssignment
Remove-RpsTaskAssignmentStatus	Remove-RpsTaskAssignment
Remove-RpsTaskMapDefDependency	Remove-RpsTaskMapStepDependency
Remove-RpsTaskMapDefFilter	Remove-RpsTaskMapStepFilter
Remove-RpsTaskMapDefinition	Remove-RpsTaskMapStep
Update-RpsNode	Set-RpsNode
Update-RpsPackageStream	Add-RpsPatch, Approve-RpsPatchStream, Reject-RpsPackageStream
Update-RpsResourceAssignmentStatus	Update-RpsResourceAssignment
Update-RpsTaskAssignmentStatus	Update-RpsTaskAssignment
Update-RpsTaskMapDefFilter	Update-RpsTaskMapStepFilter
Update-RpsTaskMapDefinition	Update-RpsTaskMapStep

Changed

- Access to the database is now limited to a SQLAccount with **Role DatabaseAccount** and the database connection string will now use this SQL account instead of integrated authentication. (Windows Authentication)
- Allowed filtering resolved parameters based on the IsActive property.

- TaskAssignment property `SmaJobGuid` is changed to `PhyrJobId`, corresponding to the system change from SMA to TMS.
- `IsSMA` property of Node Target Items is now `IsTaskManagement`, corresponding to the system change from SMA to TMS.
- Manually importing and publishing runbooks is no longer needed. Executing runbooks is now automated when copied into the Runbook folder. The folder can be located by using `Get-RpsStorageValue -Key "DefaultRunbookFolder"` and can be changed by using `Set-RpsStorageValue -Key "DefaultRunbookFolder" -Value "[Full path of the folder you wish to use]"`
- Updated Master Key encryption to support both RSA and ECC certificates.
- **Updated PowerShell Cmdlets**
 - `Export-RpsData`: Added the ability to export data as JSON using `-Format json` parameter.
 - `New-RpsPatch`: The `PackageStream` parameter is now optional so a Package can be created without a Package Stream.
 - `New-RpsTargetGroup`: Add `Properties` parameter allowing properties to be added with one command.
 - `New-RpsTaskAssignment`: `NodeIdToRunOn` Parameter added.
 - `Remove-RpsPatch`: The `PackageStream` parameter is now optional to remove the Package from the given Package Stream.
 - `Set-RpsConnectionString`: Added properties for `DatabaseType` (defaults to Postgres), `Port number`, `Connection Timeout`, and `Command Timeout`. Added examples.
 - `Test-RpsPatchManifest`: The cmdlet will return true if a manifest is valid or return false and write out a list of validation exceptions if the manifest is invalid.

Fixed

- Fixed Get-Help documentation for Rps-API PowerShell module.
- Fixed "Cannot assign Target Items to an Item Group from the Web UI" Known Issue.
- **#115891** Fixed where the same package assigned to multiple appliances on the same node would cause the DSC resource to fail when processing packages.

Known Issues

- Patch Manifest `Conditions` element `Value` field does not support multiple values separated by the pipe delimiter `|`
 - **Error Details:** The following PackageManifest code snippet is an example using pipe delimiter `|` in `Conditions`, which will fail:

```
<InstallerFileName>opera.msi</InstallerFileName>
<Conditions>
  <PackageAssignmentCondition>
    <Property>Name</Property>
    <Operator>Eq</Operator>
    <Value>ad.unit.domain|nosc.local.rps</Value>
  </PackageAssignmentCondition>
</Conditions>
```

The resulting behavior: Only the first value listed in the `Value` field will receive an assignment; all other values after the pipe delimiter `|` are ignored.

```
PS C:\Windows\system32> Get-RpsResourceAssignment -ResourceItemType 'Package' -TargetItem (Get-RpsTargetItem -Name 'ad.unit.domain' -Type 'virtualmachine') | ft ResourceItem
ResourceItem
-----
Package - Opera/70.0.0
Package - Firefox/70.0.0
Package - AdobeReaderDC/19.12.20034
Package - x64-Windows8_1-KB4486105-x64/4486105.0.0
Package - windows8.1-kb4519990-x64/2019.10.8

PS C:\Windows\system32> Get-RpsResourceAssignment -ResourceItemType 'Package' -TargetItem (Get-RpsTargetItem -Name 'nosc.rps.local' -Type 'virtualmachine') | ft ResourceItem
ResourceItem
-----
Package - Firefox/70.0.0
Package - x64-Windows8_1-KB4486105-x64/4486105.0.0
Package - windows8.1-kb4519990-x64/2019.10.8
Package - AdobeReaderDC/19.12.20034
```

In this particular example, `ad.unit.domain` is assigned the `opera` patch, because it was listed before the pipe delimiter `|`.

nosc.local.rps is not assigned the *opera* patch, because it was listed after the pipe delimiter | .

- **Current Workaround for pipe delimiter |** : Utilize the Match Operator `<Operator>Match</Operator>`, with each value in the `Value` field wrapped in parentheses `()` and with a trailing question mark `?` . Example:

```
<Conditions>
  <PackageAssignmentCondition>
    <Property>ComputerName</Property>
    <Operator>Match</Operator>
    <Value>(NFA)?(WNM)?(WNMA)?</Value>
  </PackageAssignmentCondition>
</Conditions>
```

In the above example, a target with a `Property` of *ComputerName* will be assigned if its `Value` contains *NFA*, *WNM*, **and/or** *WNMA*. This implementation only requires a **partial** value match.

For an **exact** value match, the full string in the `Value` field must be enclosed with a caret `^` and a dollar sign `$` .
Example:

```
<Conditions>
  <PackageAssignmentCondition>
    <Property>ComputerName</Property>
    <Operator>Match</Operator>
    <Value>^(NFA)?(WNM)?(WNMA)?$</Value>
  </PackageAssignmentCondition>
</Conditions>
```

In the above example, a target with a `Property` of *ComputerName* will be assigned if its `Value` contains *NFA*, *WNM*, **and** *WNMA*.

- Sync service and Patching
 - The switch to PostgreSQL breaks the Sync service and Patching (which is dependent on Sync) until the Sync service can be refactored to work with PostgreSQL.
 - The refactoring is scheduled for Alpha 3.
 - **Current Workaround:** None. The sync feature to resolve this issue is scheduled for Alpha 3.
- Certificate Management
 - From the RPS Certificate Manager plugin, there is an issue with decrypting Resource Item protected properties in order to encrypt private key with password. This will cause the automated certificate request process to be unsuccessful. This issue will be addressed in Alpha 4.
- RBAC: Possible permissions bleed-over between users
 - This bug has a small chance of occurring when multiple users are interacting with RPS, either through PowerShell or the Web GUI, which could result in a high volume of concurrent requests.
 - This bug was discovered through testing, by spamming the RPS API with multiple concurrent requests from different users. It is unlikely to occur in a real-world environment.
 - This bug is only applicable to authenticated users. Because RBAC only handles authorization, it is not possible for this bug to occur for non-authenticated users.

Released on February 28, 2020

What's new in 3.1.0 (Feb 28)

- New PowerShell CMDLETS
 - New-RpsGroupCondition: Creates a Condition object to add to a Resource Group or Target Group to dynamically add Members.
 - Get-RpsGroupFilter: Gets the filter object on a Resource Group or Target Group
 - Remove-RpsGroupCondition: Removes a condition from a Resource Group or Target Group
 - Get-RpsResolvedParameter: Resolves a Parameter against a Target Item.
 - New-RpsPatchStream: Creates new Package Stream and Packages
 - New-RpsPatch: Creates a new Package object and adds it to a Package Stream
 - Update-RpsPackageStream: Updates an existing Package Stream
 - Remove-RpsPatch: Removes an existing Package from a Package Stream and optionally removes all assignments
 - Remove-RpsPatchStream: Removes an existing Package Stream, Packages, and optionally removes all assignments
 - Get-RpsPatch: Gets an existing Package
 - Get-RpsPatchStream: Gets an existing Package Stream
 - Get-RpsMaintenanceWindow: Gets an existing Maintenance Window
 - New-RpsMaintenanceWindow: Creates a new Maintenance Window
 - Remove-RpsMaintenanceWindow: Removes an existing Maintenance Window
 - Set-RpsMaintenanceWindow: Sets a Maintenance Window if it exists or creates a new Maintenance Window if it doesn't exist
 - Enable-RpsCdn: Turns On/Off Bits and/Or Dfsr communication.
- Updated PowerShell CMDLETS
 - New-RpsResourceGroup: Add parameters Operator and Condition
 - New-RpsTargetGroup: Add parameters Operator and Condition
- Updated DSC Utilities for resolving PowerShell Parameters to take a TargetItem and the ParameterMetadata object. You now do not need to have a resource assignment to a DSC partial or the parameter imported into the CMDB. You would use this in an external PowerShell script to resolve properties from the CMDB.
- Updated PowerShell Functions
 - Resolve-RpsNode: **(Breaking change)** The function now requires user to be in an active session context or connected to a SQL database. It no longer uses hard-coded strings to resolve the names of nodes.
 - Resolve-RpsNode: Can optionally return the Node object versus just the string name
 - Resolve-RpsNode: Added ability to filter out default node
- Updated RPS Installer
 - Added ability to deploy a preconfigured RHEL virtual machine with the RHELTemplateFilename switch parameter
 - Group ContentCreators that has permission to the CDN folder for adding content (e.g. Packages)
- Sync Changes
 - Item Properties have a SyncScope to define how the property should be synchronized
 - Public - synchronize
 - Private - do not synchronize
 - Internal - synchronize only to internal nodes
 - InternalDownstream - synchronize only to internal children
 - InternalUpstream - synchronize only to internal parents
- CDN Changes
 - New properties on Node to define the protocol: ParentCdnProtocol and ChildCdnProtocol. The valid values are 'Bits' or 'Dfsr' (case and whitespace sensitive) which tell nodes the protocol to use for each Content Delivery Network.
 - Set-RpsResourceType: Add parameters:

- CDNDirection (Upstream or Downstream) - This can be used by content (e.g. Packages) to specify the direction it should be synchronized
 - IsContentDistribution - This indicates whether a resource refers to content that will be synchronized
- New Resource Item call CdnSettings that gets globally created with two Internal properties, IsBitsEnabled and IsDfsrEnabled. The two properties only sync within internal nodes. i.e. within a unit.
- New RPS Type Definitions
 - Package: Defines a Package and its properties
 - Package Stream: Defines a Package Stream and its properties
- New DSC Resources
 - RPS Package Manager: DSC Resource for RPS Package Streams and Packages. Provides tools to Get, Set, and Test Packages for a Target
- New PowerShell Modules
 - RPS Package Provider: Provides methods to Find, Install, or Uninstall packages; Get list of installed packages; Add or Remove Packages Sources
- New RPS Web UI Features
 - Packaging section added (RPS Menu > Distribution > Packaging)
 - Approvals tab: Approve or Reject Package Streams
 - Scheduling tab: Create, Edit, and Delete Maintenance Windows
 - History tab: View the deployment status of Package Streams, Packages, and Assignments
- Added new function Test-DscModuleConflict to use for testing for required DSC Module conflicts across all assigned partials for a single Target Item.
- Added Get-AdminRoleCredential to Rps-API-Utilities; used to determine which credential role to use in a task

Known issues in 3.1.0

- Web UI
 - Closing the Remove Resource modal resets filters
 - From the Resources screen, if you have a filter added for Packages and you confirm and remove a package it will clear your filters
 - Workaround: Manually re-apply your filters
 - Cannot assign Target Items to an Item Group from the Web UI
 - From the RPS Web UI menu: Targeting > Item Groups
 - Open the patchable target group by clicking on the name
 - Open the Members accordion
 - When you try to click "+ Add New Item" you'll receive an error
 - Workaround: PowerShell can be used to add Target Items to an Item Group:

```
$ti = Get-RpsTargetItem -Id IdOfMyTargetItem
$tg = Get-RpsTargetGroup -Id IdOfMyGroup
$tg.AddChildItem($ti)
$tg.Update()
```

- glyphicons halflings are not signed and do not display in Web UI when deployed due to STIG
 - Some icons (e.g. status icons for Packages) do not display in Internet Explorer when deployed due to STIG
 - Workaround: The status can be easily discerned by font color and wording of the statuses.

- API
 - Dates are parsed out of Properties as strings and we always lose timezone specifications
 - This could cause some times to be slightly incorrect depending on how they were stored and/or retrieved
 - Workaround: No known workaround.
 - Currently there is no support for Maintenance Windows that span across 2 or more days
 - Workaround: Create 2 maintenance windows - one for each day so it covers the entire period of time desired.
- PowerShell cmdlets
 - DateTime for Start/End date in Get-RpsMaintenanceWindow displays inaccurate time
 - Get-RpsMaintenanceWindow returns the Start/End date which includes a time, however, the time is inaccurate and it conflicts with the start/end times that are also returned to the user
 - Workaround: No known workaround. This is a visual/display issue. The time portion included alongside the Start/End dates that are shown are not what is being used. The actual times being used are displayed separately.

Released on September 20, 2019

What's new in 3.0.3 (Sep 20)

The primary update in this Hotfix release is to address:

- **#23726** Fix: Provisioning Node App Server fails to configure DSC. Login failed for user.

Released on September 19, 2019

What's new in 3.0.2 (Sep 19)

The following are work items completed in support of the 3Q19 Release and delivered in Hotfix 3.0.2. These fixes include Critical and High Risk Factor mitigations.

- **#23542** Fix: Address .NET 2.0 Vulnerabilities
- **#23543** Fix: Address .NET 3.0 Vulnerabilities
- **#23556** Fix: Address CVE-2017-8529 mitigation for Internet Explorer vulnerability
- **#23555** Fix: Address 'Memory Management\FeatureSettingsOverride' mitigations
- **#23547** Fix: Address Visual C++ 2008 SP1 Vulnerability
- **#23548** Fix: Address GPO setting "Hardened UNC Paths" (KB3000483)
- **#23544** Fix: Address .NET 3.5 Vulnerabilities
- **#23545** Fix: Address SQL Server 2012 SP4 Vulnerability (KB4057116)
- **#23550** Fix: Address SSL Version 2 and 3 Protocol Detection

⚠ IMPORTANT

The ContentStore has been updated to include various "binary" patches, such as for SQL Server, and Microsoft .NET. Additionally, the 3Q19 **.vhdx / .iso** have been updated to include additional Windows patches. Please ensure the latest ContentStore and the 3Q19-2012R2-0919 from the Release are used.

What's new in 3.0.1 (Sep 05)

- **#23286** Fix: Task Management Service will throw errors after running for a long time

What's new in 3.0.0 (Aug 23)

- **#19833** Timestamp Logic for BITS

- **#23135** Certificates issued by RPSRoot do not have FQDN in the SAN
- **#22992** Test DCA deployment without a PFX certificate
- **#23147** Resource Group membership cannot be updated by subsequent node imports
- **#23230** RpsDomainJoin account doesnt get the correct permissions to add a computer to the domain that is prestaged
- **#23229** RpsProvisioning dns record is set to Interanl nic instead of 996 and 59 NIC
- **#23227** Web Config files are being overwritten by RPSGUI, RPSProvisioning, and TrustedElementRepository DSC Partial
- **#23093** xDFSR uses Domain Admin account
- **#23049** RVP configured with specific registry settings for compliance
- **#21370** Master-Controller fails to resume after service or machine restart
- **#23210** SQL SA account name conflict
- **#23026** Content Delivery Network Partial is missing a mandatory parameter

What's new in 3.0.0-beta (Aug 16)

- **#22496** Update PSScriptAnalyzer to 1.18.1
- **#22311** When installing the content store to a directory other than c:/contentstore certificates are installed in the wrong path
- **#22432** Remove RVPS GUI files and install and powerstig from the release
- **#23035** RpsDomainJoin accounts are set to Create = False within RpsAccounts.csv file
- **#22647** Configure a new Packaging Repository and migrate our code out of Core.
- **#22684** Updates needed to the Ports and Protocols section of the RPS Install Guide.pdf
- **#21495** RPS currently does not have a way to continually re-publish DSC partials
- **#22537** Update OSS registrations and Third Party Notices file
- **#22913** Files located under the folder c:\ContentStore\Export are not encrypted (on the APP VM)
- **#22948** Failing resource on RVP - [xPackage]ACCM
- **#19973** Deploy PowerSTIG 3.3.0
- **#22541** Add a script resource RpsDomainController.ps1 to execute certutil.exe - installdefaulttemplates
- **#22243** Automate Axway Desktop Validator Enterprise
- **#22483** Need to update SkipRules for DSC PowerStig configuration
- **#22433** Access Database partial is assigned in colorless baseline data for RVP
- **#22683** RPS Install does not work per published installation directions
- **#22694** Export-NodeData Runbook variable \$TargetItem is not correctly referenced
- **#21970** Unable to export taskmap definitions with Export-RpsData
- **#22015** Update RPS logging during deployment to better characterize issues
- **#21971** Lot 7 NOSC NIPR RVP ActivClientAppInstall patch fails (RPS 2.4.5)
- **#22607** Failing resource - [xPackage]ActivClient71
- **#21560** DSC Partial should only require OSCore when necessary
- **#22549** Update DCA Assignments.psd1
- **#22557** Update source Certificate locations and update the Certificates.psd1 and CertificationAuthority partial.
- **#20605** TrustElementRootPath gets set to wrong path
- **#21458** APP and AD VMs do not have PowerStig configurations
- **#22246** Any website on the c:\ drive is a CAT II finding
- **#22248** RPSAdmin domain account password should be user configurable for APP and AD
- **#22417** Newly generated self-signed certificates sometimes not loaded into the CMDB
- **#22426** Import-NodeData fails on APP VM when importing node data
- **#22429** Duplicate Import-RPSNode Functions
- **#22436** OSCP website has request filters that need to be removed
- **#22449** RVP - CdnPath points to C\$ instead of share
- **#22458** Registry resource failing to add registry keys for TER authorization
- **#22593** Update to only install McAfee agent 5.5
- **#22594** RVP - CdnPath points to C\$ instead of share
- **#22598** Registry resource failing to add registry keys for TER authorization

- **#22670** DomainJoinAdmin gets access denied when joining RVP to the domain
- **#22677** Remove Install-MNRps.ps1 as it is no longer used.
- **#22682** RpsProvisioning folder path creation should not use the FQDN for folder name
- **#22691** Generated certificates are missing FQDN for subject name
- **#22743** RpsGUI not reaching desired state due to a certificate error
- **#22936** Existing WinRm settings on a target cause the set-winrm runbook to fail.
- **#23001** Missing SSL binding reg key in trust element repository partial
- **#23015** Provisioning node configuration has missing master key encryption role on several accounts
- **#23034** Copy-Baselmages references the wrong local account for credentials.
- **#22454** Failing resource on RVP - [AdcsOnlineResponder]OnlineResponder
- **#22464** OSCore New Computername Timing
- **#22666** Deploy the RPS 3.x codebase in Hyper-V
- **#22725** Install-Rps.ps1 does not update MN node target items' VhdTemplateFileName property
- **#22542** Replace \$DomainAdmin with \$CAServiceAccount in the CertificationAuthority.ps1 partial.
- **#22555** Inhibit and restart Tumbleweed service to the DesktopValidatorStandardAppInstall.ps1
- **#22587** Add the Certificate partial dependency to the CertificationAuthority partial.
- **#22945** Create a partial for the Exit Module
- **#23073** Update the CA Partial to use RPSadmin
- **#23074** DomainJoin resource fails on DCA
- **#23075** RVP needs to be a member of TPKI Writes AD Group

What's New in 3.0.0-beta (Aug 9)

- **#22432** Remove RVPS GUI files and install and powerstig from the release
- **#22311** When installing the content store to a directory other than c:/contentstore certificates are installed in the wrong path
- **#22684** Updates needed to the Ports and Protocols section of the RPS Install Guide.pdf
- **#22483** Need to update SkipRules for DSC PowerStig configuration
- **#22433** Access Database partial is assigned in colorless baseline data for RVP
- **#22694** Export-NodeData Runbook variable \$TargetItem is not correctly referenced
- **#22549** Update DCA Assignments.psd1
- **#21458** APP and AD VMs do not have PowerStig configurations
- **#22670** DomainJoinAdmin gets access denied when joining RVP to the domain
- **#22429** Duplicate Import-RPSNode Functions
- **#22691** Generated certificates are missing FQDN for subject name
- **#22426** Import-NodeData fails on APP VM when importing node data
- **#22458** Registry resource failing to add registry keys for TER authorization
- **#22598** Registry resource failing to add registry keys for TER authorization
- **#22677** Remove Install-MNRps.ps1 as it is no longer used.
- **#22682** RpsProvisioning folder path creation should not use the FQDN for folder name
- **#22449** RVP - CdnPath points to C\$ instead of share
- **#22594** RVP - CdnPath points to C\$ instead of share
- **#20605** TrustElementRootPath gets set to wrong path
- **#22593** Update to only install McAfee agent 5.5
- **#22725** Install-Rps.ps1 does not update MN node target items' VhdTemplateFileName property
- **#22496** Update PSScriptAnalyzer to 1.18.1
- **#22647** Configure a new Packaging Repository and migrate our code out of Core.
- **#22537** Update OSS registrations and Third Party Notices file
- **#22541** Add a script resource RpsDomainController.ps1 to execute certutil.exe - installdefaulttemplates
- **#22243** Automate Axway Desktop Validator Enterprise
- **#22015** Update RPS logging during deployment to better characterize issues
- **#21560** DSC Partial should only require OSCore when necessary

- **#22555** Add inhibit and restart Tumbleweed service to the DesktopValidatorStandardAppInstall.ps1
- **#22587** Add the Certificate partial dependency to the CertificationAuthority partial.
- **#20407** Configure BITS/DFSR per node type

Files excluded from the drop!

In order to improve the speed with which RPS artifacts can be integrated with other code repositories, the decision was made to exclude files from Core which required modification later in the integration process for Mission Network. These files and folders are below:

- DSC\Modules\NetworkingDSC\6.1.0.0\DSCResources\MSFT_HostsFile\MSFT_HostsFile.psm1
- DSC\Modules\MN_OfficeDSC\
- DSC\Modules\MN_SchemaExtensionDSC\
- DSC\Modules\MN_xWinEventLog\
- DSC\Modules\WINT_NetworkPolicyServer\
- Modules\MN-AnalyzerRules\
- Modules\MN-Automation\
- Modules\MN-ISO\
- Modules\MN-Rps-Api\
- Modules\MN-Ssh\
- Modules\MN-VMWare-Utilities\
- DSC\PartialConfigurations\ActivClientAppInstall.ps1
- DSC\PartialConfigurations\AdobeReaderAppInstall.ps1
- DSC\PartialConfigurations\AdSchemaExtension.ps1
- DSC\PartialConfigurations\ClientPki.ps1
- DSC\PartialConfigurations\DesktopValidatorStandardAppInstall.ps1
- DSC\PartialConfigurations\DoDInstallRootAppInstall.ps1
- DSC\PartialConfigurations\FirefoxAppInstall.ps1
- DSC\PartialConfigurations\Firewall.ps1
- DSC\PartialConfigurations\GpoWmiFilter.ps1
- DSC\PartialConfigurations\GroupPolicy.ps1
- DSC\PartialConfigurations\McAfeeHBSSAppInstall.ps1
- DSC\PartialConfigurations\MsftAppLocker.ps1
- DSC\PartialConfigurations\MsftDnsServer.ps1
- DSC\PartialConfigurations\NetBannerAppInstall.ps1
- DSC\PartialConfigurations\OcspsResponder.ps1
- DSC\PartialConfigurations\Office2013AppInstall.ps1
- DSC\PartialConfigurations\OpenSSLAppInstall.ps1
- DSC\PartialConfigurations\OracleJDKAppInstall.ps1
- DSC\PartialConfigurations\OracleJREAppInstall.ps1
- DSC\PartialConfigurations\PuTTYAppInstall.ps1
- DSC\PartialConfigurations\RvpsGUI.ps1
- DSC\PartialConfigurations\SmartCardManager90MeterAppInstall.ps1
- DSC\PartialConfigurations\SoftphoneAppInstall.ps1
- DSC\PartialConfigurations\SolarWindsETAppInstall.ps1
- DSC\PartialConfigurations\TeraTermAppInstall.ps1
- DSC\PartialConfigurations\TigerVNCAppInstall.ps1
- DSC\PartialConfigurations\TrustElementRepository.ps1
- DSC\PartialConfigurations\VMWareClientIntegrationPlugInAppInstall.ps1
- DSC\PartialConfigurations\VMWareRemoteConsoleAppInstall.ps1
- DSC\PartialConfigurations\VMWareToolsAppInstall.ps1
- DSC\PartialConfigurations\VMWarevSphereClientAppInstall.ps1

- DSC\PartialConfigurations\VMWarevSpherevCLIAppInstall.ps1
- DSC\PartialConfigurations\WaveDesktopCommunicatorAppInstall.ps1
- Images\ESX\Grangeville.cfg
- iPXE Distro\
- Provisioning\Provisioning Vlan Address Space.csv
- Runbooks\Copy-BasImages.ps1
- Runbooks\Get-TargetDhcpIPAddress.ps1
- Runbooks\Import-VMWareVirtualAppliance.ps1
- Runbooks\New-VMWareVirtualMachine.ps1
- Runbooks\Remove-VMWareVirtualMachine.ps1
- Setup\
- Utilities\

What's New in 3.0

PowerShell

This release contains the following PowerShell enhancements:

- Added Import-RpsInstanceDefinition and Export-RpsInstanceDefinition to Import/Export InstanceDefinitions as Json.
- Added Import-RpsDataMapping and Export-RpsDataMapping to Import/Export Data Mappings as Json.
- Added Import-RpsResourceItemJson and Export-RpsResourceItemJson to Import/Export Resource Items as Json.
- Added Set-RpsDataImportMapping to the API from Rps-DataMapping module.
- Added Set-RpsDataFilter to the API from Rps-DataMapping module.
- Added Set-RpsDataCondition to the API from Rps-DataMapping module.
- Added Set-RpsDataProperty to the API from Rps-DataMapping module.
- Added Set-RpsDataAssociation to the API from Rps-DataMapping module.
- Added Set-RpsMappingFilter to the API from Rps-DataMapping module.
- Added Set-RpsDataVariable to the API from Rps-DataMapping module.
- Added Set-RpsDataMapping to the API from Rps-DataMapping module.
- Added Set-RpsDataFile to the API from Rps-DataMapping module.

Rps-Installer

This release contains the following Rps-Installer enhancements:

DSC

This release contains the following DSC enhancements:

- MofStore location is now located within C:\ContentPath\DSC.
 - OutputPath parameter is no longer set on the node, or set statically.
 - Publish-DSCConfiguration now creates, and sets the OutputPath parameter for all assigned partials.
 - Removed Mandatory flag from OutputPath parameter on all partials.
 - Updated runbooks to pass OutputPath within calls to LCM functions.
 - Default value for LCM functions within the RPS-DSC module is now the present working directory for stand alone use.

RPS API

This release contains the following API enhancements:

- Adding the following Type Property constants:
 - IsContentDistribution
 - IsSoftwareDistribution
 - IsColumnDisplay

- Updated the Set-RpsResourceType cmdlet to indicate if the Resource Type is for software or content distribution. The IsContentDistribution and IsSoftwareDistribution switches are mutually exclusive in the cmdlet, however, setting the IsSoftwareDistribution switch will also set the IsContentDistribution flag on the Resource Type.
- Updated the Set-RpsTypeProperty cmdlet to indicate if the Property Type can be used for Column Displays within the Admin UI.
- Updated the New-RpsResourceGroup cmdlet to allow for properties to be provided at creation
- Updated the Write-RpsLogItem cmdlet to write to the appropriate PS stream. Tokenization in MessageTemplate is also more forgiving.
- Added Json support for InstanceDefintitions and Data Mappings to help make creation and updating easier.
- Added Json support for Resource Items to have a readable and organized way of import/exporting resource items and sharing between nodes.
- When TaskMaps are included in an export, their TaskMapSteps are also exported by default
- Added an optional Certificate parameter for passing a certificate file (.cer) that will encrypt the resulting configuration file in the Exit-RpsSession and Export-RpsData cmdlets.
- Added optional Certificate and password parameters for passing certificate file (.pfx) and password that will be used to decrypt a configuration file in the Enter-RpsSession and Import-RpsData cmdlets.
- Added support to encrypt/decrypt export files during the install process.

RPS Sync Service

This release contains the following Sync Service enhancements:

RPS CDN

This release contains the following RPS Content Delivery Network (CDN) enhancements:

- DFS-R is used for communication between Region and Site nodes.
- BITS is still used for communication between Master and Region nodes.
- Due to DFS-R mesh networking, all files are replicated to all Region and Site nodes within a domain, regardless of assignment.
- Patches will still only be installed on assigned targets

Admin UI

This release contains the following Admin UI enhancements:

- Replaced 'Patching' on the top menu bar with 'Distribution'
- Added dynamically created sections under distribution for Content Distribution and Software Distribution

Resolved Issues

Top issues addressed in 3.0:

- Fixed issue where the SMA runbook service account was denied access to the MofStore after STIGs were applied
- Fixed issue where there was a credential conflict with the Windowsfeature Net-Framework-Core resource, between the RpsSMA, and RpsSQL partial.

Known Issues

SQL Server 2012

This release contains the following SQL enhancements:

- Microsoft SQL Server 2012 has been upgraded from Service Pack 2 (SP2) to SP4

PowerShell

This release contains the following PowerShell enhancements:

- Added support to allow certificate store parameter to be passed from CMDB.
- Added support for additional certificate roles:
 - CAp7b
 - CertificationAuthorityPFX
 - CACertificateChain
 - Added Force switch for Set-RpsResourceItem, Set-RpsTargetItem, New-RpsResourceItem, New-RpsTargetItem, New-RpsResourceGroup, and New-RpsTargetGroup
- Removed module RPS-Credentials and functions:
 - Get-Credential
 - New-Credential
 - Get-ServiceAccount
 - Added data files for Users and Certificates. They can be found here 'Setup\Configuration\Data\RpsAccounts.csv' and here 'Setup\Configuration\Data\RpsCertificates.csv'. For both data files, if no password is provided in the password column, a password will be randomly generated per user/certificate.

Rps-Credential

- New-RpsCredential was updated to allow generation of a password, with or without a provided password policy.

Rps-Installer

- Import-RpsCredential was updated to allow generation of a password, with or without a provided password policy.
- MofStore location was changed from C:\Windows\Temp to C:\ContentStore\DSC

DSC

This release contains the following DSC enhancements:

- Added Certification Authority Partial to install and configure a Certification Authority node.
- Domain Admins no longer joins machines to the domain.
- The RpsDomainJoin account now joins staged computer objects, within the Computers OU, to the domain using the minimum permissions required.
- ContentDeliveryNetwork partial updated to install DFS-R for Region and Site nodes
- Updated Dsc Modules to the following versions:

MODULE	CORE VERSION
AccessControlDsc	1.3.0.0
ComputerManagementDsc	6.2.0.0
PowerStig	3.1.0
xWebAdministration	2.5.0.0

MODULE	CORE VERSION
ResourceControllerDSC	2.0.1

RPS API

This release contains the following API enhancements:

- Updated the Set-RpsTargetItem and Update-RpsTargetItem cmdlets to not allow the altering of an existing Parent if it has already been set.
- Adding the following ResourceType constants:
 - CATemplate
 - OcspUriPath
 - CdpUriPath
 - AiaUriPath
 - RegistryItem
 - CrI
 - NPSPolicyMap
 - NPSCClient
 - RegistryAccessEntry
 - RegistryAccessControlList
 - RegistryAccessRule
 - CertificationAuthority
- BREAKING CHANGE: TypeDefinitions are now enforced on Target Items. All required params will have to be set when creating the target item.
- Added Instance Definitions, which are pre-defined complex, default data for the purpose of quickly defining data but also codifying configuration data
- Added a User Profile context to the Rps API in order to provide and track the current user. This provides the foundation for a RBAC implementation.
- Added an optional method of export to provide the user with a plaintext XML file where protected properties are in the clear.
- Added an optional CertificateThumbprint parameter to the Enter-RpsSession, Exit-RpsSession, Export-RpsData, and Import-RpsData cmdlets in order to encrypt/decrypt file exports and imports.
- Added Get-RpsPasswordPolicy cmdlet
- Added Set-RpsPasswordPolicy cmdlet
- Added New-RpsPassword cmdlet
- Added Update-MasterKey cmdlet
- Added Get-RpsProtectedProperty cmdlet.
- Added Instance Definition Nodes, which are pre-defined objects that can be used to create Nodes that are associated with Instance Definitions
- Added Set-RpsTargetType cmdlet.
- Added Set-RpsResourceType cmdlet.
- Added Set-RpsSubType cmdlet.
- Added Set-RpsChildType cmdlet.
- Added Set-RpsTypeProperty cmdlet.
- Added Set-RpsTypeRA cmdlet.
- Added Set-RpsTargetAction cmdlet.
- Added Set-RpsResourceGroupType cmdlet.
- Added Get-RpsCredential cmdlet.
- Added New-RpsCredential cmdlet.
- Added Get-UnixHash cmdlet

- Added Set-RpsTargetGroupType cmdlet.

Sample:

```
$secureResult = Get-RpsProtectedProperty -TargetItem $targetItem -Name $name
$secureResult = Get-RpsProtectedProperty -ResourceItem $resourceItem -Name $name
$secureResult = Get-RpsProtectedProperty -Node $node -Name $name
$secureResult = Get-RpsProtectedProperty -TaskMapAssignment $taskMapAssignment -Name $name
$secureResult = Get-RpsProtectedProperty -ResourceGroup $resourceGroup -Name $name
$secureResult = Get-RpsProtectedProperty -TargetGroup $targetGroup -Name $name

# parameter setup
$name = "property name"

# return variable to plain text
$plainText = ConvertFrom-SecureString $secureString
```

- Added Set-RpsProtectedProperty cmdlet.

Sample:

```
Set-RpsProtectedProperty -TargetItem $targetItem -Name $name -Value $securePwd
Set-RpsProtectedProperty -ResourceItem $resourceItem -Name $name -Value $securePwd
Set-RpsProtectedProperty -Node $node -Name $name -Value $securePwd
Set-RpsProtectedProperty -TaskMapAssignment $taskMapAssignment -Name $name -Value $securePwd
Set-RpsProtectedProperty -ResourceGroup $resourceGroup -Name $name -Value $securePwd
Set-RpsProtectedProperty -TargetGroup $targetGroup -Name $name -Value $securePwd

# parameter setup
$name = "property name"
$value = ConvertTo-SecureString "Value" -AsPlainText -Force
```

- Enhanced Set-RpsTargetItem and Set-RpsResourceItem cmdlets to accept protected properties from a hashtable using the SecureString type. Sample:

```
Set-RpsTargetItem -Name $name -Type $type -Properties @{ Protected = $secureString }
Set-RpsResourceItem -Name $name -Type $type -Properties @{ Protected = $secureString }
...
```

- Breaking Change: Marked New-RPSTaskMapStructure cmdlet as Obsolete.
- Modified the API to mask protected properties when they are returned to the console.
- Added New-RpsInstanceDefinition Cmdlet. Sample:

```
$hs = @{
  Prop1 = "value1"
  Prop2 = "value2"
  New-RpsInstanceDefinition -Name testName -Properties $hs
```

- Added New-RpsInstanceDefinitionItem Cmdlet. Sample: An Instance Definition Item is a wrapper for an RPS type and associated Properties.

```
PowerShell New-RpsInstanceDefinitionItem -EntityName testEntityName -Name name2 -Properties @{Prop1 = "Value1"} -TypeDefinitionId $typedefinition.id
```

- Added Invoke-RpsInstanceDefinition Cmdlet.

```
PowerShell Invoke-RpsInstanceDefItem -Settings $resourceItem -InstanceDef $instanceDefinition
```

- Added Set-RpsInstanceDefinition Cmdlet. Sample:


```
Set-RpsInstanceDefinition -Name $Name1
Set-RpsInstanceDefinition -Name $Name1 -Properties @{Prop1 = "Value1"}
```

Added Remove-RpsInstanceDefinitionItem Cmdlet.

```
Remove-RpsInstanceDefinitionItem -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"
Remove-RpsInstanceDefinitionItem -InstanceDefinitionItem $InstanceDefinitionItem
```

- Added Get-RpsInstanceDefinition Cmdlet. Sample:

```
Get-RpsInstanceDefinition -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"
Get-RpsInstanceDefinition -Name MyInstanceDef
```

- Added Remove-RpsInstanceDefinition Cmdlet. Sample:

```
Remove-RpsInstanceDefinition -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"
Remove-RpsInstanceDefinition -InstanceDefinition $InstanceDefinition
```

- Added Set-RpsInstanceDefinitionItem Cmdlet. Sample:

```
Set-RpsInstanceDefinitionItem -Name $Name1 -TypeDefinitionId $id -EntityName $entityName
Set-RpsInstanceDefinitionItem -Name $Name1 -TypeDefinitionId $id -Properties @{Prop1 = "Value1"} -
EntityName $entityName
```

- Added Get-RpsInstanceDefinitionReference. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"
$reference = Get-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -
InstanceDefinitionItem $instanceDefItem
```

- Added New-RpsInstanceDefinitionReference. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"
$taskMapIDs = "5b8b0340-091f-4823-b2f9-de937b5b4114", "a83b5445-3cc0-433e-b5e0-0fcf70389988"
$reference = New-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -
InstanceDefinitionItem $instanceDefItem -TaskMapIDs $taskMapIDs
```

- Added Remove-RpsInstanceDefinitionReference. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"
Remove-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -
InstanceDefinitionItem $instanceDefItem
```

- Added Remove-RpsInstanceDefinitionAssociation. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"
Remove-RpsInstanceDefinitionAssociation -InstanceDefinition $instanceDef -PrimaryReference
$instanceDefItem -SecondaryReference $instanceDefItem2
```

- Added New-RpsInstanceDefinitionAssociation. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"  
New-RpsInstanceDefinitionAssociation-InstanceDefinition $instanceDef -PrimaryReference  
$instanceDefItem -Secondaryreference $instanceDefItem2
```

- Added Get-RpsInstanceDefinitionAssociation. Sample:

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"  
Get-RpsInstanceDefinitionAssociation-InstanceDefinition $instanceDef -PrimaryReference  
$instanceDefItem -Secondaryreference $instanceDefItem2
```

- Added New-InstanceDefinitionNode. Sample:

```
New-RpsInstanceDefinitionNode -EntityName testEntityName -Name name2 -Hostname hostname -IPAddress  
1.1.1.1 -SyncEndpointUrl syncEndpoint -certificateThumbprint certThumbprint -pollingInterval 1
```

- Added Set-InstanceDefinitionNode. Sample:

```
PowerShell Set-RpsInstanceDefinitionNode -Name name1 -EntityName testEntityName2 -Hostname hostname2 -  
IPAddress 2.2.2.2 -SyncEndpointUrl syncEndpoint2 -certificateThumbprint certThumbprint2 -pollingInterval 2
```

- Added Get-InstanceDefinitionNode. Sample:

```
PowerShell $instanceDefNode = Get-RpsInstanceDefinitionNode -Name name1
```

- Added Remove-InstanceDefinitionNode. Sample:

```
PowerShell Remove-RpsInstanceDefinitionNode -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423" Remove-  
RpsInstanceDefinitionNode -InstanceDefDefinitionNode $InstanceDefinitionNode
```

- Updated demo data scaffolding to support DCA image testing.
 - Added NOP79190 node to Nodes.psd1 to support DCA image testing.
 - Added NOP79190 target item to TargetItems folder to support DCA testing.
 - Updated Assignments.psd1, ResourceGroups.psd1, and Initialize-Image.ps1 to support DCA image testing.
- Updated demo data scaffolding to support NDM image testing.
 - Updated TCN79192 demo data scaffolding to support NDM image testing.
 - Added TCN79192 node to Nodes.psd1 to support NDM testing.
 - Added TCN79192 target item to TargetItems folder to support NDM testing.
 - Updated Assignments.psd1, ResourceGroups.psd1, and Initialize-Image.ps1 for NDM image testing.
- Modified Initialize-Baseline.ps1 to support dynamic testing of images.
- Updated PartialConfigurations-cmdb.tests.ps1 to support dynamic testing of images. Now includes the DCA and NDM image.

Admin UI

- Added Generate Random Password functionality for Resource Item, Target Item, and Patch Password fields.
- Added the ability for Generate Random Password to be based on a Password Policy.
- Modified the UI to mask protected properties.
- Added password/protected property reveal functionality to the UI.
- Added the ability to supply a certificate thumbprint for encrypting/decrypting CMDB file export/import via the UI.
- Updated the Task Map Step Number and Depends On columns, so it has a consistent sort order.

Resolved Issues

Top issues addressed in 3.0:

- Fixed issue in the the RPS Install that was causing DSC to fail on Node Registration and import.
- Fixed issue with SID translation that would force manual intervention.
- Fixed issue where user rights assignment settings within the core repo were conflicting when STIGs were applied.

Known Issues

Issues Addressed in RPS release 2.4.6

- **#21479** LCM Configuration Mode is not controllable per target from the CMDB
- **#21445** Missing utilities folder in local content store
- **#21422** RpsProvisioning cannot configure virtual drive
- **#21557** Rps-encryption breaking SAN's
- **#21558** Rps-Network doesnt allow existing exclusion assignments
- **#21559** RpsDomainController only applies tombstone lifetime to primary dc
- **#20900** 2Q19 User Principal Name suffix isn't being configured on DSC VM or AD.rps.local
- **#21469** Access Database partial is assigned in colorless baseline data for RVP
- **#21483** Remove RVPS GUI files and install and powerstig from the release
- **#21538** Prov Vlan updates from GDMS
- **#21539** TaskMap Updates from GDMS
- **#21566** Runbook retries implemented where network communications can be a factor
- **#21536** GPO Updates to UnifiedAD
- **#20788** Duplicate Import-RPSNode Functions
- **#21472** OSCP website has request filters that need to be removed
- **#20670** PowerSTIG Service rules fail if the expected service does not exist

Issues Addressed in RPS release 2.4.5

- **#21275** Cannot add patches to a target and republish
- **#21396** RVP computer account on Dev node deployment not added to correct OU
- **#21371** OsCore does not create a Disk resource for targets with multiple disks
- **#20952** 2Q19 Update to only install McAfee agent 5.5
- **#21269** Publish-DSCPatch.ps1 pathing bug breaks Patching
- **#20951** 2Q19 Update VMware-tools-10.3.10-12406962
- **#16291** RVP - CdnPath points to C\$ instead of share
- **#20897** RVP missing ImagesParentPath property
- **#20867** New-ProvisioningNodeConfiguration.ps1 has incorrect property name images parent folder property
- **#20875** Unneeded array item causes a duplicate resource ID error during compilation
- **#20674** Failing resource on DSC - [xADForestProperties]
- **#20878** RpsSQL and RpsSMA resource controller has incorrect import version
- **#17860** On a running Prov Laptop APP VM, when a taskmap assignment and RunOnLocalNode has been issued, the first 3 workflows fail
- **#21354** Get-DSCStatus is not assigned to targets therefore patch status is not updated
- **#20897** RVP missing ImagesParentPath property
- **#20928** 2Q19 Utilities and Certificates not present in localContent Store Path
- **#20931** Created Dev Enclave/Node to reduce the complexity of deployments across teams
- **#20932** Remove hardcoded data from installer. Modify Hyper-V VM creation scripts to ensure VM environments are generated as specified
- **#20948** 2Q19 Update Adobe Reader to 19.012.20034
- **#20950** 2Q19 Update Java\jre-8u212-windows-x64.exe
- **#21395** Ocsponder partial is skipped on RVP due to missing property

- **#21383** RPSOSCore.ps1 network profile configuration can only set one interface to Private, otherwise there are resource conflicts
- **#21006** 2Q19 Certificate generation creates malformed SANs during import
- **#20928** 2Q19 Utilities and Certificates not present in localContent Store Path

Issues Addressed in RPS release 2.4.4

- **#19832** Copy-ContentStore does not log an alert when a file copy fails
- **#19666** ResourceGroups.ps1 for the DSC missing items
- **#20757** Adobe Acrobat version needs updated in partial
- **#20559** Firewall Rules only allow traffic for specific applications
- **#20603** GPO SIDs are not being translated into domain accounts when imported
- **#20607** TrustElementRepository Reader/Writer sites have incorrect bindings
- **#20690** Failing resource on RVP - [xPackage]McAfee Agent
- **#19528** Test-DscMof does not detect resource conflicts between PowerStigConfiguration and other partials
- **#20605** TrustElementRootPath gets set to wrong path
- **#20611** Failing resource on RVP - [AdcsOnlineResponder]OnlineResponder
- **#20776** RVP data has assigned partials that should not be assigned
- **#19832** Copy-ContentStore does not log an alert when a file copy fails
- **#19661** Set contentfreshness for sysvol replication on all domain controllers to 365 days maxtimeofflineindays setting
- **#20604** Conflicting ComputerManagementDsc module versions
- **#20821** SMA Runbook account needs to have LogonAsAService permissions
- **#20607** TrustElementRepository Reader/Writer sites have incorrect bindings
- **#19528** Test-DscMof does not detect resource conflicts between PowerStigConfiguration and other partials
- **#20605** TrustElementRootPath gets set to wrong path
- Update to PowerSTIG 3.2.0
- STIG Rule Updates: 41023, 41024, 4102, 41026, 41407, 41021, 41022, 41027 41028, 41029, 41030, 41031, 41032, 41033, 41035, 41042 41305, 41306, 41307, 41037, V-41251, V-40950, V-69169 V-40952, V-40953, V-41016, V-41017

Issues Addressed in RPS release 2.4.3

Released on May 15, 2019

- Removed dependency on xCAPIstore resources
- Removed unneeded service restart in domain controller resource
- Updated Install-MNRps so execution can occur without \$VhdFolderPath and \$VMTemplateFileName
- Separated reader and writer SSL settings for the Trusted Element Reader website
- Fixed duplicate resources being created between PowerStig and RPS partials
- The \$allComputers variable in Rps-Installer module was not properly populated and resulted in unexpected deployment
- Fixed issues where DNS Zones were not loaded; the same fix addresses 'Domain Controller promotion fails due to unknown root cause; possible SMB contention issue' and 'replica DC promo fails multiple connection issue'
- Added an array for value data on IPv6 disable resource
- Fixed Master-Controller failing to resume after service or machine restart
- Addressed xWebAdministration version references mismatch
- Added a reboot for ssl binding registry update to address Registry resource failing to add registry keys for TER
- Fixed inbound Reader and Writer traffic being blocked to the RVP
- Fixed the issue where PFX files were attempting to get uploaded to the TER site
- Removal of ResourceControllerDSC module version 1.3.1 and added ResourceControllerDSC module version 2.0.0.
- Updated version for import-module calls for ResourceControllerDSC
- Added use of ResourceController for Allow Log on Locally, and Log on as a Service URAs within RpsSync partial.
- Within the RpsSecLIS partial, added three resources that leverage ResourceControllerDSC to remove .NET v.4.5, and .NET v.4.5 Classic accounts from Log on as a service, Generate security audits, and replace a process level token before they

become rogue/dead sids.

- COTS Update - McAfee
 - Agent 5.5.1.462
 - ACCM 3.2.5
 - RSD 5.0.6.125
 - SIEM Collector **new**
- COTS Update - ActivClient 7.1
- COTS Update - Adobe Reader 19.10.20091.53467

Issues Addressed in Core release 2.4.2

Released on April 17, 2019

- Added and updated tombstone parameter
- Added forest name to domain object for laptop build
- User and Group property updates
- Add MC check to ensure only one MC is running
- Add max reserved memory for SQL
- Added RSAT for DNS
- Add tombstone configuration

Issues Addressed in DSC_Images tagged 2.4.2

- Changes made to partial to reflect the most current VMWare tools software
- Integrate PowerStig 3.1
- Bug fix to allow for duplicate name
- Add a forest name property to the ADDomain object
- Add valid task map action
- Updating the NT Auth store can fail - this breaks CAC login
- Updated NPS partial to use NPS group configuration from CMDDB
- Update tombstone value on domain controller
- Added property for max memory

Issues Addressed in 2.4.1

Released on April 2, 2019

- partial update for the gpomanagementdsc module update in DSC_Images
- adding UPN Suffix to adobjects
- Update RpsDomainController.ps1
- disable ipv6
- Using Registry instead of xRegistry

What's New in 2.4

Released on January 29, 2019

PowerShell

This release contains the following PowerShell enhancements:

- The RPS Installer was updated to support complex task map execution in order to provide the ability to create ESXi, VMware, or Hyper-V based hosts and virtual machines.
- Added support for ESXi Host and virtual machine configurations.
- Improved Installer's ability to generate representative XML for RPS Import by reducing the number of switches required during the installation/configuration.
- Reorganized RPS PowerShell Modules into:

MODULE	DESCRIPTION
Rps-API	Core API functions
Rps-Credential	Create and access credentials in RPS CMDB
Rps-Dsc	Utility to help publish, manage and test RPS DSC Partials
Rps-Encryption	Manage certificates and encryption
Rps-Installer	RPS Configuration, Data Import and Installation helpers
Rps-IpSheet	Import networking information from an IPSheet Excel document
Rps-Network	Network Utilities
Rps-Snmp	Communicate with network switches
Rps-Types	Create and manage RPS Type Definitions
Rps-Utilities	Additional Utilities
Rps-Virtualization	Management of Virtualization

- Refactored New-HypervVirtualMachine to support additional configuration options. The new runbook is now called Set-HyperVVirtualMachine. Enhancements include support for the following:
 - Generation 1 virtual machines
 - Vhd disks
 - All virtual switch types (Internal, External, Private)
 - N number of disk/dvd drives and nics (Up to Hyper-V limitations)
 - Processor configuration
 - Static/Dynamic memory configuration
 - Image from .iso, differencing disk, existing disk
 - Virtual network adapter IP address configuration, including VLAN tagging

In order to take advantage of all these configurable options, the data must be representative of the configuration that is desired. Below is a representation of the relationship within the Rps type definitions:

OBJECT	RPS ENTITY TYPE	RPS TYPE	RPS SUBTYPE	PARENT OBJECT	ASSIGNMENT
Host	Resource/Target	Host	HyperV	N/A	VirtualMachine
Virtual Machine	Target	VirtualMachine	N/A	N/A	Host

OBJECT	RPS ENTITY TYPE	RPS TYPE	RPS SUBTYPE	PARENT OBJECT	ASSIGNMENT
Virtual NIC	Target	NIC	VirtualMachine	N/A	VirtualSwitch
VHD(X)	Target	Drive	Disk	VirtualMachine	N/A
Dvd	Target	Drive	DVD	VirtualMachine	N/A
Processor	Target	Processor	N/A	VirtualMachine	N/A
Virtual Switch	Resource	VirtualSwitch	HyperV	N/A	NIC

To see the configurable properties on each of these objects, please reference the Rps type definitions located at "ContentStore\Setup\Configuration\Import-RpsTypes.ps1".

Sample configurations are located at "ContentStore\Demos\Set-HyperVVirtualMachine".

DSC

This release contains the following DSC enhancements:

- Added support for multi-step software installs to the Software Distribution Partial.
- Updated Runbook Guidance based on lessons learned from ESXi and SNE MVP.
- Added support for additional DHCP configuration options in the RpsDhcp partial such as:
 - Scope option definitions
 - Scope definitions
 - Exclusion ranges
 - Server bindings
- Updated Dsc Modules to the following versions:

MODULE	CORE VERSION
ComputerManagementDsc	6.0.0.0
ResourceControllerDSC	1.3.1.0
SqlServerDsc	12.1.0.0
xActiveDirectory	2.22.0.0
xHyper-V	3.13.0.0
xWebAdministration	2.3.0.0

- Added support for PKI functionality to support DCA image with new DSC resource MN_ActiveDirectoryCSDsc (Forked from ActiveDirectoryCSDsc 3.1.0.0). New resource include:
 - AdcsAiaExtension
 - AdcsCdpExtension
 - AdcsCertificateTemplate

- AdcsImportCrl
 - AdcsInstallCertificate
 - AdcsOcspExtension
 - AdcsPublishCert
 - AdcsPublishCrl
- Added support for GPO Management functionality to support NDM image with DSC resource MN_GpoManagementDsc. New resource include:
 - GpSecurityFilter

RPS API

This release contains the following API enhancements:

- Added support for structured logging during unattended RPS Installer executions.
- Updated the RPS API to optimize Target loading with several Task Map Assignments.
- Resource Items and Resource Assignments can now be retrieved by Role, which is a special property designated for tracking the purpose of a resource item or its relationship to a target item. The Role property can be placed on a Resource Item or the Resource Assignment and can hold multiple values separated by the `|` symbol. To get resource items that have a specific role or have an assignment with a specific role, use the `-MatchAssignmentRole` parameter.

Sample:

```
$clientAuthCerts = Get-RpsResourceItem -Type Certificate -Role "ClientAuth"
$localAdmins = Get-RpsResourceItem -TargetItem $computer -Type Credential -Role "LocalAdministrator"
-MatchAssignmentRole
```

Sample:

In this example, a Credential (Resource) is assigned to a Computer (Target). The assignment is given a Role of "LocalAdministrator". We can retrieve the designated Local Administrator credential for the computer by using the `-Role` parameter.

```
# assign credential and set roles
$computer = Get-RpsTargetItem -Type "Computer" -Name "Win137"
$credential = Get-RpsResourceItem -Type "Credential" -Name "RpsAdministrator"
$assignedCredential = New-RpsResourceAssignment -TargetItem $computer -ResourceItem $credential
$assignedCredential.Role = "LocalAdministrator|RpsUser"
Update-RpsResourceAssignment -ResourceAssignment $assignedCredential

# retrieve the LocalAdmin credential for the computer
$localAdminAssignment = Get-RpsResourceAssignment -TargetItem $computer -Role "LocalAdministrator"
```

- Added the `-Scope` parameter on the `New-RpsTaskStep` cmdlet.

Sample:


```

New-RpsTaskItem -WorkflowName "Resolve-TargetMacAddress"
New-RpsTaskItem -WorkflowName "Wait-TargetReady"
New-RpsTaskItem -WorkflowName "Wait-TargetReady"
New-RpsTaskItem -WorkflowName "Copy-BaseImages"
New-RpsTaskItem -WorkflowName "New-VMWareVirtualMachine"
New-RpsTaskItem -WorkflowName "Resolve-TargetDhcpIPAddress"

# parameter setup
$baremetalConfig = @{ TargetItemType = "Computer"; Filters = @{ IsHypervisor = "False" } }
$esxConfig = @{ TargetItemType = "Computer"; Filters = @{ IsHypervisor = "True" } }
$vmConfig = @{ TargetItemType = "VirtualMachine"; Filters = @{ "IsAppliance" = "False" } }
$vmApplianceConfig = @{ TargetItemType = "VirtualMachine"; Filters = @{ "IsAppliance" = "True" } }
$rvpConfig = @{
    TargetItemType = "VirtualMachine";
    Filters = @{ "IsAppliance" = "False"; "Designation" = "RVP" }
}

# Task Map creation
$map = New-RpsTaskMap -Type "ProvisionSystemDemo" -Name "ProvisionSystemDemo"
$mapConfig = @{ TaskMap = $map; AllowMultipleTargets = $true; IsTargetRequired = $true }

# Adding steps to task map
$resolveMac = New-RpsTaskMapStep @mapConfig -RunbookName "Resolve-TargetMacAddress" -TargetItemType
"Switch"
$waitBaremetal = New-RpsTaskMapStep @mapConfig -RunbookName "Wait-TargetReady" -TargetItemType
"Computer"

$baremetalHV1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Copy-BaseImages" -Dependencies
$waitBaremetal @esxConfig
$baremetalVM1 = New-RpsTaskMapStep @mapConfig -RunbookName "New-VMWareVirtualMachine" -Dependencies
$baremetalHV1 @vmConfig

#Adding step with dependency
$baremetalVM2 = New-RpsTaskMapStep @mapConfig -RunbookName "Resolve-TargetDhcpIPAddress" @vmConfig -
Dependencies $baremetalVM1 -Scope Self

```

Admin UI

This release contains the following Admin UI enhancements:

- Added LocalNode UI option on execution of Task Map Assignment.
- Defaulted Resource Assignment state to Ready where no approval action needs to take place.
- Calculated the file hash of an imported file on Import.
- Corrected Pending Task Individual Count.
- Removed the Pending Actions on targeting list views.
- Made the Active and Global flags display consistently throughout the UI.
- Replaced the Edit and Remove hyperlinks with command buttons on the TaskMap derail views.
- Updated the UI to optimize loading a Target Item's details when several Task Map Assignments exist.
- Combined Pending Tasks and Task Information sections into one Job section on the Target Details view.

Resolved Issues

Top issues addressed in 2.4:

- Resource Groups fails to import when group references already exist.
- Provisioning Service returns a 500 error if a duplicate object is found.
- Calling Set-RpsResourceItem and/or Set-RpsTargetItem with null properties causes a null-reference exception.
- Set-RpsResourceItem does not update parent's state when adding children.
- Import TaskMap with non-default dependency when scope is ignored.
- IpSheet import fails due to missing Access Database Engine pre-requisite.
- Task Assignment History not saved while in Session.

- RPS Session failing to refresh deleted Task Assignments from Target Item.
- Pending Task Individual Count is incorrect in the Target Item Detail view.
- Modified the Wait-TargetReady runbook to support both PhysicalMachine and Computer Types.
- The Installer's -GenerateXmlOnly switch fails to generate usable file when -ConfigFilename specified.
- Added the ability for more than one process to access isolated storage at the same time.
- Exported data doesn't include the Task Map Assignment if assigned to Child Item.
- Task Map Step Dependency scope is not imported.
- Added a fix for Installer when Script fails to fully execute when not running elevated.
- Inception deployment fails with an HttpSetServiceConfiguration error.
- Import-Rpslpsheet on Testlpsheet takes too long.
- Encrypted Dsc partials fail to decrypt when a partial without a credential is applied first.

Known Issues

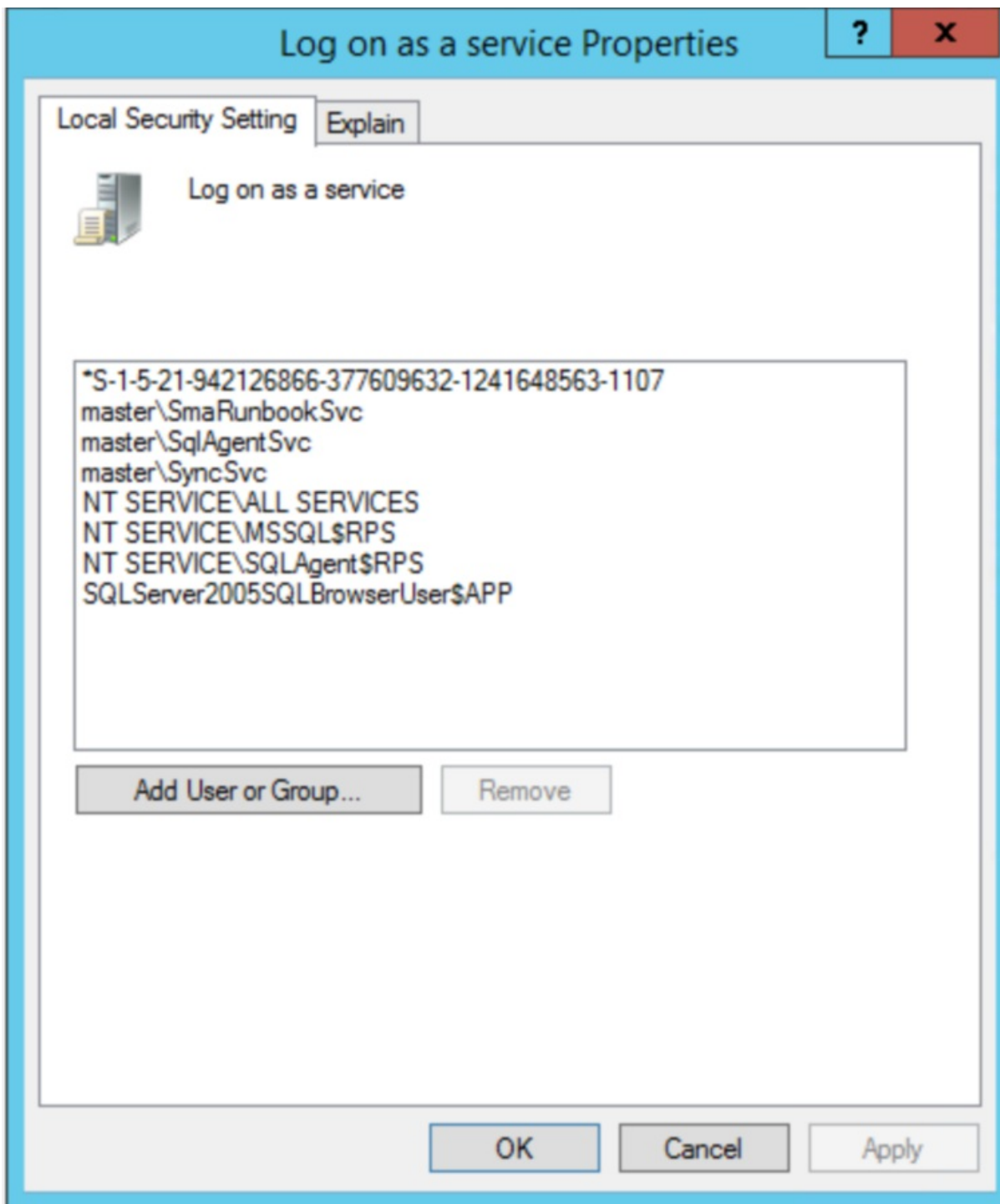
- RPS Install isn't exporting Host Node info, causing DSC to fail on Node Registration and import.
- UserRightsAssignment Dsc resource can sometimes fail due to failure to translate SIDs. See [here](#) for more information on the details. You can see this error exposed in Dsc:

```

VERBOSE: [APP]: LCM: [ Start Test ] [[UserRightsAssignment]RpsSyncLogOnAsService]
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Adjust_memory_quotas_for_a_process
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Adjust_memory_quotas_for_a_process
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-21-942126866-377609632-1241648563-1107 on Policy: Log_on_as_a_service
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Replace_a_process_level_token
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Replace_a_process_level_token
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Generate_security_audits
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Generate_security_audits
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Testing SyncSvc is present on
policy Log_on_as_a_service

```

A workaround for this is to open Secpol.msc and remove any untranslated SID's for the targeted user right:



What's New in 2.3

Released on October 30, 2018

PowerShell

This release contains the following PowerShell enhancements:

- Master-Controller now has the ability to run recurring tasks and scheduled tasks.
- The Get-DscStatus runbook will now by default run every two hours.
- Virtual disk file locations will use the Hyper-V default filepath when creating a virtual machine. You can also optionally specify an alternate location to store the vhd.

- ServerAdmin role created for all administrative functions required by Rps. It previously required the DomainAdmin role.
- Installer can dynamically generate self-signed certificates per deployment. It will use the configuration data supplied to populate their properties. Can also supply your own certificates. See the *Certificate Usage* document for details.
- Added the capability to suppress reboots for individual software installs.
- Reorganized RPS PowerShell Modules into:

MODULE	DESCRIPTION
Rps-Api	Core API functions
Rps-Credential	Create and access credentials in RPS CMDB
Rps-Dsc	Utility to help publish, manage and test RPS DSC Partials
Rps-Encryption	Manage certificates and encryption
Rps-Installer	RPS Configuration, Data Import and Installation helpers
Rps-IpSheet	Import networking information from an IPSheet Excel document
Rps-Snmp	Communicate with network switches
Rps-Types	Create and manage RPS Type Definitions
Rps-Utilities	Additional Utilities

DSC

This release contains the following DSC enhancements:

- Created/Updated DSC Partial Configurations to support cross-forest configurations, Provisioning Service, CDN Service
- Updated Dsc Modules to the following versions:

MODULE	CORE VERSION
AccessControlDsc	1.1.0.0
CertificateDsc	4.4.0.0
ComputerManagementDsc	5.2.0.0
NetworkingDsc	6.1.0.0
SecurityPolicyDsc	2.4.0.0
SqlServerDsc	11.4.0.0
xActiveDirectory	2.21.0.0
xDatabase	1.9.0.0
xDhcpServer	2.0.0.0

MODULE	CORE VERSION
xDnsServer	1.11.0.0
xPSDesiredStateConfiguration	8.3.0.0
xSmbShare	2.1.0.0
xWebAdministration	2.2.0.0
xWindowsUpdate	2.7.0.0

RPS API

This release contains the following API enhancements:

- The Task assignment restrictions were relaxed so that a Task Map can be assigned to non-root Target items. The New-RpsTaskAssignment Cmdlet previously restricted an assignment to only root-level target items. However, the restriction is no longer applicable within vehicle provisioning scenarios, where the vehicle is the root, DCEs are child items and virtual machines are grandchildren targets.
- Updated the Task Map Dependency scope to allow defining dependencies scoped to "all" (target), "self", and "parent". This provides the capability to have DCE1 tasks run parallel to DCE2 tasks, given that the DCEs are both children of a SNE parent.
- Simplified Task Map creation by creating the New-RpsTaskMapStep Cmdlet. The Cmdlet may accept a runbook name parameter instead of "TaskItem", essentially eliminating the need for using the existing Task Map structure. In addition, the New-RpsTaskMapStep Cmdlet will accept filters and dependencies inline.
- The New-TaskMapDefinition, New-TaskMapDefFilter, and New-TaskMapDefDependency Cmdlets are marked as obsolete and have been replaced by the New-RpsTaskMapStep, New-RpsTaskMapStepFilter, and New-RpsTaskMapStepDependency Cmdlets respectively.

Sample:

```

# Create target items
$sne1 = New-RpsTargetItem -Type Vehicle -Name SNE1
$switch1 = New-RpsTargetItem -Type Switch -Name "Cisco Switch" -ParentItem $sne1
$dce1 = New-RpsTargetItem -Type DCE -Name "DCE 1" -ParentItem $sne1
$dce2 = New-RpsTargetItem -Type DCE -Name "DCE 2" -ParentItem $sne1
$dce3 = New-RpsTargetItem -Type DCE -Name "DCE 3" -ParentItem $sne1
$rvpVM = New-RpsTargetItem -Type VM -Name "RVP" -ParentItem $dce2

# Create tasks
$task1 = New-RpsTaskItem -WorkflowName "Wait-Switch"
$task2 = New-RpsTaskItem -WorkflowName "Wait-DCE"
$task3 = New-RpsTaskItem -WorkflowName "Set-DCEConfig"
$task4 = New-RpsTaskItem -WorkflowName "New-ESXIVM"
$task5 = New-RpsTaskItem -WorkflowName "Publish-Dsc"

# Create task map
$map = New-RpsTaskMap -Type "Provision-Vehicle" -Name "Provision-SNE"
$step1 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task1 -TargetItemType Switch
$step2 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task2 -TargetItemType DCE -Dependencies $step1
$step3 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task3 -TargetItemType DCE
New-RpsTaskMapStepDependency -PreviousStep $step2 -Step $step3 -Scope Self
$step4 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task4 -TargetItemType VM
New-RpsTaskMapStepDependency -PreviousStep $step3 -Step $step4 -Scope Parent
$step5 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task5 -TargetItemType VM
New-RpsTaskMapStepDependency -PreviousStep $step4 -Step $step5 -Scope Self

# Assign map
New-RpsTaskAssignment -TaskMap $map -TargetItem $sne1

```

Sample: Inline filters and dependencies

```

$byFilter = New-RpsTaskMapStep -TaskMap $map -Filters @{ Type = "VirtualMachine"; IsDsc = $true }
$withDependencies = New-RpsTaskMapStep -TaskMap $map -Dependencies @( $step1, $step2 )
$byRunbookName = New-RpsTaskMapStep -TaskMap $map -RunbookName "Publish-Dsc"

```

- Added the ability to nest Resource Groups in order to enable RPS to model many complex scenarios such as AD Security groups.

Sample:

```

# Define a new Type Definition with the IsGroupReference flag
Set-RpsResourceType -Name "ADGroup" -IsRoot -IsGroupReference

# Create AD Groups
$ADDomainUsersGroup = New-RpsResourceGroup -Type "ADGroup" -Name "All Domain Users"
$ADAdminGroup = New-RpsResourceGroup -Type "ADGroup" -Name "Domain Admins"
$ADDNSAdminGroup = New-RpsResourceGroup -Type "ADGroup" -Name "DNS Admins"
$ADDirectorsGroup = New-RpsResourceGroup -Type "ADGroup" -Name "Directors"

# Create AD Users and assign them to groups
$AdUser1 = New-RpsResourceItem -Type "AdUser" -Name "AdUser1" -ResourceGroup $ADDomainUsersGroup -
IsGlobal $true
$AdUser2 = New-RpsResourceItem -Type "AdUser" -Name "AdUser2" -ResourceGroup $ADAdminGroup -IsGlobal
$true
$AdUser3 = New-RpsResourceItem -Type "AdUser" -Name "AdUser3" -ResourceGroup $ADAdminGroup -IsGlobal
$true
$AdUser4 = New-RpsResourceItem -Type "AdUser" -Name "AdUser4" -ResourceGroup $ADDNSAdminGroup -IsGlobal
$true
$AdUser5 = New-RpsResourceItem -Type "AdUser" -Name "AdUser5" -ResourceGroup $ADDirectorsGroup -IsGlobal
$true
$AdUser6 = New-RpsResourceItem -Type "AdUser" -Name "AdUser6" -ResourceGroup $ADDirectorsGroup -IsGlobal
$true

# Add AdUser3 to the AD Directors Group as well
$ADDirectorsGroup.AddChildItem($AdUser3)
Update-RpsResourceGroup -ResourceGroup $ADDirectorsGroup

# Get Group references
$ADAdminGroupRef = Get-RpsResourceItem -Id $ADAdminGroup.Id
$ADDNSAdminGroupRef = Get-RpsResourceItem -Id $ADDNSAdminGroup.Id
$ADDirectorsGroupRef = Get-RpsResourceItem -Id $ADDirectorsGroup.Id

# Assign Group references to All Domain Users Group
$ADDomainUsersGroup.AddChildItem($ADAdminGroupRef)
$ADDomainUsersGroup.AddChildItem($ADDNSAdminGroupRef)
$ADDomainUsersGroup.AddChildItem($ADDirectorsGroupRef)
$ADDomainUsersGroup.Update()

```

- Find-Rps* Cmdlets have been deprecated and renamed to Get-Rps* with the same functionality. The original Find cmdlets have been retained and marked obsolete. However, they will be removed in a future release.
- Added new Set-RpsTargetItem and Set-RpsResourceItem Cmdlets. Both Target and Resource Items can be created and edited via their respective Set-RpsTargetItem and Set-RpsResourceItem Cmdlets.

Sample: Create a new target item via Set-RpsTargetItem

```
$computer = Set-RpsTargetItem -Type "Computer" -Name "Win137" -ParentItem $serverRack
```

Sample: Update an existing target item via Set-RpsTargetItem

```
$computer = Set-RpsTargetItem -Type "Computer" -Name "Win137" -IsActive $false
```

Sample: Create a new resource item via Set-RpsResourceItem

```
$resourceItem = Set-RpsResourceItem -Type "type" -Name "name"
```

Sample: Update an existing resource item via Set-RpsResourceItem

```
$resourceItem = Set-RpsResourceItem -Type "type" -Name "name" -IsActive $false
```

- Updated the Target Type Definitions to include a child type for Actions. Actions link a Target of a certain type to a TaskMap.

This allows a user to easily determine the status of an Action via the Admin UI.

- Added support for the retrieval of Target items, Target groups, Resource items, and Resource groups via wildcard property filters. The Get-RpsTargetItem, Get-RpsTargetGroup, Get-RpsResourceItem, and Get-RpsResourceGroup Cmdlets will return target\resource items and target\resource groups respectively using the properties supplied. If no properties are supplied, all items\groups will be returned. When using the -Filter Parameter, a \$null value may be passed as a wildcard.

Sample: Get target items by properties

```
$foundItem = Get-RpsTargetItem -Filter @{"MAC" = "00:11:22:33:44:55"}  
$foundAllItemsWithMACProperty = Get-RpsTargetItem -Filter @{"MAC" = $null}
```

- Modified the API to allow for duplicate Task Map assignments to be created. RPS prevented assigning a Task Map to the same Target item more than once. This restriction was a legacy component in order to prevent Task Maps from changing after they were assigned. However, many scenarios such as the patching and provisioning processes are required to be run multiple times. Allowing Task Maps to be run multiple times enables RPS to have a cleaner user interface, cleaner logic, and overall better response times.
- Added a new Get-RpsConstants Cmdlet that will return all the defined RPS constants.

Sample:

```
$rps = Get-RpsConstants
```

- Added Get-RpsInstanceDefinitionItem Cmdlet. Sample:

```
Get-RpsInstanceDefinitionItem -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Get-RpsInstanceDefinitionItem -Name MyInstanceDefItem  
Get-RpsInstanceDefinitionItem -ResourceItem $resourceItem -Filter $filterHashtable
```

Admin UI

This release contains the following Admin UI enhancements:

- Added the TaskMap dependency scope to the TaskMap detail page.
- Added the ability to navigate between nested Resource Groups within the user interface.
- Changed the user interface's default landing page to the local Node's detail page.
- Added a section to the Node's detail page to display the status of its child Target items.
- Added the display of associated Actions to the Target Item detail page. This allows a user to easily determine the status of an assigned TaskMap, such as "SNE Provisioning", and start the TaskMap if necessary via the Admin UI.
- The Folder detail page was modified to list the files contained within the CDN folder.
- Added bread crumbs to the user interface to simplify site navigation.
- Simplified the Target details page by modifying the view to present just the high-level processes that are running and to use drill-downs view to access the more detailed information.
- Added a new detail page for Task Map Assignment.
- Added a new detail page for Resource Assignment.
- Modified the Target Group page to allow for adding and removing Target items to and from a group.
- Added the ability to select from any Task Map or Task item when assigning a new task to a Target item.

RPS Sync Service

This release contains the following Sync Service enhancements:

- Separated process of requesting changes and sending changes, so a child node will not block operations on a parent node.
- Queue received changes on all nodes, so changes won't be re-transferred on merge errors.
- Added Snapshot Isolation to transactions to avoid inconsistent data when gathering changes.
- Audit fields have been added to CMDB objects for use by Sync processes. These will be used by API in 2.4.

RPS CDN

This release contains the following RPS Content Delivery Network (CDN) enhancements:

- CDN now uses Background Intelligent Transfer Service (BITS) to transfer files from Parent to Child CDN.
- CDN uses hierarchical topology, where child requests files from parent, instead of full mesh used by DFS-R.
- CDN includes a new Indexer Service which stores File and Folder information in the CMDB to reduce duplicate transfers.

Baremetal Provisioning Service

The RPS Provisioning Service is an HTTP-based Web API hosted in IIS for use in brokering information from the RPS CMDB to a pre-execution environment such as iPXE for installation of a defined image and configuration. For instance, iPXE can be configured to "point to" the Provisioning Service which will return a boot script file for the MAC address requested.

This release contains the new Baremetal Provisioning Service with the following features:

- Return iPxe boot scripts from an http/https service based on matching devices in the CMDB.
- Host full images (such as .iso, .wim) in the service for download from iPxe.
- Host ESXi Kickstart scripts for ESXi configuration support.
- Support approval of base image through Resource Assignments in CMDB.
- Avoid boot looping through a configurable iPxe expiration period.

Resolved Issues

Top issues addressed in 2.3:

- RPS objects were not consistently setting dates\times to UTC dates\times.
- The DependsOn attribute was not handling all options.
- In RpsSession, the TaskAssignment Cmdlets attempted to transact with the Database.
- Internet Explorer failed to display glyph icons when using custom Cache-Control.
- The RpsGui Partial throws an error when applying SSL Certificate.
- In Server 2012, the New-HyperVVirtualMachine Runbook fails to add NICs to new VirtualMachines.
- Remove-ItemProperty fails in RpsSession.
- Test-DSCConfiguration returns false on CMDB deployment even after dacpac is deployed.
- Property Bag missing support for deleting a property.
- Target Item to GetResourceGroups and Resource to GetTargetGroups returns duplicate groups if group contains multiple members.
- Duplicate node error upon entering previously saved RpsSession.
- Calling Install-Rps for a specific node fails to create a parent node.
- Sync property replication failure.
- Sql Encryption fails to apply correctly.

Known Issues

- UserRightsAssignment Dsc resource can sometimes fail due to failure to translate SIDs. See [here](#) for more information on the details. You can see this error exposed in Dsc:

```
VERBOSE: [APP]: LCM: [ Start Test ] [[UserRightsAssignment]RpsSyncLogOnAsService]
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Adjust_memory_quotas_for_a_process
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Adjust_memory_quotas_for_a_process
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-21-942126866-377609632-1241648563-1107 on Policy: Log_on_as_a_service
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Replace_a_process_level_token
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Replace_a_process_level_token
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-271721585-897601226-2024613209-625570482-296978595 on Policy: Generate_security_audits
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Could not translate SID:
S-1-5-82-3876422241-1344743610-1729199087-774402673-2621913236 on Policy: Generate_security_audits
VERBOSE: [APP]: [[UserRightsAssignment]RpsSyncLogOnAsService] Testing SyncSvc is present on
policy Log_on_as_a_service
```

A workaround for this is to open Secpol.msc and remove any untranslated SID's for the targeted user right:

Log on as a service Properties



Local Security Setting

Explain



Log on as a service

```
*S-1-5-21-942126866-377609632-1241648563-1107  
master\SmaRunbookSvc  
master\SqlAgentSvc  
master\SyncSvc  
NT SERVICE\ALL SERVICES  
NT SERVICE\MSSQL$RPS  
NT SERVICE\SQLAgent$RPS  
SQLServer2005SQLBrowserUser$APP
```

Add User or Group...

Remove

OK

Cancel

Apply

Using the RPS API

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

NOTE

Find-Rps* Cmdlets have been deprecated and renamed to Get-Rps* with the same functionality, Find verb will be removed in a feature release.

Target Items

Create Target Items

TIP

Target Items can also be created via Set-RpsTargetItem.

Create a new target item

```
$computer = New-RpsTargetItem -Type "Computer" -Name "Win137"
```

Create a new target item child

Specify the parent item when creating a child item. You can nest target items as deep as necessary.

```
$nic = New-RpsTargetItem -Type "NetworkInterface" -Name "137-NIC1" -ParentItem $computer
```

Add a child code to an existing Node

```
Add-RpsChildNode -ChildNode $childNode
```

Set-RpsNode and update its parameters

Update a node by object

```
Set-RpsNode -Node $node
```

Update a node by parameters

```
Set-RpsNode -Name name -HostName hostName -IpAddress 10.0.0.1
```

Create a new target item with Properties

Use a PowerShell Hashtable to pass properties when creating a new target item.

```
$platform = @{  
  Architecture = "x64"  
  OSVersion = "8.1"  
  OSType = "Windows"  
}  
$computer = New-RpsTargetItem -Type "Computer" -Name "Win137" -Properties $platform
```

Create a target item for a different Node

By default, RPS will create new target items for the current node, but you can supply an alternative node.

```
$site2 = Get-RpsNode -Name "Site 2"  
New-RpsTargetItem -Type "Computer" -Name "Win137" -Node $site2
```

Get Target Items

Use the `Get-RpsTargetItem` cmdlet to retrieve target items from RPS. Retrieve a single item using the `-Id` parameter, or retrieve multiple items using any combination of parameters.

Get target items by type

```
$allRouters = Get-RpsTargetItem -Type "Router"
```

Get target items by property value

Target Items can be filtered by a specific property or properties.

```
$deviceByMac = Get-RpsTargetItem -Filter @{ "MacAddress" = "00:11:22:33:44:55" }
```

Target Items can be filtered by a specific property that contains a given value.

```
$deviceByMac = Get-RpsTargetItem -Filter @{ "Contains-MacAddress" = "33:44" }
```

Get target items by property

Target Items can be retrieved which have any value for a specific property. To retrieve items with a property, use `$null` for the value in the `-Filter` parameter.

```
$allDevicesWithMac = Get-RpsTargetItem -Filter @{ "MacAddress" = $null }
```

Get target items by Node

Target items can be retrieved for a specific node.



TIP

The Node parameter is currently limited to just selecting **root** target items within the specified node. A future update may expand the search to **all** items within the Node, including child items.

```
$site2Computers = Get-RpsTargetItem -Node $site2 -Type "Computer"
```

Update Target Items

Update target items using the `Set-RpsTargetItem` cmdlet. The `Set-Rps*` cmdlets will also create items if they don't exist.

The standard way to update an item uses the item's logical identifier, which is Type and Name. To update an existing target item, supply the `-Type` and `-Name` parameters and any additional parameters you wish to change.

Update an existing target item

```
$computer = Set-RpsTargetItem -Type "Computer" -Name "Win137" -IsActive $false
```

Resource Items

Create Resource Items

Create a new resource item

```
$resourceItem = Set-RpsResourceItem -Type "type" -Name "name"
```

Update an existing resource item

```
$resourceItem = Set-RpsResourceItem -Type "type" -Name "name" -IsActive $false
```

Get Resource Items

Get all resource items

```
$allItems = Get-RpsResourceItem
```

Get resource items by properties

```
$foundItemsWithSpecificKBNumber = Get-RpsResourceItem -Filter @{"KbNumber" = "3135782"}  
$foundAllItemsWithKBNumber = Get-RpsResourceItem -Filter @{"KbNumber" = $null}
```

Resource Items can be filtered by a specific property that contains a given value.

```
$foundItemsWithPartialKBNumber = Get-RpsResourceItem -Filter @{"Contains-KbNumber" = "782"}
```

Get resource items by role

Resource Items can be retrieved by Role, which is a special property designated for tracking the purpose of a resource item or its relationship to a target item. To get resource items that have a specific role or have an assignment with a specific role, use the `-MatchAssignmentRole` parameter. See the section below on Resource Assignments for more info.

```
$clientAuthCerts = Get-RpsResourceItem -Type Certificate -Role "ClientAuth"  
$localAdmins = Get-RpsResourceItem -TargetItem $computer -Type Credential -Role "LocalAdministrator" -  
MatchAssignmentRole
```

Target / Resource Groups

Get target groups by name and properties : Sample 1

```
$foundGroupByName = Get-RpsTargetGroup -Name "Target Group Name"  
$foundGroups = Get-RpsTargetGroup -Filter @{"Color" = "Yellow"; "Food" = "Apples"; }  
$foundAllGroupsWithColor = Get-RpsTargetGroup -Filter @{"Color" = $null}
```

Target Groups can be filtered by a specific property that contains a given value.

```
$foundGroups = Get-RpsTargetGroup -Filter @{"Contains-Color" = "llow"; }
```

Get resource groups by Id. : Sample 1

```
$foundGroup = Get-RpsResourceGroup -Id $resourceGroup.Id
```

Get resource groups by properties. : Sample 2

```
$foundGroups = Get-RpsResourceGroup -Filter @{"Color" = "Yellow"}  
$foundAllGroupsWithColor = Get-RpsResourceGroup -Filter @{"Color" = $null}
```

Resource Groups can be filtered by a specific property that contains a given value.

```
$foundGroups = Get-RpsResourceGroup -Filter @{"Contains-Color" = "llow"}
```

Add resource group

```
$resourceGroup = New-RpsResourceGroup -Type "testGroupType" -Name "GroupName"
```

Add resource group with properties

```
$resourceGroup = New-RpsResourceGroup -Type "testGroupType" -Name "GroupName" -Properties @{ "Prop1" = 'Prop1' }
```

Add Resource Item to Resource Group

There are two methods to add a Resource Item to a Resource Group. A Resource Item can be assigned to a Resource Group during its initialization. The Resource Group must exist prior to assigning a Resource Item to a Resource Group during its initialization, as shown in the first example.

Add Resource Item in Resource Group in initialization

```
$resourceItem = New-RpsResourceItem -Type "testType" -Name "TestResourceItem" -ResourceGroup $TestResourceGroup
```

Additionally, if the Resource Item and the Resource Group already exist, the Resource Item can be added to the Resource Group, as shown in the next example.

Add Resource Item to Resource Group via API

```
$testResourceGroup = Get-RpsResourceGroup -Name "TestResourceGroup"  
$testResourceItem = Get-RpsResourceItem -Name "TestResourceItem"  
  
Add-RpsResourceGroupMember -ResourceGroup $testResourceGroup -ResourceItem $testResourceItem
```

Replace/Set child resource items and update properties on existing Resource Group

Using `Set-RpsResourceGroup` with the parameter `-ResourceItem` will replace any existing child resource items.

```
$testResourceGroup = Get-RpsResourceGroup -Name "TestResourceGroup"  
$testResourceItem = Get-RpsResourceItem -Name "TestResourceItem"  
$properties = @{Color = "Blue"}  
  
Set-RpsResourceGroup -ResourceGroup $testResourceGroup -ResourceItem $testResourceItem -Properties $properties
```

Add Target Item to Target Group

There are two methods to add a Target Item to a Target Group. A Target Item can be assigned to a Target Group during its initialization. The Target Group must exist prior to assigning a Target Item to a Target Group during its initialization, as shown in the first example.

Add New Target Item to Target Group

```
$targetItem = New-RpsTargetItem -Type "Computer" -Name "Server 150" -TargetGroup $servers
```

Additionally, if the Target Item and the Target Group already exist, the Target Item can be added to the Target Group, as shown in the next example.

Add Target Item to Target Group via API

```
$testTargetGroup = Get-RpsTargetGroup -Name "TestTargetGroup"  
$testTargetItem = Get-RpsTargetItem -Name "TestTargetItem"  
  
Add-RpsTargetGroupMember -TargetGroup $testTargetGroup -TargetItem $testTargetItem
```

Replace/Set child target items and update properties on existing Target Group

Using `Set-RpsTargetGroup` with the parameter `-TargetItem` will replace any existing child target items.

```
$testTargetGroup = Get-RpsTargetGroup -Name "TestTargetGroup"
$testTargetItem = Get-RpsTargetItem -Name "TestTargetItem"
$properties = @{Color = "Blue"}

Set-RpsTargetGroup -TargetGroup $testTargetGroup -TargetItem $testTargetItem -Properties $properties
```

Resource Assignment

A Resource Assignment is the assignment of a specific Resource Item to a specific Target Item. The association between the two items can contain Properties, a Status, and other useful information for automations. For example, a resource assignment can track the assignment of a Software Package (Resource) to a Computer (Target). The status may be used to indicate if that software is approved, installed, or up to date.

For convenience, you can specify a resource group in order to quickly assign multiple resources to a target. You can also specify a target group, or both!

NOTE

As of Release 2.2, duplicate assignments are not allowed. While allowing the assignment to groups, we are enforcing constraints to prevent duplicitous Assignments.

Create Resource Assignments

Assign a Resource Item to Target Item

```
$assign = New-RpsResourceAssignment -ResourceItem $software -TargetItem $computer
```

Assign a Resource Item to Resource Source

```
$assign = New-RpsResourceAssignment -ResourceItem $resourceItem -TargetItem $targetItem -ResourceState $approved
```

Assign multiple Resource Items to Target Items using Groups

```
$assignments = New-RpsResourceAssignment -ResourceGroup $securityHotfixes -TargetGroup $allComputers
```

Get Resource Assignments

Use the `Get-RpsResourceAssignment` cmdlet to retrieve assignments.

Get Assignments for a Target Item

```
$computer = Get-RpsTargetItem -Type "Computer" -Name "Win137"
$assignedCredentials = Get-RpsResourceAssignment -TargetItem $computer -Type $rps.ResourceTypes.Credential
```

Get Assignments by Role

Role is a special property used frequently in RPS to track the purpose of the resource item's association to a target item. The **Role** property can be placed on a Resource Item or the Resource Assignment, and can hold multiple values separated by the `|` symbol.

In this example, a Credential (Resource) is assigned to a Computer (Target). The assignment is given a Role of "LocalAdministrator". We can retrieve the designated Local Administrator credential for the computer by using the `-Role` parameter.

```
# assign credential and set roles
$computer = Get-RpsTargetItem -Type "Computer" -Name "Win137"
$credential = Get-RpsResourceItem -Type "Credential" -Name "RpsAdministrator"
$assignedCredential = New-RpsResourceAssignment -TargetItem $computer -ResourceItem $credential
$assignedCredential.Role = "LocalAdministrator|RpsUser"
Update-RpsResourceAssignment -ResourceAssignment $assignedCredential

# retrieve the LocalAdmin credential for the computer
$localAdminAssignment = Get-RpsResourceAssignment -TargetItem $computer -Role "LocalAdministrator"````
```

Update Resource Assignments

Like Task Assignments, Resource Assignments track the history of State changes. Update a resource assignment with the Update-RpsResourceAssignment cmdlet.

Update state of the patch assignment to Denied

The system default of a Resource Assignment is **Approved**. Here, we want to be able to set whether an administrator is going to **Approve** or **Deny** the patch. To manually update the Resource State for the patch, we run the following:

```
$assignment.ResourceState = "Denied"
Update-RpsResourceAssignment -ResourceAssignment $assignment
```

Create Patch Group

An updated feature for Release 2.2 is the ability to create Patch Groups. With Patch Groups, a user can add multiple children (members) to the group. This allows for the deployment of multiple patches, simultaneously.

Here, we are creating a new RPS Resource Group, called: `$patchGroup`.

```
# Creating patch group
$patchGroup = New-RpsResourceGroup -Type Patch -Name DemoPatches

# Add children to the patch group
Add-RpsResourceGroupMember -ResourceGroup $patchGroup -ResourceItem $hotfixJuly, $patch2, $patch3
```

Now that we have created our patch group and added our 3 children, we can assign our updated Patch to the Target Item.

Assign group of patches to a laptop

Assign the updated Patch Group to the Target Item, `$demoLaptop`.

```
New-RpsResourceAssignment -ResourceGroup $patchGroup -TargetItem $demoLaptop
```

Protected Properties

A Protected Property is a property in any property list that is marked as protected. Once a property is marked as protected, the data is encrypted and the property is marked as protected. To create a Protected Property use the cmdlet Set-Rps ProtectedProperty described below. To retrieve the value from a Protected Property use the cmdlet Get-RpsProtectedProperty as described below.

Get a Protected Property

Retrieve a protected property from a Rps item by passing in the item and name of the property. The cmdlet returns a SecureString.


```

$name = 'Property Name'
$secureResult = Get-RpsProtectedProperty -TargetItem $targetItem -Name $name
$secureResult = Get-RpsProtectedProperty -ResourceItem $resourceItem -Name $name
$secureResult = Get-RpsProtectedProperty -Node $node -Name $name
$secureResult = Get-RpsProtectedProperty -TaskMapAssignment $taskMapAssignment -Name $name
$secureResult = Get-RpsProtectedProperty -ResourceGroup $resourceGroup -Name $name
$secureResult = Get-RpsProtectedProperty -TargetGroup $targetGroup -Name $name

# return variable to plain text
$plainText = ConvertFrom-SecureString $secureResult

```

Set a Protected Property

To add or update a protected property to any Rps properties collection by passing in the item, name of the property, and a SecureString for the value. The cmdlet returns a true if successful.

```

#setup
$securePwd = ConvertTo-SecureString "Password" -AsPlainText -force

$result = Set-RpsProtectedProperty -TargetItem $targetItem -Name "Password" -Value $securePwd
Set-RpsProtectedProperty -TargetItem $targetItem -Name $name -Value $securePwd
Set-RpsProtectedProperty -ResourceItem $resourceItem -Name $name -Value $securePwd
Set-RpsProtectedProperty -Node $node -Name $name -Value $securePwd
Set-RpsProtectedProperty -TaskMapAssignment $taskMapAssignment -Name $name -Value $securePwd
Set-RpsProtectedProperty -ResourceGroup $resourceGroup -Name $name -Value $securePwd
Set-RpsProtectedProperty -TargetGroup $targetGroup -Name $name -Value $securePwd

```

Password Policy

A Password Policy is a set of password guidelines for a particular system or group of systems. The policy is saved in the database as a [Resource Item](#).

- The default minimum password length is 16 characters
- The default maximum password length is 64 characters
- A WhiteList, when specified, denotes the only characters allowed in a password
- A BlackList, when specified, denotes the only characters disallowed from the standard list, which can be found by executing the code below `PowerShell $charList = [Rps.Api.Utils.PasswordUtils]::DefaultPasswordCharacters`

Set Password Policy

The following code creates a password policy with a minimum and maximum length and saves that policy to the connected data store

```
$policy = Set-RpsPasswordPolicy -Name WindowsPasswordPolicy -MinLength 8 -MaxLength 16
```

Get Password Policy

The following code retrieves the policy that was previously created

```
$retrievedPolicy = Get-RpsPasswordPolicy -Name WindowsPasswordPolicy
```

Password

A Password is a secret string of characters used to gain access to something

Generate Password

A password can be generated using default parameters or by specifying a [Password Policy](#) to use. The password can be returned as either a plaintext or secure string by using the AsSecureString switch. Passwords are not persisted to the data store by default.

```
$newPassword = New-RpsPassword -AsSecureString
```

```
$policy = Set-RpsPasswordPolicy -MinLength 20 -MinUppers 5 -MinNumbers 3  
$newPassword = New-RpsPassword -PasswordPolicy $policy
```

Instance Definition

Instance Definitions are pre-defined data sets that consist of various Type Definitions, Items, and Associations between Items that can be used to create complex concrete objects.

Create Instance Definition

Creates an Instance Definition by providing a Name, Properties, and a Parent Node

```
$nodeId = "81B8272D-B49C-4350-A8F4-ABBB9CE29C68"  
$hs = @{  
    Prop1 = "value1"  
    Prop2 = "value2"  
}  
New-RpsInstanceDefinition -Name testName -Properties $hs -ParentNodeId $nodeId
```

Get Instance Definition

Retrieves an Instance Definition by ID or Name

```
Get-RpsInstanceDefinition -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Get-RpsInstanceDefinition -Name testName
```

Set Instance Definition

Creates or Updates Instance Definitions, associated Properties, and associated parent Node.

```
Set-RpsInstanceDefinition -Name $Name1  
Set-RpsInstanceDefinition -Name $Name1 -Properties @{Prop1 = "Value1"}  
Set-RpsInstanceDefinition -Name $Name1 -Properties @{Prop1 = "Value1"} -ParentNodeId $nodeId
```

Remove Instance Definition

Removes an Instance Definition by ID or by object

```
Remove-RpsInstanceDefinition -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Remove-RpsInstanceDefinition -InstanceDefDefinition $InstanceDefinition
```

Invoke an Instance Definition

Creates an instance from the definition using settings stored in a resource item.

```
Invoke-RpsInstanceDefItem -Settings $resourceItem -InstanceDef $instanceDefinition
```

Get Instance Definition Reference

Retrieves an Instance Definition Reference by providing an Instance Definition and an Instance Definition Item.

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$reference = Get-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -  
InstanceDefinitionItem $instanceDefItem
```

New Instance Definition Reference

Creates a new Instance Definition Reference.

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$taskMapIDs = "5b8b0340-091f-4823-b2f9-de937b5b4114", "a83b5445-3cc0-433e-b5e0-0fcf70389988"  
$reference = New-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -  
InstanceDefinitionItem $instanceDefItem -TaskMapIDs $taskMapIDs
```

Remove Instance Definition Reference

Removes an Instance Definition Reference by Instance Definition and Instance Definition Item.

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
Remove-RpsInstanceDefinitionReference -Name "name" -InstanceDefinition $instanceDef -  
InstanceDefinitionItem $instanceDefItem
```

Instance Definition Item

An Instance Definition Item is a part of an Instance Definition that can be used to create concrete items such as Target Items, Resource Items, etc.

Get Instance Definition Item

Retrieves an Instance Definition Item by ID, by Name, or by Resource Item.

```
Get-RpsInstanceDefinitionItem -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Get-RpsInstanceDefinitionItem -Name MyInstanceDefItem  
Get-RpsInstanceDefinitionItem -ResourceItem $resourceItem -Filter $filterHashtable
```

Create Instance Definition Item

Creates a new Instance Definition Item by providing an Entity Name, Name, Properties, and Type Definition ID

```
New-RpsInstanceDefinitionItem -EntityName testEntityName -Name name2 -Properties @{Prop1 = "Value1"} -  
TypeDefinitionId $typedefinition.id
```

Set Instance Definition Item

Creates or Updates Instance Definition Items and associated Properties.

```
Set-RpsInstanceDefinitionItem -Name $Name1 -TypeDefinitionId $id -EntityName $entityName  
Set-RpsInstanceDefinitionItem -Name $Name1 -TypeDefinitionId $id -Properties @{Prop1 = "Value1"} -EntityName  
$entityName
```

Remove Instance Definition Item

Removes an instance definition item by either its Id or the instance definition item

```
Remove-RpsInstanceDefinitionItem -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Remove-RpsInstanceDefinitionItem -InstanceDefinitionItem $InstanceDefinitionItem
```

Instance Definition Node

Create an Instance Definition Node

An Instance Definition Node is a wrapper for an RPS type and associated Properties.

```
New-RpsInstanceDefinitionNode -EntityName testEntityName -Name name2 -Hostname hostname -IPAddress 1.1.1.1 -  
SyncEndpointUrl syncEndpoint -certificateThumbprint certThumbprint -pollingInterval 1
```

Set Instance Definition Node

Creates or updates an Instance Definition Node.

```
Set-RpsInstanceDefinitionNode -Name name1 -EntityName testEntityName2 -Hostname hostname2 -IPAddress 2.2.2.2  
-SyncEndpointUrl syncEndpoint2 -certificateThumbprint certThumbprint2 -pollingInterval 2
```

Get Instance Definition Node

Get an Instance Definition Node by name.

```
Get-RpsInstanceDefinitionNode -Name name1
```

Remove Instance Definition Node

Removes an Instance Definition Node by ID or by object

```
Remove-RpsInstanceDefinitionNode -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Remove-RpsInstanceDefinitionNode -InstanceDefDefinitionNode $InstanceDefinitionNode
```

Remove Instance Definition Association

Removes an Instance Definition Association by ID or by object

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"  
Remove-RpsInstanceDefinitionAssociation-InstanceDefinition $instanceDef -PrimaryReference $instanceDefItem -  
Secondaryreference $instanceDefItem2
```

New Instance Definition Association

Creates an Instance Definition Association by ID or by object

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"  
New-RpsInstanceDefinitionAssociation-InstanceDefinition $instanceDef -PrimaryReference $instanceDefItem -  
Secondaryreference $instanceDefItem2
```

Get Instance Definition Association

Creates an Instance Definition Association by ID or by object

```
$instanceDef = Get-RpsInstanceDefinition -Name "MyDefinition"  
$instanceDefItem = Get-RpsInstanceDefinitionItem -Name "MyItem"  
$instanceDefItem2 = Get-RpsInstanceDefinitionItem -Name "MyItem2"  
Get-RpsInstanceDefinitionAssociation-InstanceDefinition $instanceDef -PrimaryReference $instanceDefItem -  
Secondaryreference $instanceDefItem2
```

Types

Set Target Type

To Create or Update a RPS Target Type.

```
Set-RpsTargetType -Name Computer -IsRoot
```

Set Resource Type

To Create or Update a RPS Resource Type.

```
Set-RpsResourceType -Name Host -IsRoot -EnableSubType
```

Set Sub Type

To Create or Update a RPS Sub Type.

```
Set-RpsSubType -Parent $patchDef -SubType 'CAB'
```

Set Child Type

To Create or Update a RPS Child Type.

```
Set-RpsChildType -Parent $def -ChildType $Rps.TargetTypes.NIC  
-DisplayName 'Network Adapters' -IsRequired -AllowMultiples
```

Set Type Property

To Create or Update a RPS Type Property template.

```
New-RpsTypeProperty -Parent $template -Name VmType -PropertyType Text -IsRequired -DefaultValue 2012R2
```

Set Type Resource Assignment

Creates or updates a Resource Assignment template.

```
Set-RpsTypeRA -Parent $template -ResourceType 'Host' -DisplayName 'Hypervisor Host' -IsRequired
```

Set Target Action

Associates an action (TaskMap Type) with a Target Type definition.

```
Set-RpsTargetAction -Parent $template -TaskMapType 'Provision-Vehicle' -Description 'Provision SNE'
```

Set Resource Group Type

Creates or updates a Resource Group Type

```
Set-RpsResourceGroupType -Name MyResourceGroupType -IsGroupReference
```

Set Target Group Type

Creates or updates a Target Group Type

```
Set-RpsTargetGroupType -Name MyTargetGroupType -IsGroupReference
```

RPS Sample Scenario – Tourism

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

The Rapid Provisioning System is designed to provide a mobile, modular, and extensible automation framework that allows coded automation activities to be executed across a wide-area network environment with limited connectivity and bandwidth. The breadth of customization options can sometimes be overwhelming, so this Sample Scenario has been designed as an example of what RPS can do even on a small scale. A working knowledge of RPS entities and architecture is recommended prior to reading this guide. It is recommended that this sample data NOT be imported into a production environment.

Sample Dataset

Tourism Sample Dataset

In this example, RPS is used to support the technical operations of a Grand Canyon Touring company, GC Tour Guides. GC Tour Guides must rely on satellite and line-of-sight networking solutions for guides on tour, due to the extreme geology of the terrain and lack of reliable network connectivity on tour routes. Guides are equipped with a handheld device that contains all the necessary information and applications to get through their tour routes, including maps, a GPS tracker, communications services such as email and instant messaging, and inventory management. Additionally, guides take a minimum of two pack animals – mostly donkeys – with them on the trails. Each donkey has a tracker attached with sensors that detect and report on many environmental and biological events or changes, such as GPS coordinates, current heading, humidity, and even health statistics of the donkey, such as skin temperature, heart rate, number of steps and so on.

GC Tour Guides has deployed RPS to automate configuration and data management between the guides' handhelds, the pack animals' trackers, and headquarters. Additionally, RPS is used to transfer patches and updated map files to these devices. Finally, GC Tour Guides defines and deploys the configuration of their devices through PowerShell Desired State Configuration.

The donkeys are grouped together in a Target Group (TG) called "Donkeys" with the type "Tracker." Each Donkey is defined as a Target Item (TI) within the Donkeys TG. Target Items can have several custom properties assigned, and in this example, we have defined Manufacturer and Model as custom TI properties. Another TG is created for the guides, with a guide TI named "Clarence." Handheld TIs are grouped in a TG called Handhelds.

Before a guide leaves on a tour, a TG is created with all the entities that are scheduled to go on the tour. In our example, the guide Clarence, two donkeys, and a handheld are scheduled for June 2017 tour within a TG called "Guide Tour June 2017." By grouping the entities within a TG, inventory can be tracked and alerts can be set to go off if a Donkey TI's GPS Coordinates drift too far from the Handheld TI's GPS Coordinates, for example.

Continuing, we group two Patches as Resource Items (RI) within a Resource Group (RG) called "TrackerPatches." These are patches that will apply to just Tracker TIs. Finally, the Patch RG is associated with the Donkeys TG using a Resource Assignment (RA), which creates a link between each patch RI and applicable donkey TI. Figure 1 illustrates the example dataset.

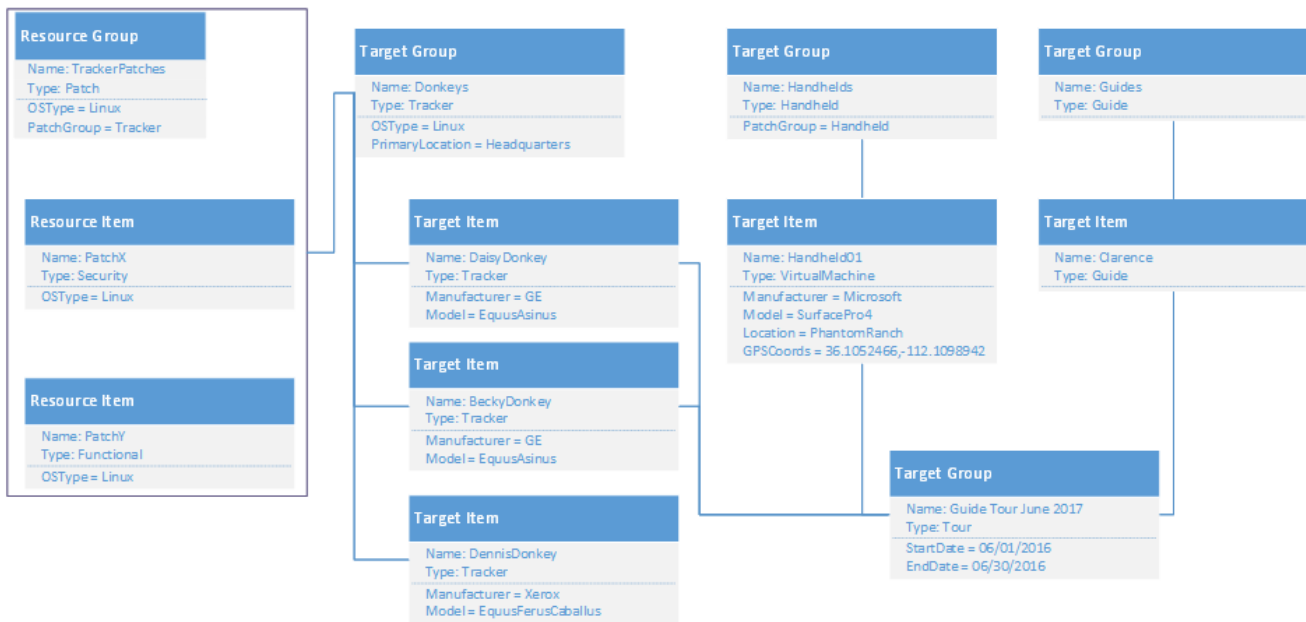


Figure 1 GC Tour Guides Example Dataset

Importing Sample Data

A fully functioning RPS Node is required to import the sample data from the script `Import-TourGuidesGCSampleData.ps1` (located at `$\Documents\Samples`). To import the sample data, simply run the script from any RPS machine from a PowerShell console as shown in Figure 2. You will initially be prompted to clean the RPS database of any existing RPS Entities.

```

Administrator: Windows PowerShell
PS C:\Users\administrator.RPS\Desktop> .\Import-TourGuidesGCSampleData.ps1

Confirm
Are you sure you want to perform this action?
Performing the operation "Import-TourGuidesGCSampleData.ps1" on target "Remove all RPS data from the cmdb (This will
remove all data from the RPS Database)?".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): _
  
```

Proceed as necessary, and the objects described in this document will be created in the RPS CMDB. Once the data is imported (expected output is shown in Figure 3), you can begin manipulating the data as necessary to familiarize yourself with the RPS system.

```

Administrator: Windows PowerShell
PS C:\Users\administrator.RPS\Desktop> .\Import-TourGuidesGCSampleData.ps1

Confirm
Are you sure you want to perform this action?
Performing the operation "Import-TourGuidesGCSampleData.ps1" on target "Remove all RPS data from the cmdb (This will
remove all data from the RPS Database)".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): a

Resulting Objects
Donkeys Group:

Name      Type      ChildItems      Properties
-----
Donkeys Tracker {Dennis, Daisy, Becky} {OSType, PrimaryLocation}

Guide Tour June 2017 Group:

Name      Type      ChildItems      Properties
-----
Guide Tour June 2017 Tour {Handheld01, BaseCampLaptop, Daisy, Becky} {EndDate, StartDate}

Handhelds Group:

Name      Type      ChildItems      Properties
-----
Handhelds virtualMachine {Handheld01} {Manufacturer, Model, GPSCoords, Location}

Patch Target Group:

Name      Type      ChildItems      Properties
-----
TrackerPatches Patch {PatchX, PatchY} {OSType, PatchGroup}

PS C:\Users\administrator.RPS\Desktop>

```

Figure 3 Expected output of the sample data import script

Desired State Configuration

GC Tour Guides would like to configure Clarence’s handheld with DSC to ensure that it is in the correct state during expeditions.

For this scenario, GC Tour Guides has decided that one of RPS’s existing DSC partial configurations will work for their needs. If this was not the case, they could author their own DSC partial configuration and add it to the system.

Before the configuration can be applied, we need to add some information to the database. Figure 4 below displays a partial section of the entity graph from Figure 1 with supplemental information specific to this DSC example. Two properties – ComputerName and ConfigurationName – have been added to the Handheld01 Target Item. Additionally, a Resource Group has been added that defines the configuration applied to the handheld. For more detailed information on how these properties are connected and what other DSC features are supported by RPS, reference the DSC specific documentation listed in the Supporting Documentation section of this document.

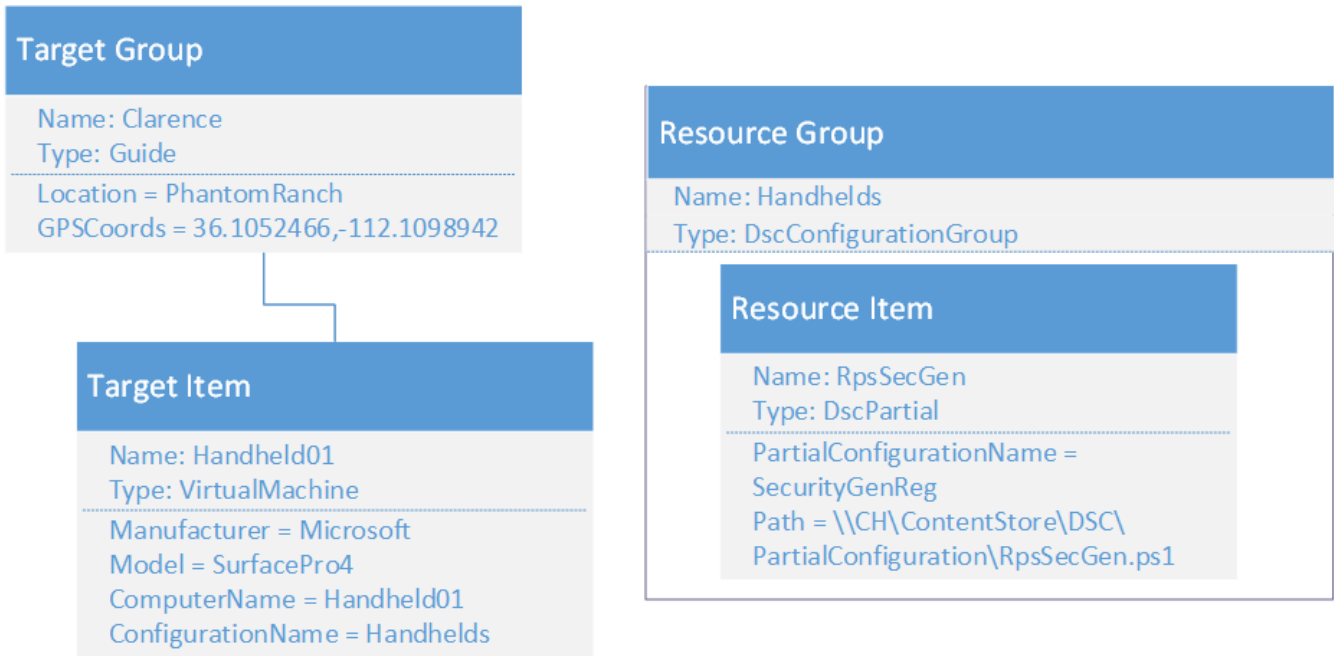


Figure 4 DSC Resource Group and Resource Item

Executing the Configuration

With the additional supporting DSC elements in place, a Task Item can be generated to execute the RPS Start-Dsc Runbook (which will apply the DSC configuration) and a Task Assignment to initialize the Runbook.

Executing the Import-TourGuidesGCSampleData.ps1 script with the DscDemo switch will implement the DSC configurations with the sample dataset.

NOTE

After running the Import-TourGuidesGCSampleData.ps1 script you will be able to view the process of the configuration within the RPS website. Since the hardware does not exist, it is unreachable, so the Runbook fails and displays the error through the UI.

Assignments

Target Item	Target Group	Task Workflow	Task Map	Status	Assignment Date	Message
Handheld01		Start-Dsc		ErrorStop	2017-01-05T23:07:27.263	WinRM cannot process the request. The following error occurred while using Kerberos authentication: Cannot find the computer Handheld01. Verify that the computer exists on the network and that the name provided is spelled correctly.

Figure 5 Task Assignment view in the UI shows runbook could not contact the Target Item.

Content Delivery Network

On Patch Tuesday, the GC guides are leading a group through the Grand Canyon. The GC IT team would like to patch the donkey's trackers with security patch X. First, however, they need to use the RPS CDN to distribute the patch from the parent RPS server stack to the Child RPS server.

To accomplish this goal, they will need to update their database to configure DFS-R to replicate the patch files. For this example, our script Import-TourGuidesGCSampleData.ps1 script with the CdnDemo switch will handle making the changes for us.

First, the script will create the below directory structure on our DFS-R endpoint in the parent node (the CH server).

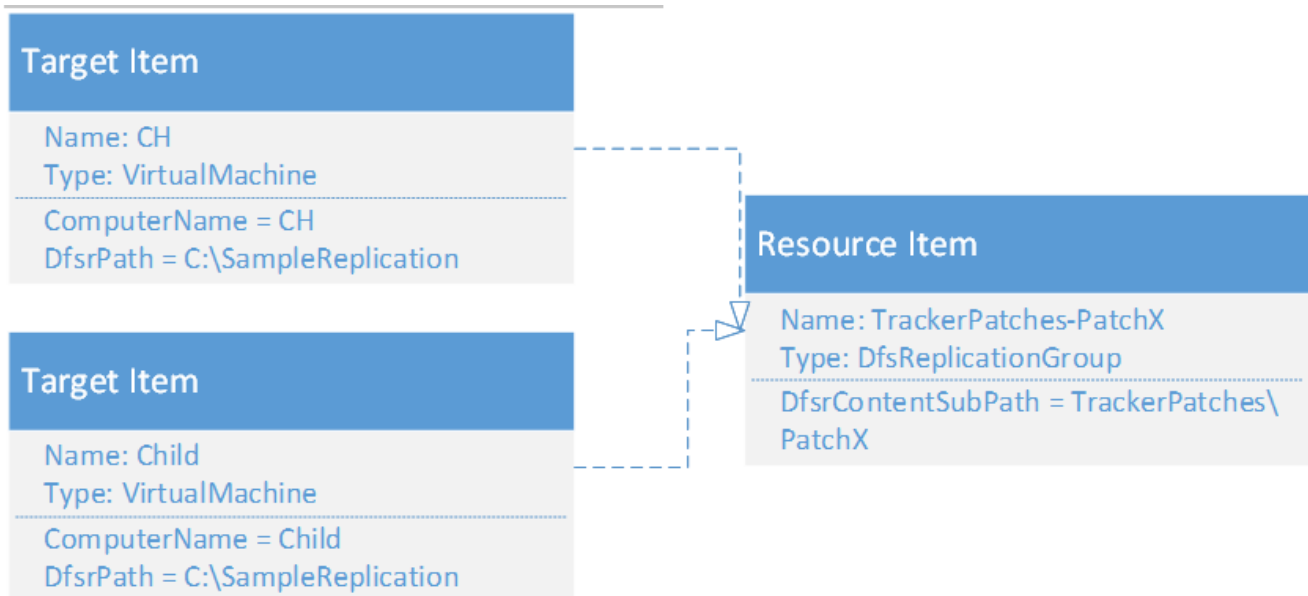
```

| C:\SampleReplication
|   TrackerPatches
|     PatchX
|       PatchX.smp1

```

After the directory structure is created the CMDB is used to create the resource item TrackerPatches-PatchX as well as a replication group for the directory. Once this is completed the CH and Child target items are created or updated with the DfsrPath properties. Since DFS-R can replicate information to different relevant paths on each endpoint, each target items sets its DfsrPath that is concatenated with the resource item's property DfsrContentSubPath. This allows each endpoint to specify its own location but supports a reasonable usage of relevant paths during automation. An assignment is created between the CH and Child target items with the DfsReplicationGroup.

In the end the RPS CMDB matches the below.

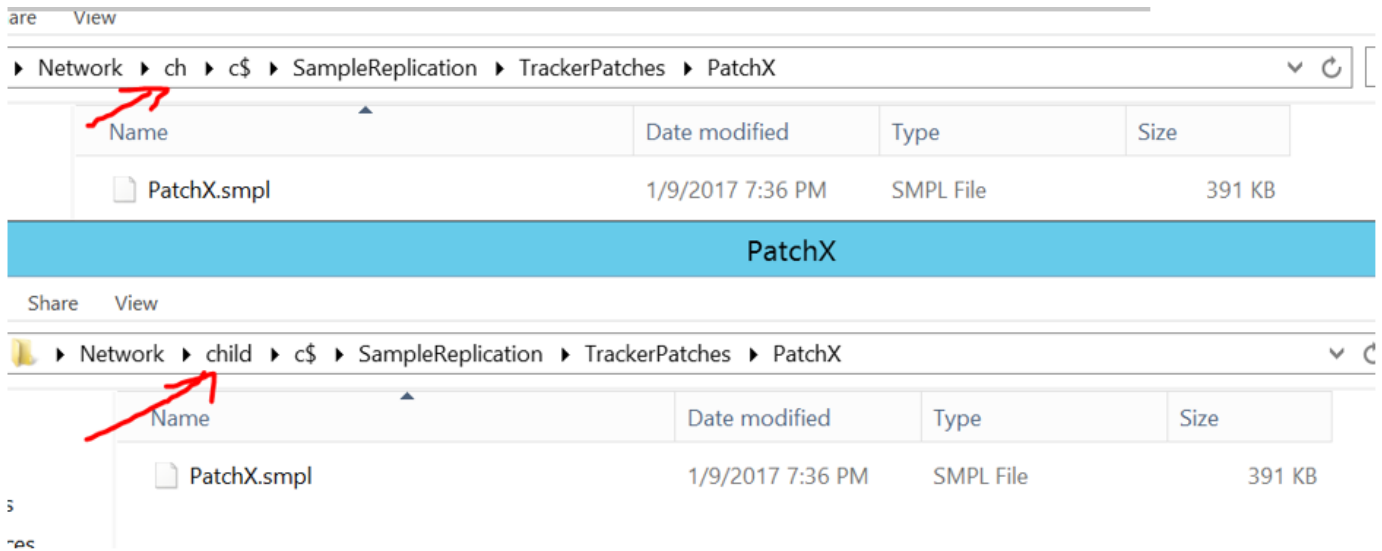


The DFS management console shows a replication group with two members, CH and Child.

The screenshot shows the DFS Management console for the replication group 'TrackerPatches-PatchX (RPS.local)'. The 'Memberships' tab is selected, showing two entries under 'Sending Member'.

State	Sending Me...	Sending Site	Connection...	Receiving ...	Receiving Si...	Schedule T...
Sending Member: CH (1 item)						
	CH	Default-First-...	Enabled	CHILD	Default-First-...	Replication G...
Sending Member: CHILD (1 item)						
	CHILD	Default-First-...	Enabled	CH	Default-First-...	Replication G...

After waiting a few minutes, the CH and Child server should have matching directories.



Sync Service

The GC Tour Guides company would like to administer and monitor patching and weather update tasks from their main office (HQ). They will use a mobile laptop acting as a full RPS Node to sync with HQ. The RPS Sync Service runs in the background and synchronizes Target Items, Resources and Tasking data down to a child node and telemetry data back up to the parent node.

In our demo, the RpsChild VM will act as the mobile laptop. We'll assign Target Items which are part of a guided tour to the Child Node, and the RPS Sync Service will automatically synchronize the data and related tasks to that node when a connection is available. We'll create a TaskMap to simulate update of our weather app and update to the latest weather data.

Run the Demo

Run the Import-TourGuidesGCSampleData.ps1 script on the CH.RPS.local VM with the -SyncDemo switch to demonstrate how the Sync Service will synchronize automations to a child node and return telemetry data.

Verify Target Item and Assignments

From the RPS VM, open RPS Web at <http://localhost:8080>. You should see containers imported from the sample script. Although it isn't visible here, the BaseCampLaptop Target Item (TI) is assigned to the Child.RPS.local Node, and all automations will run there.

Containers				
Name ↑	Type	IsActive	# Of Children	Pending Actions
BaseCampLaptop	Laptop	true	0	
Becky	Tracker	true	0	
Daisy	Tracker	true	0	
Dennis	Tracker	true	1	
Handheld01	virtualMachine	true	0	

The BaseCampLaptop TI is assigned to a Task Map with two sequential steps. Step 1 is an approval step for updating the Weather App, and Step 2 imports the latest weather data. Click on the BaseCampLaptop link or navigate to Tasking, Assignments to see the two tasks. Initially, the Status should appear to be NotReady, but after a few minutes, Step 1 will run on the child node.

Assignments

Target Item ↑	Target Group	Workflow	Task Map	Status	Assignment Date	Message
BaseCampLaptop		Approve-WeatherAppV2	Update-BaseCampWeatherV2	PendingUserAction	2017-01-10T14:13:31.653	Approve-WeatherAppV2 was started, and is waiting on user approval
BaseCampLaptop		Import-WeatherUpdates	Update-BaseCampWeatherV2	NotReady	2017-01-10T14:13:31.667	

Open RPS Web on the child VM. Notice that the BaseCampLaptop TI has been synchronized from the parent node, but not the other items. All the related Tasking and Target data has been synchronized as well.

Containers

Name ↑	Type	IsActive	# Of Children	Pending Actions
BaseCampLaptop	Laptop	true	0	Show

Step 1 - Approve Weather App Update

The first assignment in our Task Map requires user approval. This approval can be done via RPS Web at the parent or child node. The first Task Assignment should be in the PendingUserAction status, and you will see a button to show Pending Actions. Click the Show button to open the approval.

The screenshot shows a modal dialog box titled "BaseCampLaptop - Laptop". It has a "Pending" section with a table:

Target Item	Message	Actions
BaseCampLaptop	Approve Weather App Update?	Approve Deny

Below this is a "Canceled" section with a table:

Target Item	Actions

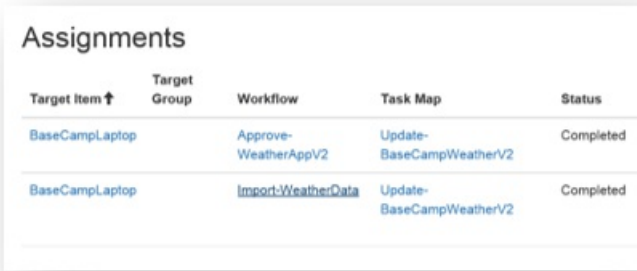
At the bottom right of the dialog is a "Close" button. In the background, a "Pending Actions" section with a "Show" button is visible.

After approval, the Task Assignment status will be completed, and the status will synchronize between child and parent nodes.

Step 2 – Import Weather Data

Step 2 will update to Ready status once Step 1 is complete. Then, the runbook to update the weather data will begin and the status will change to Running and finally Completed. The sample runbook, Import-WeatherData, will simulate the update by

adding a special property to the BaseCampLaptop TI.



Target Item ↑	Target Group	Workflow	Task Map	Status
BaseCampLaptop		Approve-WeatherAppV2	Update-BaseCampWeatherV2	Completed
BaseCampLaptop		Import:WeatherData	Update-BaseCampWeatherV2	Completed

Open RPS Web to verify the Task Assignments are all synchronized. Click the BaseCampLaptop link to view the properties, and observe the property denoting the weather data version. This property is prefixed with “__SYNC:”, which tells the Sync Service to synchronize the value back to the parent node.



BaseCampLaptop - Laptop

Actions 0

Properties 7

```
Id 80bd54e3-9b0a-4e88-ab07-5eaf7948ce40
ParentItemId 80bd54e3-9b0a-4e88-ab07-5eaf7948ce40
Name BaseCampLaptop
Type Laptop
IsContainer 
IsActive 
__SYNC:WeatherDataVer 201606.2
```

Additional Scenarios

This sample just uses a simple Target Item and Task Map to illustrate how the Sync Service operates and how automations can be viewed and controlled remotely. In a more complex scenario, we'd see more Nodes and Containers with many child items representing physical or virtual devices, and complex task maps that administer patches, collect data, etc. In the GC Tour Guides example, we could imagine that each guided group has a Laptop which acts as an RPS Node. The physical things, including donkeys, people, handheld devices and communication equipment may be represented as Target Items in one container. Headquarters, which acts as the parent RPS Node, would monitor all of the various groups in RPS Web.

Conclusion

RPS is a very customizable solution, which can sometimes be overwhelming. This sample scenario describes a simple implementation of how RPS can be used to accomplish automation tasks with minimal overhead.

More Resources

- [RPS Software Design](#)

RPS Install Guide

Last updated on September 14, 2021.

Document Status: Document Feature Complete as of September 14, 2021; PENDING EXTERNAL REVIEW.

Purpose

The purpose of this document is to provide an overview of the Rapid Provisioning System (RPS) Install process and detailed instructions for setting up a virtualized RPS Environment using Hyper-V.

Audience

This document is intended for Lead Systems Integrators (LSI), Field Service Representatives (FSR), IT staff, and Developers. Users should have some familiarity with core RPS Concepts as well as basic PowerShell and Windows Systems operations.

System Requirements

1. Windows 10/Server 2016
2. PowerShell/WMF 5.1
3. 16GB RAM
4. 100GB HDD free space
5. Hyper-V PowerShell Module & Management Tools

NOTE

RPS Authored content is signed, but 3rd party code may not be. RPS Installer was tested with PowerShell Execution Policy set to RemoteSigned.

Ports, Protocols, and Service Accounts

RPS specific Port, Protocol, and Service Account information can be found in the [Ports, Protocols, & Security Guide](#).

NOTE

DomainAdmin membership is required to create a new Domain Controller. After initial creation, the account should be removed from DomainAdmins, but should still retain permissions to manage AD Users, Computers, Groups, and OUs.

RPS Installation (Default)

Installing RPS requires the latest RPS release, the install media for PostgreSQL, Windows features, and a Hyper-V image for the Windows 2012R2 VMs that will be created. The instructions below can be used to build a default Root RPS Node, which is also a Domain Controller for the root.local domain and the application server for RPS. You can also choose to create a NOSC Node, TCN Node, and SNE Node based on available resources.

Media and Content Store

Before installing RPS, ensure you have gathered all required media from your Distribution Source, LSI, or FSR. Extract and save the RPS media to a location such as **C:\RPS**.

NOTE

This location will be referred to as **Install Root**.

Once you have completed extracting and saving the RPS media, the **Install Root** folder should contain the following sub-directories:

- \ContentStore
 - \Certificates
 - \CMDB
 - \ConfigurationData
 - \Demos
 - \DeploymentShare
 - \Documents
 - \DSC
 - \Export
 - \GenerateDV
 - \Images
 - \iPxeDistro
 - \Modules
 - \Office
 - \Packages
 - \Patches
 - \Plugins
 - \PostgreSQL
 - \Provisioning
 - \RpsBitsDownloadService
 - \RpsCdn
 - \RpsGui
 - \RpsProvisioning
 - \RpsSync
 - \RpsTaskManagement
 - \RpsWebApi
 - \Runbooks
 - \RvpsGui
 - \Setup
 - \SQLSecurity
 - \SystemTest
 - \TrustElementRepository
 - \Utilities
 - \Windowserver2012

Extract and/or save the Windows Server image to a location other than the previously identified **Install Root**, such as **C:\WindowsSvr**.

Install RPS

The example script is responsible for preparing the PowerShell session so that the Install-RpsNode cmdlet can be executed. The steps required to complete the preparation can be different from those outlined in the example script. The example script provided will simply ensure that:

1. All the files in **InstallRoot** are unblocked so they can be executed
2. That the correct modules can be located and are imported
3. The context has administrative privileges
4. Creates an RPS session in memory and imports all TaskMaps, DscPartials, and Types into the session

5. Generate CMDB data in the RPS Session (in memory)
6. Ensures that the correct parameters are used to execute Install-RpsNode cmdlet

How to Execute Install-RpsNode

1. Open PowerShell as administrator. Right-click the PowerShell Icon from Start Menu or Task Bar and select **Run as administrator**.

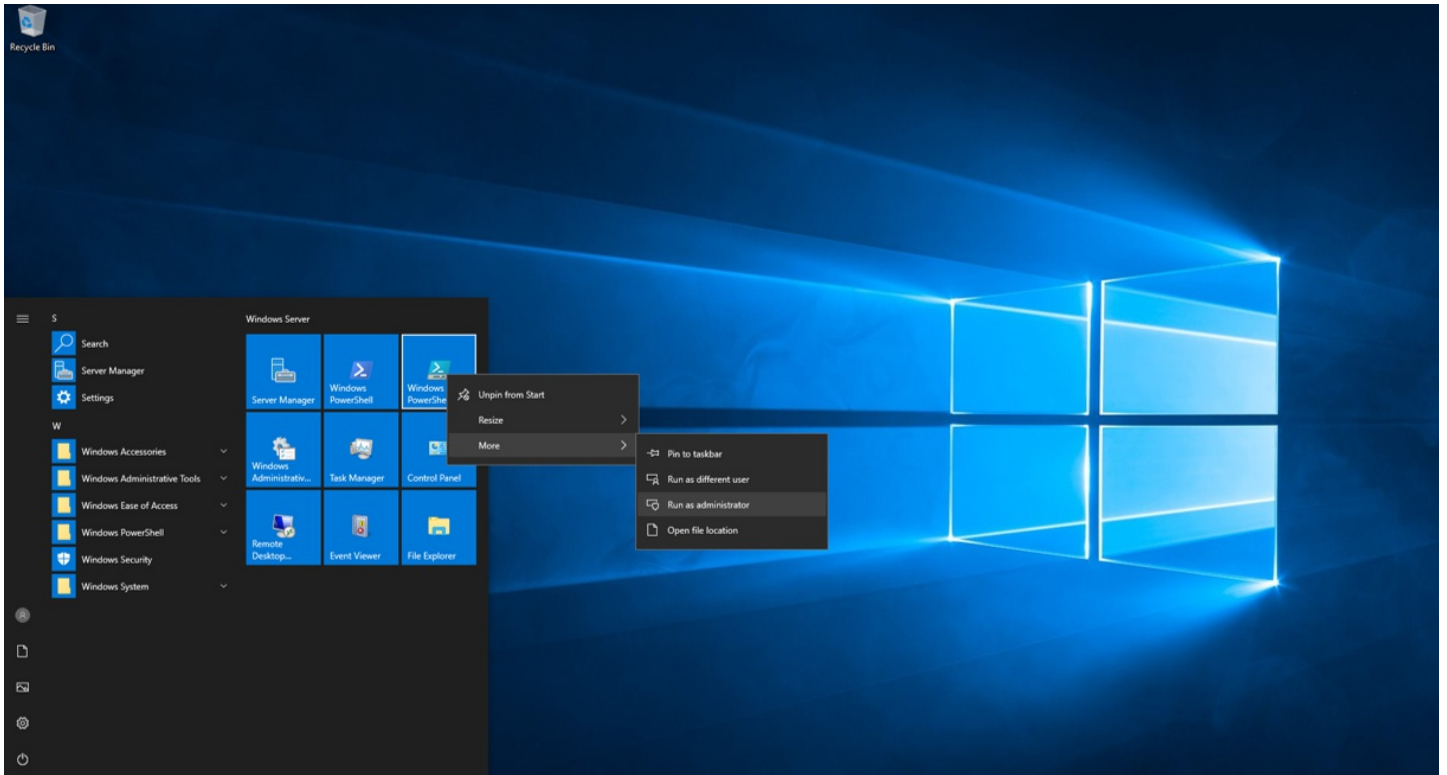


Figure 1: Run PowerShell as administrator

1. Set location to **Install Root**\ContentStore\ by executing the following cmdlet:

```
Set-Location c:\RPS\ContentStore\
```

2. Install RPS and supply the location of the VM Template VHDX, the NodeType and, if needed, specify specific configuration using -NodeConfigurationName. See examples below:

```
Install-Rps.ps1 -VMTemplateFilename D:\Common\Windows_Server_2012_R2_VL-dev.vhdx -NodeType 'root'
```

[Install-Rps.ps1](#)

```
-VMTemplateFilename D:\Common\Windows_Server_2012_R2_VL-dev.vhdx -NodeType root, nosc -SkipDscModuleCopy -DeleteVMs
```

NOTE

For further options, refer to the [Install-Rps Parameter Definition](#) table below.

Install-Rps Parameter Definitions

PARAMETERS	DESCRIPTIONS
VMTemplateFilename (Required)	Path to the .vhdx file which will serve as the Hyper-V VHDX template.
RhelTemplateFileName	Path to the .vhdx file which will serve as the Hyper-V VHDX template used to create RHEL VMs.

PARAMETERS	DESCRIPTIONS
VhdFolderPath	Path to a folder to store the .vhdx files used to create VMs. If not specified, .vhdx files will be stored in the same directory path as specified with VMTemplateFileName .
ConfigFileName	Name of file containing the RPS Configuration data. If one is not specified, it will be created in the Export directory.
NodeType (Required)	The desired node you want to install (i.e., Root). If you want to install all nodes then choose 'All'.
SkipVMCreation	Switch to use Virtual Machines already created and configured with proper networking.
SkipCopyContent	Switch to not copy InstallRoot to the CDN server (i.e., CH).
SkipDSCPrep	Switch to not setup DSC MOF Encryption, WinRM HTTPS Listener, and to not copy DSC Resources.
ComputerName	Array of the Virtual Machines you want to install from the Node (i.e., Root).
SkipDSCPublish	Switch to skip publishing DSC.
CopyCDN	Switch to copy patches that were imported into the configuration.
DeleteVMs	Switch to delete the Virtual Machines if they exist during a new installation.
SkipDscModuleCopy	Switch to not copy DSC Required Modules to the target's PowerShell module path.
Esxi	Switch to indicate whether the hypervisor is ESXi (default is Hyper-V).
TaskMapName	Name of the desired system task map that will be assigned to the target items. Default value of Install-Rps.

Manual Node Installation/Repair

1. The Install-RPSNode function from the RPS-Installer module will manually start a install for the Node you specify. See example below:

```
Enter-RpsSession -Path "D:\Exports\Root.xml"
Install-RpsNode -NodeType "Root"
```

NOTE

Refer to the parameter table below for available options when manually running the 'Install-RPSNode' function.

PARAMETERS	DESCRIPTIONS
VMTemplateFilename (Required)	Path to the .vhdx file which will serve as the Hyper-V VHDX template.
RhelTemplateFileName	Path to the .vhdx file which will serve as the Hyper-V VHDX template used to create Rhel VMs.
VhdFolderPath	Path to a folder to store the .vhdx files used to create VMs. If not specified, .vhdx files will be stored in the same directory path as specified with VMTemplateFileName .
NodeType (Required)	The desired node you want to install (i.e., Root).

PARAMETERS	DESCRIPTIONS
SkipVMCreation	Switch to use Virtual Machines already created and configured with proper networking.
SkipCopyContent	Switch to not copy InstallRoot to the CDN server (<i>i.e., CH</i>).
SkipDSCPrep	Switch to not setup DSC MOF Encryption, WinRM HTTPS Listener, and to not copy DSC Resources.
ComputerName	Array of the Virtual Machines you want to install from the Node (<i>i.e., @App.Root.local</i>).
SkipDSCPublish	Switch to skip publishing DSC.
CopyCDN	Switch to copy patches that were imported into the configuration.
DeleteVMs	Switch to delete the Virtual Machines if they exist during a new installation.
SkipDscModuleCopy	Switch to not copy DSC Required Modules to the target's PowerShell module path.
Esxi	Switch to indicate whether the hypervisor is ESXi (default is Hyper-V).
ContentStorePath	Path to the content store.
TaskMapName	Name of the desired system task map that will be assigned to the target items. Default value of Install-Rps.

Ports, Protocols, & Security Guide

Last updated on August 3, 2021.

Last Reviewed and Approved on PENDING REVIEW

Ports and Protocols

RPS uses various ports and protocols for operation. Some ports are configurable as part of the RPS deployment and configuration, and some are outside the management of RPS and/or not configurable. The table below shows these RPS components (where * indicates port is configurable via RPS).

COMPONENT	DESCRIPTION	PORTS	PROTOCOLS
RPS API	Direct management of configuration data in PostgreSQL Server.	5432	TCP
RPS Sync Plugin	Synchronize Data and Static Files between RPS Nodes and managed RPS Targets.	777*	HTTPS
DFSR	Transfers files between nodes within a domain.	445, 135	RPC, TCP
BITS	Transfers files between nodes on different domains.	80, 443	HTTP/S
RPS Web	Administrative Website for RPS.	8080*	HTTPS
RPS Provisioning Service	Bare-metal/iPXE Service via the specific DNS name rpsprovisioning .	443*	HTTPS
TER Reader	Trust Element Repository – Reader (DCA)	3443	TCP
TER Writer	Trust Element Repository – Writer (DCA)	5443	TCP
WinRM	Windows Remote Management	5985/5986	HTTP/HTTPS
SMB	File Sharing	445	SMB/HTTPS
ICMP	Device Availability		ICMP
DHCP	Dynamic Host Configuration Protocol	67-69	UDP
DNS	Domain Name Server	53	UDP/TCP

Table 1: RPS Ports and Protocols

The Host-Based Security System (HBSS) uses some common ports (e.g., ports 80, 443, 1433, etc.), though it requires additional ports be used for full operation. Please see the HBSS documentation, at <https://kc.mcafee.com/corporate/index?page=content&id=KB66797>.

Service Accounts

The following RPS accounts are used by RPS for the setup and maintenance of RPS nodes.

RPS Account Roles: Domain Accounts

ACCOUNT (ROLE)	DESCRIPTION	PERMISSIONS
DomainAdmin	Has full control of the domain. Administrator rights on all domain controllers and member servers.	AD: Domain Admin ¹
DomainJoinAdmin	Used to join computers to the domain. Rights are scoped specifically for that purpose.	AD: Force change password Read/Write Computers
ProvisioningServiceAccount	Provisioning Website App Pool Identity	SQL: Service Permissions ²
GuiServiceAccount	RPS Website App Pool Identity	AD: Domain Admin ¹ RPS: Master Key Encryption
SqlServiceAccount	SQL Server Service	AD: Log on as a Service
CdnServiceAccount	Has Access to the CDN folder.	CDN Folder, BITS Message Queue
DFSAdmin	Has minimum required permissions in Active Directory to manage DFSR.	AD: DFSR Management
MasterKeyEncryption	Users with this role will be granted read permissions to the MasterKey certificate private key.	Read Only
PluginClientAuth	Users with this role will get the RPS Web API client authentication certificates installed in their certificate store.	
WebApiServiceAccount	Account used to run the Web API service and Sync.	AD: Domain Admin ¹ SQL: Service Permissions ²
FileTransferServiceAccount	Account used to transfer files from ContentStore.	ContentStore NTFS permissions
DhcpServiceAccount	Account used to authorize DHCP.	DHCP Admin
DomainSchemaAdmin	Account used to extent the AD Schema and move domain controllers to different AD site.	AD: Schema Administrator, Domain Admin ¹
DomainUser	Account used to export certificates from Root and CA local machine certificate store (ClientPki.ps1).	Logon Permissions
TaskManagementServiceAccount	Domain level service account authorized to execute runbooks across the RPS domain.	AD: Domain Admin ¹
ServerAdmin	Push certificates and settings; manage DSC configuration; pull files from content store.	AD: Domain Admin ¹

Table 2: Domain Accounts

¹ Domain Administrator membership is required to create a new Domain Controller. After initial creation, the account should be removed from this group, but should still retain permissions to manage AD Users, Computers, Groups, and OUs.

² Service SQL permissions are scoped to the RpsDb only and include Execute, Select, Insert, Update, Delete, and SyncHistory change tracking view permissions.

RPS Account Roles: Server Accounts

ACCOUNT (ROLE)	DESCRIPTION	PERMISSIONS
LocalAdmin	Manage machine settings for non-domain joined computer.	Local Admin
FileTransferServiceAccount	Account used to transfer files from ContentStore.	ContentStore NTFS permissions
VMWareAdmin	Account used for VMWare configuration.	VMWare Administrator
LocalAdminProvisioningOnly	Local Admin account, but only used for provisioning.	Local Admin

Table 3: Server Accounts

RPS Account Roles: Other

ACCOUNT (ROLE)	DESCRIPTION	PERMISSIONS
PostgreSqlSuperAccount	PostgreSQL administrative account.	SQL
DatabaseAccount	PostgreSQL account used by RPS to connect to the database.	SQL
DomainSafeModeAdmin	Account that credentials are used to create the Domain Controller DSRM password; only used in ADSitesAndSubnets.	

Table 4: Other Accounts

Security

Partial Configurations

All RPS partial configurations must define the following parameters:

- **IPAddress** - Accessible IP Address of the computer we will publish DSC Configuration to.
- **DSCEncryptionCertificate** - Information about the certificate used to encrypt the MOF (configuration). The LCM is set to use this certificate and any partials that are not secured will not run on a target.
- **OutputPath** - Location to temporarily store the MOF file once it is compiled.

For additional information, refer to the RPS article [Authoring RPS DSC Partial Configurations](#).

RPS Runbooks

Many RPS PowerShell runbooks will need to connect to the Target (Computer) to perform their duty. To connect, you must get the appropriate credential and then establish a secure connection.

Runbooks use the `Get-RpsCredential` or `Get-AdminRoleCredential` cmdlet to load the right credential for the target, then uses `New-SecureSession` from Rps-Api to make the connection.

For additional information, refer to the RPS article [Authoring RPS Runbooks](#).

Patching

Patch Management in RPS requires communication via HTTPS. The certificate authority (CA) that signed the web server's certificate must be trusted by the Linux client or patches will not be downloaded. This is done by installing the public certificate of the CA.

For additional information, refer to [RPS Patching](#).

Certificates

The RPS Solution uses certificates for a variety of functions, including:

- Website SSL binding for HTTPS encrypted transport between server (e.g., RPS Website) and client.
- RPS Sync Service for client/server authentication between **subscriber** (e.g., RPS Sync Service on Region) and **distributor** (e.g., RPS Sync Service on Master) nodes. The certificate thumbprints for all trusted nodes are whitelisted in the RPS CMDDB.
- RPS Sync Service for HTTPS encrypted transport between server and client.
- DSC MOF file credentials encryption (by default, DSC encrypts the entire MOF file).
- Client Authentication for the DSC Pull Server.
- WinRM for HTTPS encrypted transport between server and client.
- SQL for HTTPS encrypted transport between server and client.
- Provisioning SSL binding for HTTPS encrypted transport between server (e.g., RPS Provisioning) and client.
- Encryption of secrets in the database (protected properties).
- Encryption of XML configurations.

Each certificate must be derived from a trusted root certificate that resides in the Trusted Root Certification Authorities store in Certificate Manager on the RPS server(s).

ROLE	DISTRIBUTION	KEY USAGES	PURPOSE
DscEncryption	Per VM	Key Encipherment, Data Encipherment (30)	MOF credential encryption.
DscPullServer	Per VM	DigitalSignature, Client Authentication	DSC Pull Server Client Authentication
ProvisioningSSL	APP Master	Key Encipherment, Data Encipherment	HTTPS support for Provisioning Website.
RpsClientCdn	Per VM	Client Authentication	Patching Certificate Authentications.
RpsGuiSSL	Per APP VM	Digital Signature, Non-Repudiation, Key Encipherment (e0)	HTTPS support for RPS GUI Website.
iPxeSSL	Per APP VM	Digital Signature, Non-Repudiation, Key Encipherment (e0)	HTTPS support for iPXE Website.
MasterKeyEncryption	Per Node	Document Encryption, Key Encipherment, Data Encipherment	Protecting the Master Key.
NodeEncryption	Per APP VM	Document Encryption, Key Encipherment, Data Encipherment	Encrypting node configuration.
RpsRoot	Per VM	Certificate Signing, Off-line CRL Signing, CRL Signing (06)	Deriving other certificates.
RpsSync	Per APP VM	Client Authentication	Allowing Sync to occur between nodes.
RpsSyncSSL	Per APP VM	Key Encipherment, Digital Signature, Non-Repudiation	Data-in-transit encryption for node sync.
SqlSSL	Per APP VM	Server Authentication	Data-in-transit encryption for SQL data.
WinRM	Per VM	Server Authentication, Key Encipherment	Secure Connections to Targets.
CertificateApi	Per VM	Client Authentication	Certificate Manager API REST Certificate Client Authentication.

ROLE	DISTRIBUTION	KEY USAGES	PURPOSE
CertificateManager	Per VM	Client Authentication	Certificate API REST Certificate Client Authentication.
RpsAPI	Per VM	Client Authentication	RPS API REST Certificate Client Authentication.
RpsWebAPISSL	Per APP VM	Digital Signature, Non-Repudiation, Key Encipherment (e0)	HTTPS Support for RPS Web API Host.
WindowActivation	All	Digital Signature	CA Chain to Activate Office and Windows.
WindowsActivationCA	All	Digital Signature, Certificate Signing, Offline CRL Signing, CRL Signing	CA Chain to Activate Office and Windows.

Table 5: Certificates

Master Key

The Master Key (MK) is used to protect secrets in the database (i.e., credential/certificate passwords). Since the MK is high value, it is encrypted using the public key of a certificate. Appropriate users are given access to the private key of the Master Key Encryption Certificate (MKEC) so that they may access the MK and decrypt protected properties in the database.

The same MK should be used for all nodes that will share secrets. The default boundary for secrets is an Active Directory domain since domain accounts will likely need access to all domain computers. This implementation is fungible; however, any changes to the default implementation made by the customer/integrator may risk customer data.

Accounts that are preconfigured with the MasterKeyEncryption role during setup will have permissions to manipulate protected properties in the target environment. In order to give this permission to new users once RPS is installed, the role should be added to the appropriate account in the CMDDB and DSC should be republished (at minimum, the RpsCertificate partial).

When a protected property is retrieved or set, access is determined by retrieving the MasterKeyCertThumbprint property on the node. If the user has access to the corresponding certificate private key in the LocalMachine\My store, they are granted access to the MK. If the user does not have rights to the MKEC, access to protected properties will be denied.

PostgreSQL

Last updated on February 16, 2021.

Last Reviewed and Approved on PENDING REVIEW

We are utilizing PostgreSQL 12.4 to maintain compatibility with Windows Server 2012 R2.

NOTE

External connectivity is authorized through a firewall rule that allows inbound TCP connections to Port 5432. See [Ports, Protocols, & Security Guide](#) for more details.

PostgreSQL 12 Documentation

Official PostgreSQL Documentation:

<https://www.postgresql.org/docs/12/index.html>

Accessing the Server

Once installed, PostgreSQL can be accessed by running pgAdmin on any of the following supported browsers:

- Edge
- Firefox
- Chrome

WARNING

Internet Explorer is not supported.

pgAdmin 4 is bundled with the PostgreSQL installation package.

pgAdmin 4 Documentation

Official pgAdmin Documentation:

<https://www.pgadmin.org/docs/pgadmin4/4.24/index.html>

Guidance for Pull Requests into the COMMON Repository

Last updated on December 11, 2020.

Last Reviewed and Approved on PENDING REVIEW

Introduction

Pull Requests (PR) allow a developer the opportunity to have multiple sets of eyes on their code in a collaborative space. This helps to ensure the best possible code is deployed into the live environment. These instructions provide guidance on how to submit a Pull Request in the RPS Common Repository (Repo).

Assumptions

1. You have access to the Mission Network RPS Azure DevOps instance at www.devops.peoc3t.com.
2. You are submitting a PR to the COMMON repo NOT the Core Repo.
3. You are using and familiar with Visual Studio to create/edit/submit code changes. Pull Request Etiquette

Pull Request Etiquette

Sizing

- Simplify the amount of changes in your pull request. A large pull request (>100 lines of code changed) usually indicates that the user story was not properly decomposed.
- o Smaller PRs allow for quicker reviews and subsequent merges to the default branch, leading to increased work velocity
- o Smaller PRs allow for quicker defect identification and mitigation, since the scope of changes are much more discrete

Naming

- Provide meaningful descriptions for every commit. This assists PR reviewers to better understand the context of code changes and for version history
- o For some great guidance, see: <https://chris.beams.io/posts/git-commit/>

Expectations

- Be polite. Conversations should occur as they would if you were in person.
- This is a team environment and success is dependent upon being accountable to each other. If you are assigned a PR, do your best to perform the review in a timely manner. If a blocker exists, let your scrum master, project manager or PR creator know as soon as possible.
- When making a comment as a PR reviewer, take the time to make a recommendation. This will help others grow and potentially avoid the same mistake in the future.

Culture

- Code reviews are a safe place and meant to be a tool for communication.
- Review the code, not the person. As a reviewer, be aware that your sole job is to review the changes, not make assumptions about someone's personality.
- On the flip side, as the owner of the code review understand that any feedback you are getting should not be considered a personal attack on you. It is merely a review of the code.
- Leave the communication channel open for discussion as long as needed. Marking comments resolved without a chance for the commenter or owner to reply closes that channel prematurely when it could've otherwise provided more valuable conversation.

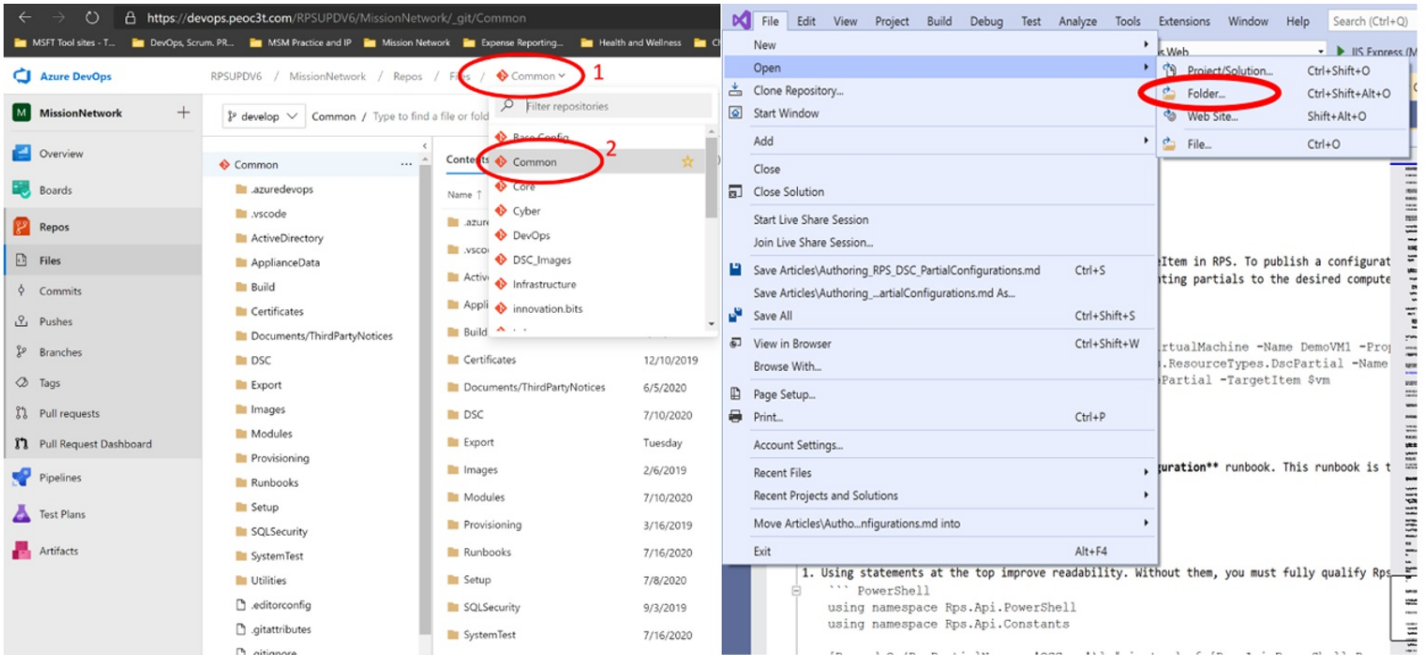
PR Process

PR Process



**Not part of this document

Note: It is critical to ensure you are working in the Common repo in ADO and the Common folder in Visual Studio.



Branch Creation

Before you begin development, a branch should be created for the work in Visual Studio.

Branch Naming

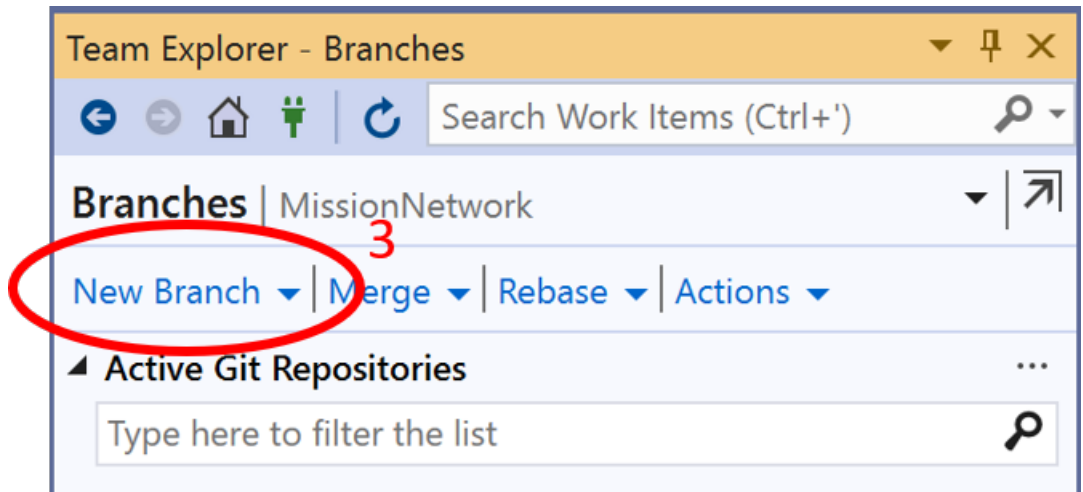
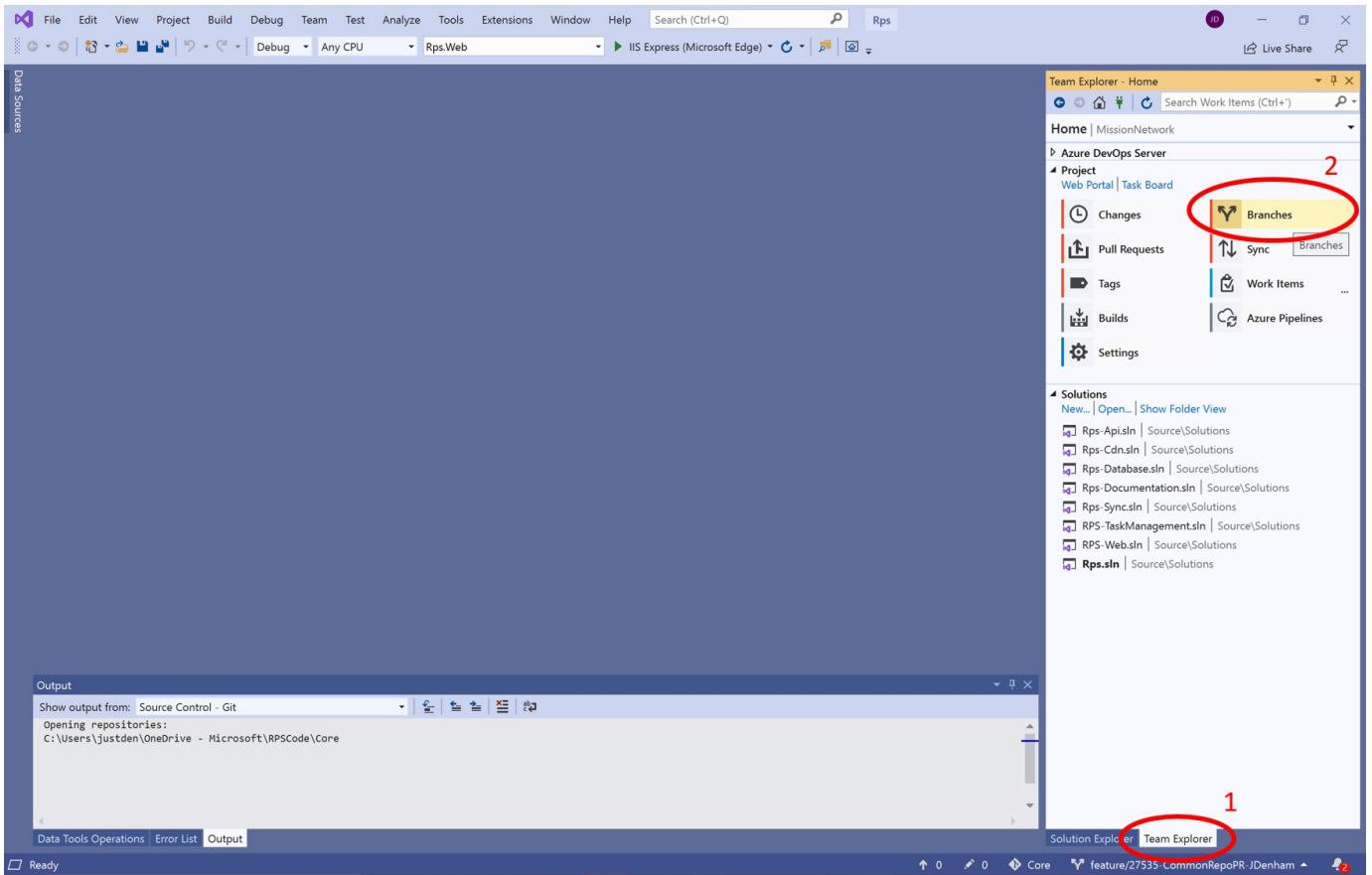
All branch names for Backlog Item-level work should:

1. Start with "feature/"
2. Followed by the TFS alias of the developer submitting the PR and a hyphen
3. Followed by the backlog item number surrounded by hyphens (see example below)
4. Followed by a short (2-3 word) description of the work
5. Output: feature/jskerrett-23456-WriteDocumentation

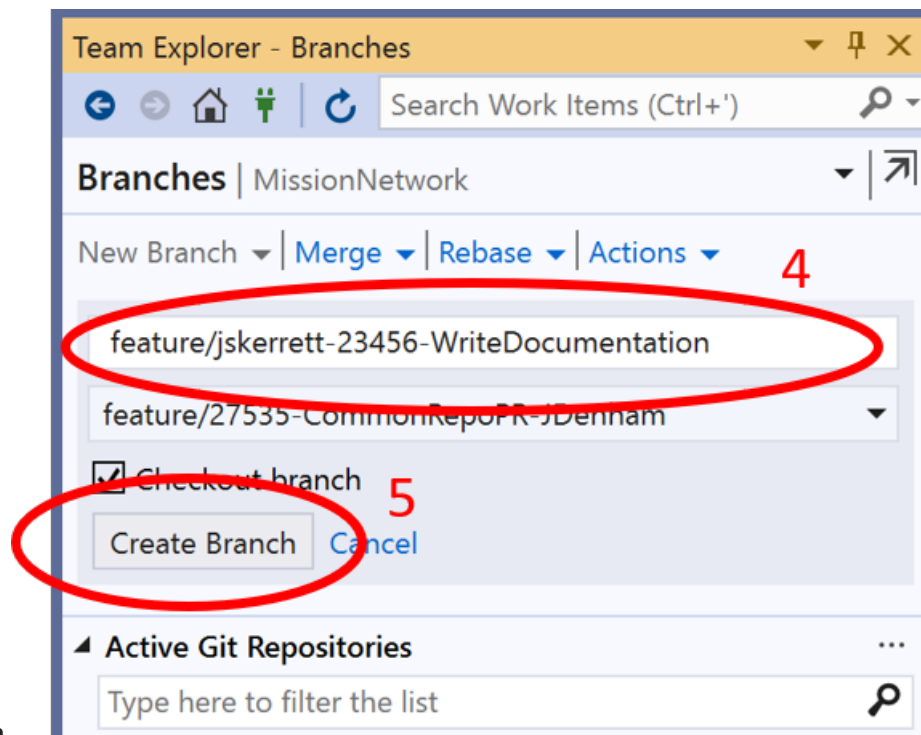
Creating the Branch

Take the following steps in Visual Studio:

1. Open Team Explorer
2. From the Team Explorer Home page, select **Branches**



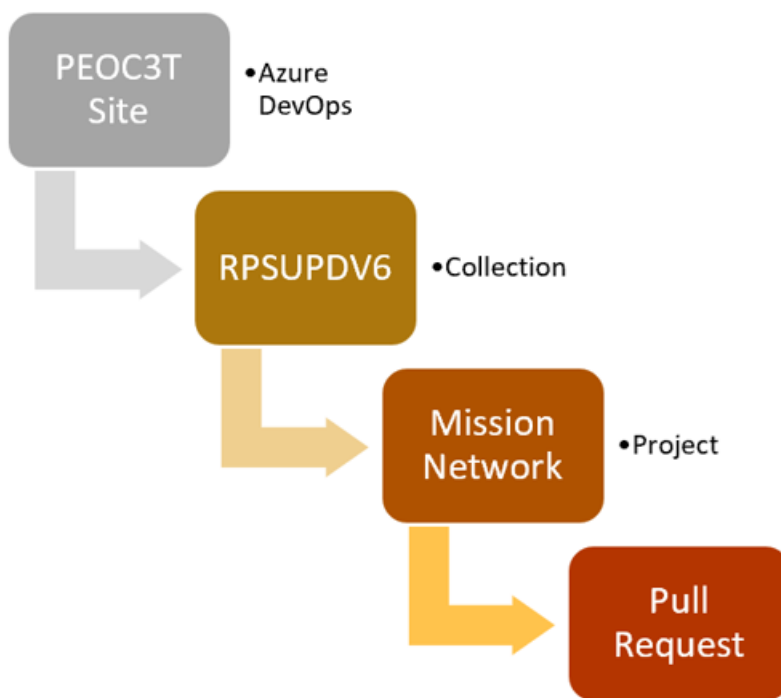
3. Click on **New Branch**
4. Type the name of the PR



5. Click on **Create Branch**

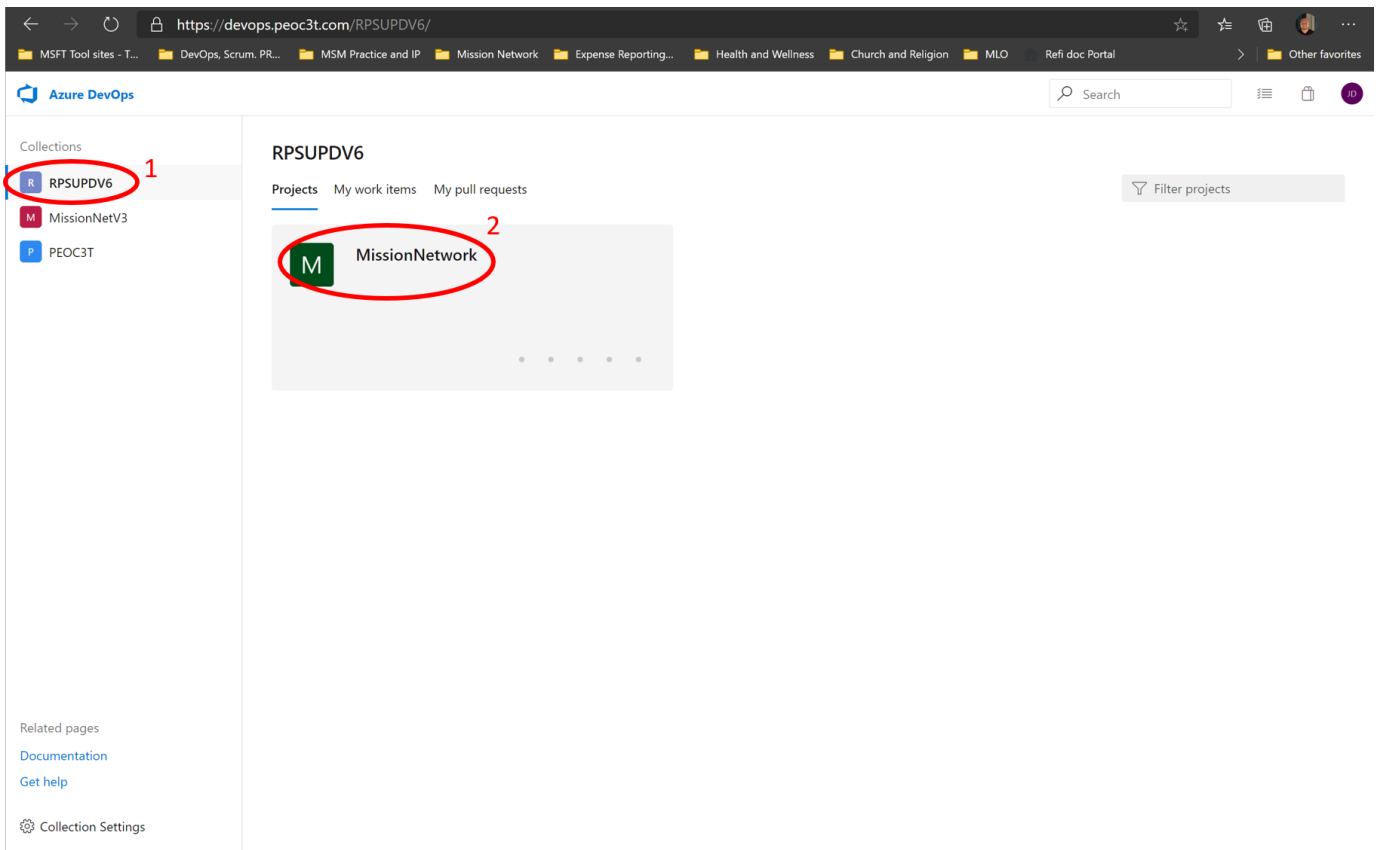
Pull Request Creation

Once your development changes are finished, a pull request should be created to begin the process of merging the proposed changes to the default ('develop') branch. The high level process begins at the PEOC3T ADO site and ends with a pull request

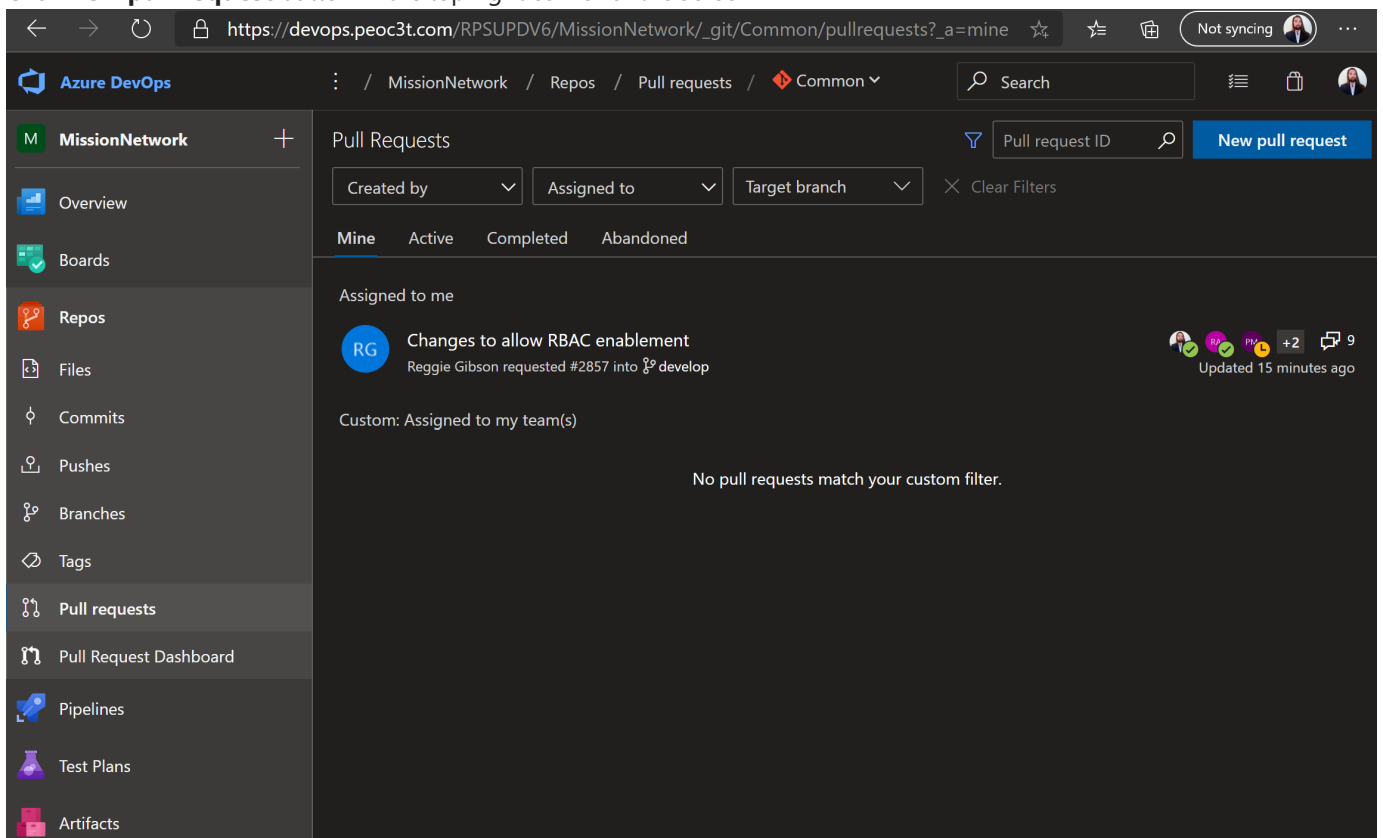


generated.

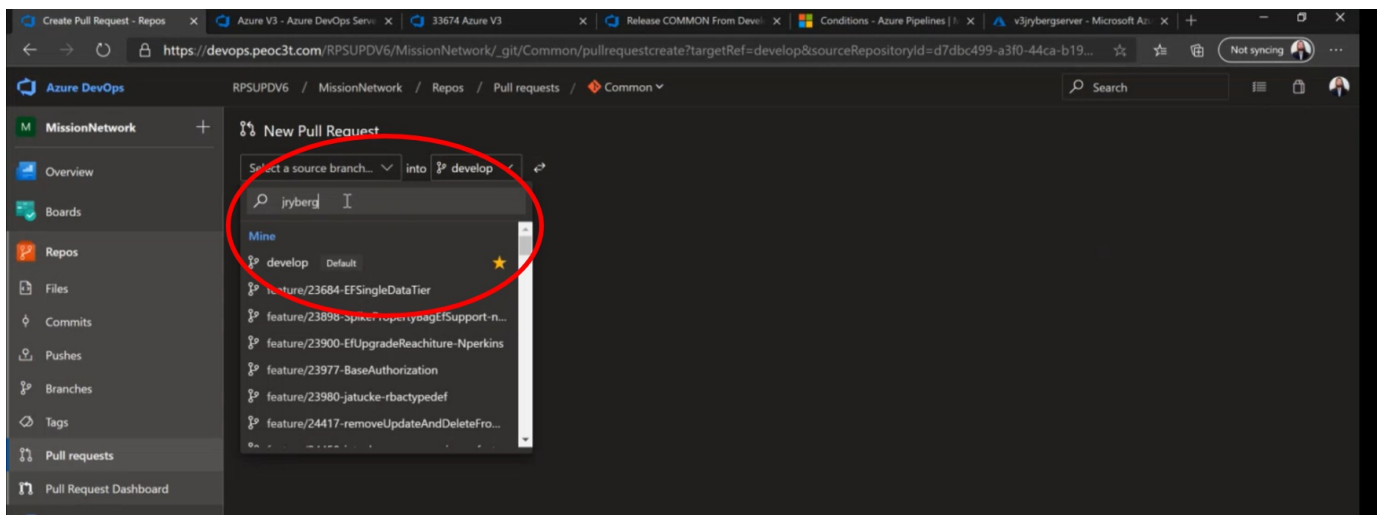
1. To begin creating a Pull requests in Azure DevOps, Navigate to <https://www.devops.peoc3t.com>



2. Click on the **Pull requests** Menu item in the left menu bar
3. Click **New pull request** button in the top right corner of the screen



4. Click the **Select a source branch...** dropdown list
5. Select the branch to be merged.



6. Fill in the required fields: a. Descriptive title b. Fill out the Description field c. Ensure a work item is linked to the PR by selecting the appropriate backlog item from the 'Work Items' dropdown list d. Click the Create button at the bottom of the **New Pull Request** Form

New Pull Request

27250-jwalker-LargePackagePerfTest into develop

Title *

add code to performance test packages

Add label

Description

The purpose of this PR is:

These changes are required because:

Items I'd like to discuss or ask for extra eyes on are:

People I'd specifically like to review this PR are:

This PR:

- Compiles
- Passes all tests
- Has a >=80% code coverage ratio for new/edited code
- Meets all defined Acceptance Criteria
- Conforms to style guidelines
- Contains applicable documentation for the customer
- Has a migration path for any breaking changes

Markdown supported.

Add commit messages Add a template

The purpose of this PR is:

These changes are required because:

Items I'd like to discuss or ask for extra eyes on are:

People I'd specifically like to review this PR are:

This PR:

- Compiles
- Passes all tests
- Has a >=80% code coverage ratio for new/edited code
- Meets all defined Acceptance Criteria
- Conforms to style guidelines
- Contains applicable documentation for the customer
- Has a migration path for any breaking changes

Reviewers

Search users and groups to add as reviewers

Work Items

Search work items by ID or title

27250 Load testing large packages

Create

7. Pull Request Has now been created

PR Approval

Approval Requirements

The following criteria must be met before a PR can be completed and merged to the default branch:

- PR must be approved by 2 required approvers
 - For COMMON repository, the required approvers are limited to the MCS 'Required Approvers' group
 - Other approvals on the PR are recommended, as more eyes on code changes not only increase confidence in the change, but also increase awareness of changes to the codebase

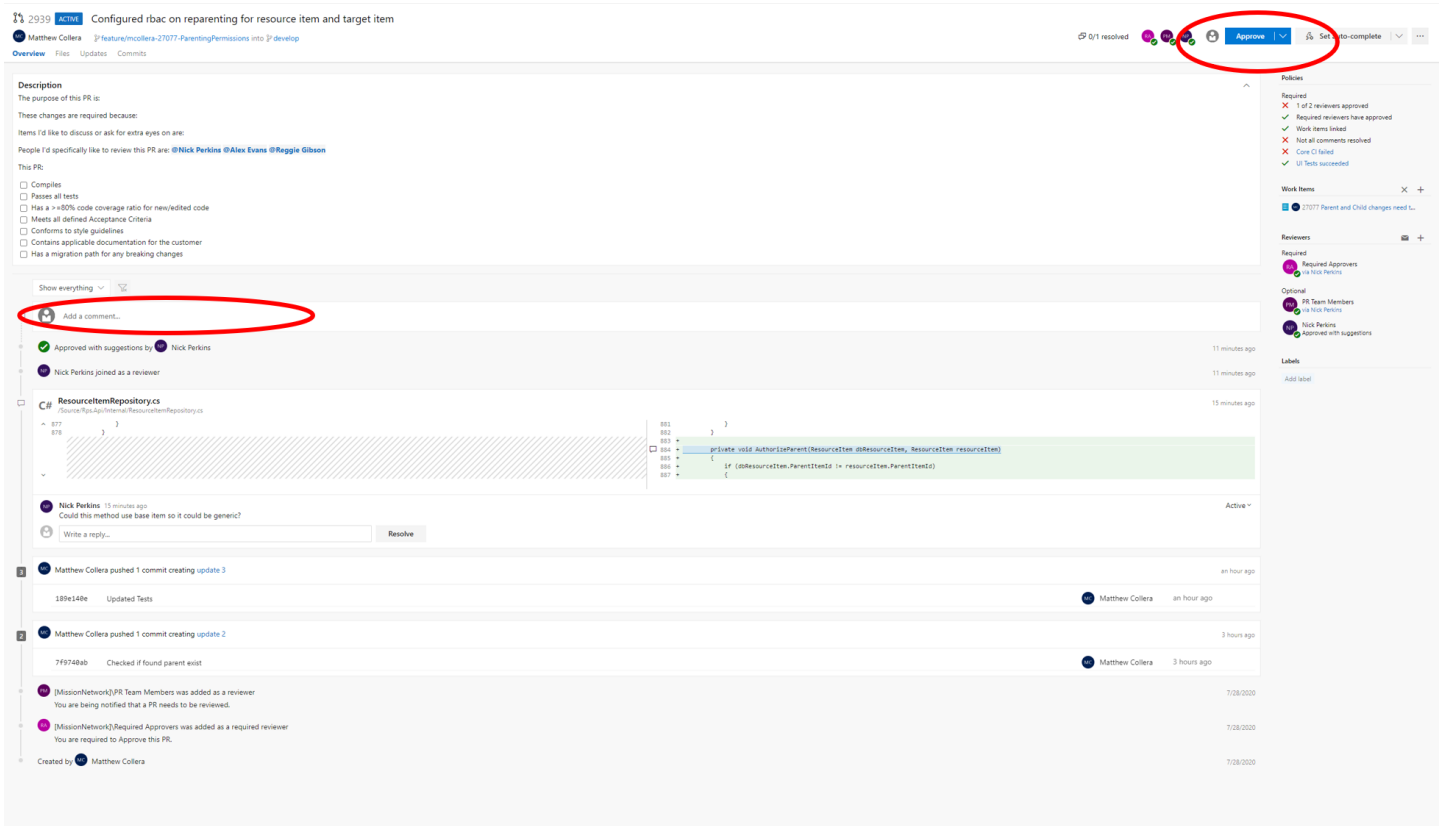
- PR cannot be approved with a blocking approver
- Reviewers cannot approve their own changes
- A work item (backlog item or bug) must be linked to the PR
 - This creates historical context for code changes
- All comments must be resolved
- CI Builds must all pass
 - PowerShell Unit tests
 - PowerShell Meta tests
 - DSC Modules tests

The screenshot shows a pull request interface with the following sections:

- Policies:** A list of requirements with status indicators:
 - Required: 1 of 2 reviewers approved (Red X)
 - Required reviewers have approved (Green checkmark)
 - Work items linked (Green checkmark)
 - Not all comments resolved (Red X)
 - Core CI failed (Red X)
 - UI Tests succeeded (Green checkmark)
- Work Items:** A section with a close (X) and add (+) button, containing one item:
 - 27077 Parent and Child changes need t...
- Reviewers:** A section with a close (X) and add (+) button, showing:
 - Required:**
 - Required Approvers via Nick Perkins (Green checkmark)
 - Optional:**
 - PR Team Members via Nick Perkins (Green checkmark)
 - Nick Perkins Approved with suggestions (Green checkmark)
- Labels:** A section with an "Add label" button.

Currently, all GD users will have access to view and approve PRs into the COMMON repository.

Note: If the PR has passed the Policy requirement, a green check mark will show beside it. If the PR has not passed the policy requirement, then a red X will show beside it.



Completing the PR - Definition of Done

For a Work Item (at the Product Backlog Item or Bug level) to be considered done, the following must be addressed as part of the Pull Request:

- The latest updates from 'develop' are merged into the working "feature" branch prior to PR creation
- Altered code meets Style Guidelines
- Unit and/or Integration Test(s) implemented and/or updated
- All tests are run locally and pass
- For non-trivial updates, a lab is deployed locally to ensure there are no breaking changes. RPS functionality is verified.
- All Work Item Acceptance Criteria met or discussed in work item (and tagged Dev Lead/Product Owner)
- (Runbooks) Test Runbook within TMS from either TaskMap or Schedule
- (Runbooks) Has standardized input (TaskAssignmentId), Target loading
- DSC Resource updates are documented and communicated
- Applicable documentation updated/created
- OSS Registrations created for any new Open Source software/version
 - If new / updated 3rd Party / OSS, update the file at \$/Documents/ThirdPartyNotices.txt
- Update Release Notes, as appropriate, with each PR
- Product Backlog Item/Bug moved to Done state when PR is Approved and Completed

RPS Software Design

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

The Rapid Provisioning System (RPS) is designed to provide a mobile, modular, and extensible framework that allows coded automation activities to be executed across a wide area network environment with limited connectivity and bandwidth. This solution was designed to survive the inherent constraints and challenges that have been observed within a mobile tactical network. RPS is designed to be a flexible solution for programs with similar constraints and automation requirements. This document will outline the technology, approach, and functionality provided by this solution.

Basic RPS Definitions

This section provides the definition for terms that are used throughout this document. Refer to this section for how these terms are being used within the RPS system.

TERM	DEFINITION
Cmdlet	A lightweight command that is used in the Windows PowerShell environment
Configuration Item (CI)	A vehicle that contains entity definitions
DSC	Microsoft's PowerShell Desired State Configuration software package
RPS Node	A running instance of the RPS system

Constraints and Challenges of a Tactical Environment

There are many constraints and challenges inherent to a mobile tactical network:

- Network capability is limited to modern satellite and line-of-sight (LOS).
- Vehicles don't always have network connectivity for extended periods of time. This period can range from a few days to several weeks.
- When a satellite connection is available, vehicles experience extremely high latency combined with extremely low throughput.
- Line-of-sight (LOS) provides an improvement in the network capabilities of the vehicle but has limited accessibility when vehicles are on the move (OTM).
- Immediate response capabilities for combat situations:
 - RPS is designed as a combat-ready system to support the warfighter in the field.
 - Service downtime must be schedulable and interruptible.
- Long lead times to field changes:
 - Time for changes to hardware and software takes years to progress from adoption to operating in a fielded solution.
 - Fielded solutions often have a long lifespan due to costs associated with high cost to upgrade and difficulty implementing changes to the field.
 - The RPS solution will be designed to use the latest commercial off the shelf (COTS) solutions to provide for the longest possible lifespans.

Design Goals

The RPS toolset is designed to provide the following functionality to meet the requirements of the tactical Environment:

- Targeted automation across various device types (routers, switches, radios, servers, etc.).
- Codependency on automation tasks (To accommodate cross-device requirements and prerequisites).
- A componentized automation model to simplify development of automation tasks.
- An extensible toolset to meet automation needs to accommodate environmental variances.
- Offline automation capabilities, specifically the ability to discretely operate with limited or no connectivity to upper-tier systems.
- Low-compute capabilities for executing on low resource systems.

Architectural Overview

The RPS system is designed to provide the infrastructure upon which various PORs can extend to automate tasks specific to their missions and deployments.

RPS Functionality

The common set of functionalities provided by RPS is shown below.

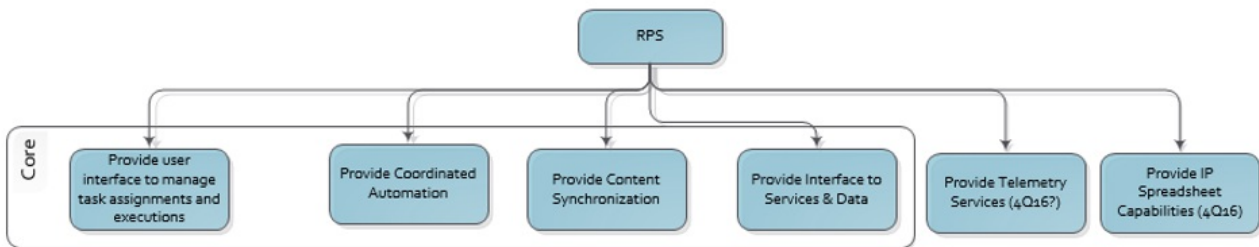


Figure 1 - Functional Hierarchy

(source: Document > Architecture > SW Design Diagrams.vsd > 'Overall Func Arch' sheet)

1. **User interface to manage task assignments and executions** – Provides the ability for a user to manage the provisioning process at a granular level. The user can start, halt, and return feedback on provisioning processes for a specific target item.
2. **Coordinated Automation** – Provides the ability to automate and coordinate a variety of tasks.
3. **Content Synchronization** – Provides the ability to synchronize content across RPS nodes to support the tasks being automated.
4. **Interface to Services & Data** – Provides a standard interface to access RPS services through a well-documented API.
5. **Telemetry Services** – Provides the ability to collect heuristic information from a system.
6. **IP Spreadsheet Capabilities** – Provides the ability to read in the standard TNACC/TNIC provided IP spreadsheet and deliver expected results.

RPS Components

COMPONENT	CORE TECHNOLOGY	DESCRIPTION	FUNCTIONALITY MAPPING
Application Server	IIS	Provides the capability to host a web application.	UI
Configuration Management	DSC	Underlying configuration management framework to specify, setup and maintain a desired machine configuration.	Coordinated Automation Content Synchronization Interface to Services and Data UI
Content Delivery	BITS, DFSR	Underlying content delivery framework to replicate and deliver content across the network.	Content Synchronization

COMPONENT	CORE TECHNOLOGY	DESCRIPTION	FUNCTIONALITY MAPPING
Core Modules	C#, PowerShell	RPS specific implementation to provide a generic capability to perform task automation. This includes all the RPS runbooks, modules and capabilities that are exposed via the RPS API.	Coordinated Automation, Content Synchronization, Interface to Services and Data, Telemetry Services, IP Spreadsheet Capabilities
Data Persistence	PostgreSQL Server	Underlying data storage that maintains the metadata needed by the system to manage the task automation. Commonly referred to as RPS CMDB.	Coordinated Automation
Messaging Service	C#	Underlying messaging framework to provide command and control communication capability between RPS nodes.	Content Synchronization
RPS Web Application	.ASP.NET MVC, IIS, jQuery, Knockout, C#	RPS specific user interface to allow a user to initiate automated tasks on target systems.	UI
Task Management Service (TMS)	Windows Service / RPS Phyr	Automated Task Management service used to process RPS Task Assignments.	Coordinated Automation

NOTE

Refer to each component's individual documentation for more details on the internals of each component.

Data Persistence (CMDB)

The detailed software design documentation for this component is available at:

RPS > Documents > Operations > **RPS Data Persistence (CMDB) Design.docx**

Automation Framework

The detailed software design documentation for this component is available at:

RPS > Documents > Operations > **RPS Automations Package Guidelines.docx**

Configuration Management

The detailed software design documentation for this component is available at:

RPS > Documents > Operations > **RPS Configuration Management (DSC) Design.docx**

Content Delivery

The detailed software design documentation for this component is available at:

RPS > Documents > Operations > **RPS Content Management.docx**

Messaging Service

The detailed software design documentation for this component is available at:

RPS > Documents > Development > **RPS Sync Services.docx**

Application Server

The application server provides the ability to host a web application. RPS uses Microsoft's IIS web server to provide this capability.

RPS Web Application

The detailed software design documentation for this component is available at:

RPS > Documents > Software > **TODO**

Software Development Kit (SDK)

The RPS SDK provides the developer-oriented documentation for the RPS system. It will include:

- RPS binary code
- Sample PowerShell source code runbooks
- RPS > Documents > Samples
- API and cmdlet documentation
- RPS > Documents > Development > **RPS API Documentation.PDF**

RPS Server Architecture

The RPS server architecture is intended to provide scalability for most automation requirements. To accommodate this, each RPS node will be able to either operate on its own or receive instructions from another RPS node. Those instructions will provide operations information to the RPS solution to request executions of automation code, or to manage its own environment.

Software Requirements

The RPS infrastructure requires the following software:

- A Microsoft Windows operating system (Windows Server 2012 R2 or greater)
 - The core operating system version the solution will run on.
- RPS API
 - Provides the .NET and PowerShell modules necessary to access, manage and maintain a RPS Node.
- PostgreSQL database
 - Provides the underlying PostgreSQL database support for the custom Configuration Management Database (CMDB)
- Task Management Service (TMS)
 - RPS Task Management
- .NET Framework 5
 - Provides enhanced PowerShell frameworks and support for advanced automation.
- Windows Management Framework (WMF) 5.1
 - Extends PowerShell support to enhance Desired State Configuration functionality.
- Microsoft Distributed File System Replication services (DFSR).
- Background Intelligent Transfer Service (BITS)
- Software for maintaining content distribution over slow, unstable WAN links.
- RPS Sync Service
 - Provides Windows service to synchronize RPS nodes.

Server Role Minimum Requirements

ROLE	CPU CORES	HARD DISK (GB)	RAM (GB)
CDN	2	100 (Depends on content)	4
GUI	2	50	4

ROLE	CPU CORES	HARD DISK (GB)	RAM (GB)
PostgreSQL	2	10	2

Server Architecture

It is important to note that the server layout of the RPS components is designed to be fluid based on requirements. This section will outline the software components of the RPS architecture and how they interact, rather than the server architecture of a specific implementation.

The software components of a given node can be placed on any server layout provided network communications and software prerequisites are available. Each collection of RPS components is identified as a "RPS Node". These nodes are intended to operate both individually, and in a distributed workload capacity. Node layouts will vary based upon the requirements of a given implementation. For example:

- Single server RPS node (all software on one system)
- Multi-server RPS node (each component on its own instance of Windows across multiple servers)

RPS API - PowerShell Module and Data Access

The RPS API is a single library to be used in conjunction with the RPS CMDB. The API provides .NET and PowerShell interfaces for managing and manipulating RPS data and actions.

PowerShell and API Functionality

Table 2 outlines the available actions the RPS API and PowerShell module offers to support and maintain the RPS environment.

HIGH LEVEL ACTIONS	PURPOSE
Managing RPS Infrastructure Data	Actions such as registering and managing RPS nodes and their relationships to each other.
Managing Resource Assignments	Manage Assignments between Resource Items and Target Items.
Managing Resource Items	Manage resources, their properties, and metadata.
Managing Target Items	Manage Target Items, properties, their relationship to other items, and other item related metadata.
Managing Tasks	Provides the commands necessary to manage and register tasks.
Managing Task Assignments	Manage Assignments between Tasks/Task Maps and Target Items.
Managing Task Maps	Manage Task Maps and their definitions, what tasks to run, what their dependencies are, filtering criteria, etc.
Misc. Toolset Functionality	Actions such as resets, initiating Task Map evaluations, setting requests for user-interaction, or other functions that may not necessarily directly relate to CMDB data.

More Resources

- [RPS Data Persistence \(CMDB\) Software Design](#)
- [RPS Configuration Management \(DSC\) Software Design](#)

RPS Configuration Management (DSC) Software Design

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

This guide shows the underlying RPS framework to specify, setup and maintain a desired machine configuration for the Windows computers, including RPS itself, through the RPS API and Desired State Configuration (DSC).

Configuration Management using DSC

PowerShell Desired State Configuration (DSC) is a built-in capability of most Windows devices used to manage discrete configuration of a device or system. Additionally, it has direct support for Linux systems and non-OS devices by proxy of a Windows Device (e.g. Routers).

RPS facilitates a process in which DSC configurations can be correlated to devices. This allows for the management of configurable devices in a consistent manner.

This section will outline the suggested structure for authoring DSC configurations.

TERM	DEFINITION
RPS Node	A running instance of the RPS system.
DSC	Microsoft PowerShell Desired State Configuration.

DSC Partial Configurations

DSC allows for configurations to be structured as Partial Configurations which represent small configurations for individual pieces of configuration data. By authoring DSC configurations for delivery in RPS into smaller Partial configurations, they can be pieced together and rearranged in new ways for delivery in multiple configuration sets. This modular approach greatly improves the reusability of individual configurations.

The following link provides additional information on DSC Partial Configurations: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/pull-server/partialconfigs?view=powershell-7.1>

DSC Resources

If a DSC partial configuration contains the information of what the configuration will be, a DSC resource contains the information of how the configuration is set. RPS uses DSC resources from the open-source community and the RPS team itself.

NOTE

The RPS documentation includes information on authoring or adding new DSC resources to the solution. This documentation can be found at [\\$/Documents/Operations/Authoring DSC Resources.docx](#).

The RPS DSC Structure

The RPS API structure used by DSC is shown below in a distilled format. This structure allows us to develop generic DSC partial configurations that can be combined in many ways using resource groups and applied to any number of target items.

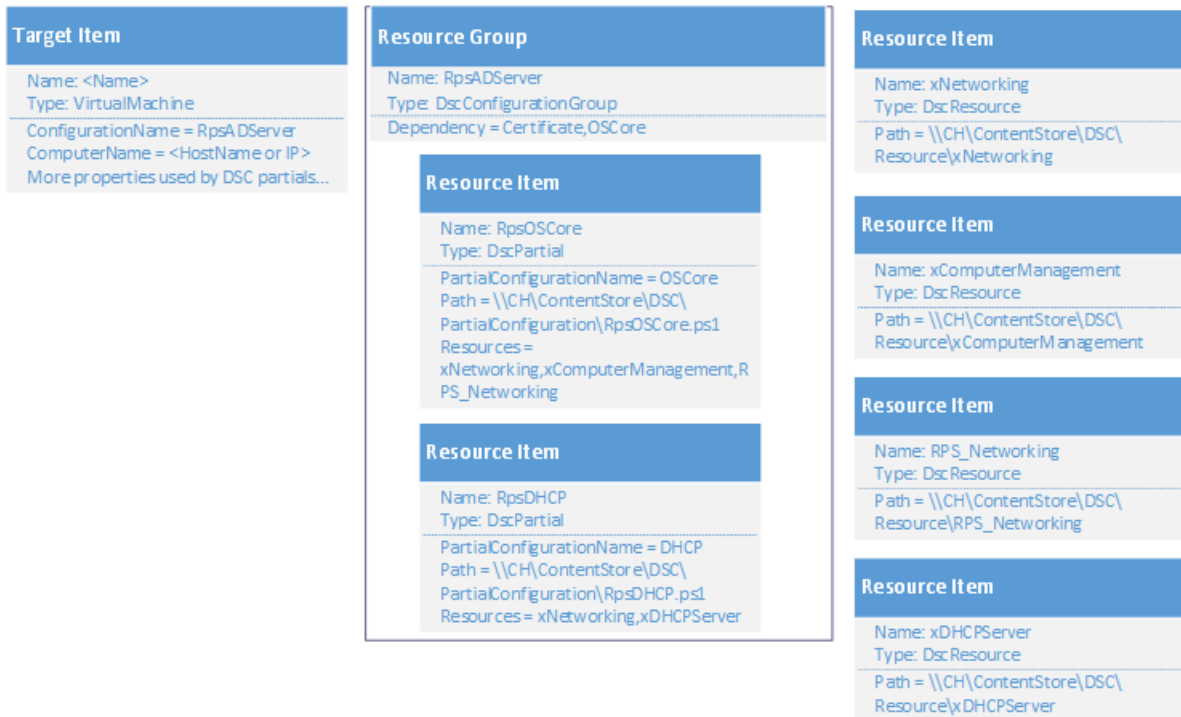


Figure 1: Example RPS items for DSC use.

Each target item configured by DSC is required to be of **type** VirtualMachine. The property ConfigurationName is used to associate the target item with a resource group by the same name. Resource items within the group are the DSC partial configurations applied to the target item when Start-Dsc or Start-DscOffline is called. The property ComputerName is used by the DSC push process described below to connect to the target machine.

The resource group of type DscConfigurationGroup is used to group multiple partial configurations into groups for application to target items. This group type has a single optional property named Dependency. This is a semi-colon separated set of comma separated pairs naming the dependencies between DSC partial configurations.

For example, if the dependency property is "Certificate;OSCore;SQL;OSCore" a dependency for both the Certificate and SQL partial would be set on the OSCore partial.

The resource items of **type** DscPartial represent individual DSC partial configurations. Two of the properties, PartialConfigurationName and Path, are required with Resources being optional. Any resources used by the DSC partial listed as a comma separated list within the Resources property will be copied to the target item computer by the Start-Dsc Runbook or the Start-DscOffline script.

Finally, the resource item of **type** DscResource represents the DSC resources used by the partial configurations. Only the Path property is required.

Nearly all the DSC partial configurations have specific required parameters. Each of these expected parameters should be a property on all the target items using the partial configuration.

Start-Dsc and Start-DscOffline will use the properties on the target item as the parameters for call the DSC partial configurations. If the target item is missing a required property, the scripts will check higher memory scopes (This allows for temporary development environments with modified settings). If no property or variable is found matching all required parameters an exception will be thrown.

Compiling and Pushing Partial Configurations

When utilizing the DSC solution described in this document, there is a difference between how these DSC configurations can be used in an offline (RPS is not yet installed) scenario versus an "Online" (RPS is present) scenario.

The reason for this is, the same process used by RPS to push and configure DSC routines is the same process used to install RPS.

By building the system this way it creates a circular flow that is self-supporting. This section will describe the differences between the online and offline implementations of RPS pushing DSC configurations. Simply put, the differences in functionality reside simply in where configuration data is sourced from.

Online

The "Online" version of pushing DSC with RPS consists of a series of Powershell runbooks (workflows) authored as Task Items in RPS. These runbooks are:

- Start-Dsc
- Test-Dsc

When Start-DSC is assigned to a TargetItem through the normal processes of RPS, the configurations are pulled and compiled as described previously in this document . By creating those groups of partials, the system can pair up TargetItem property data with required parameters from the database directly and execute the DSC push. The location in which the push is targeted is defined by the TargetItem referenced.

Currently, each targeted item used in this process must have a ComputerName property with appropriate data to supply connectivity information (such as an IP address, or host name).

In this mode, the data is sourced straight from the RPS Node CMDB database that is pushing the Start-DSC Task as an executed item.

The Test-Dsc Runbook is used to provide information on the current state of the DSC configuration on the target computer.

Offline

The "Offline" version of pushing DSC with RPS consists of a series of scripts that execute in the same manner described in the Online mode, however they source their data from local files. The scripts in this use case are:

- Start-DSCOffline
- Invoke-RpsInitialization

The XML files supplied to this process are serialized using the built in RPS Commands for serialization. These XML files are created using another existing RPS installation or using the Rps-APIMock module.

A series of XML files must be exported to support an Offline DSC operation. They must include all required data by the RPS DSC Process including:

- ResourceItem data
- ResourceGroup data
- TargetItem data

By exporting this data into XML files, Start-DSCOffline can parse them and utilize the information to execute the same way the Online use case is executed.

Invoke-RpsInitialization provides a wrapper mechanism to ensure proper files, supporting content, and metadata is present at appropriate locations before DSC Scripts are applied and begin executing the configuration process.

Building the processes this way enables RPS to self-replicate new copies to new servers, which can also maintain their own health, self-recovery and report the state of configuration.

The offline initialization process supports XML files encrypted with the Rps-Encryption module as well as plain text files.

More Resources

- [RPS Software Design](#)
- [RPS Data Persistence \(CMDB\) Software Design](#)

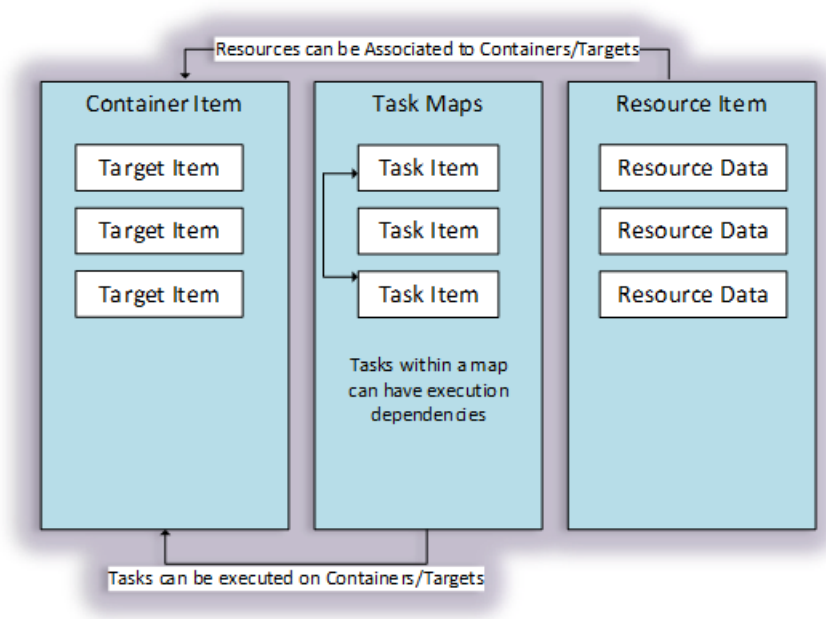
RPS Data Persistence (CMDB) Software Design

Last updated on March 24, 2021.

Last Reviewed and Approved on PENDING REVIEW

The data persistence feature of the Rapid Provisioning System (RPS) provides the underlying data storage needs of the system. The RPS Configuration Management Database (CMDB) maintains the metadata that is needed by RPS to targetable containers of items, the resources to be applied to those items, tasks and task maps for orchestrating task execution on those items, and the node hierarchy in which these items reside.

The figure below shows a high-level relationship of how these meta-data types interact for the purposes of automations.



RPS CMDB – ER Database Diagram

The detailed entity relationship diagram documentation for the RPS CMDB is available at: [RPS > Trunk > Documents > Architecture > RPSEntityRelationshipDiagram.vsd](#)

NOTE

This document is intended to outline the physical database layout.

TERM	DEFINITION
Automation	A single automated task that is executed by the RPS system.
CMDB	Configuration Management Database (custom for RPS).
DSC	Microsoft PowerShell Desired State Configuration software package.
Device	Any computer, router or other networked device which can be targeted by an automated risk.
Node	A running instance of the RPS system.
RPS	Rapid Provisioning System.

TERM	DEFINITION
Resource	Any item of resource that can be applied to a target through a running task.
Target	Any container or item on a container that can be directly targeted for task execution by RPS.
Task	Any workflow/runbook that can be executed on a target by RPS.
TMS	Task Management Service - hosts & runs runbooks, used by RPS.

Metadata Storage – RPS Tables

This section will describe the tables needed to persist data for the RPS API. This document will further describe how each table is related and leveraged by the RPS API.

RPS coordinates automations by leveraging the CMDB for target, task, resource, and node configuration metadata. This data represents the containers/items to target, resources to assign, tasks to assign, orchestrate & automate, and hierarchal nodes to locate and synchronize. This data is all stored in a very generic format to offer maximum flexibility and extensibility for any user of the solution. Breaking down data into a generic data model allows for any type of target, resource, and task information to be stored and utilized within the RPS Database for coordinated automation and synchronization. The table below briefly describes the current tables represented within the RPS CMDB:

DATA TYPE	FUNCTION
Container	A Container is metadata about a TargetItem that is a top-level parent of a set of target Items. Task maps are only assigned to target items that are containers.
ItemProperty	An ItemProperty is a key/value property bag item that contains all custom metadata for an RPS item allowing for extensibility. The ItemProperty table is used by TargetItem, TargetGroup, ResourceItem, ResourceGroup, Node, TaskItem, TaskAssignment, and ResourceAssignmentStatus.
LocalConfig	A LocalConfig is metadata about a local RPS instance. An example is the identification of the local RPS node.
Node	A Node is metadata about an identifiable location that is a running instance of RPS. Nodes can have parent/child relationships. Changed data is synchronized between related nodes.
ResourceAssignmentStatus	A ResourceAssignmentStatus is metadata about a resource assignment to a targetable item and its current state. Resource assignments are defined and managed by individual tasks (workflows).
ResourceGroup	A ResourceGroup is metadata about a logical grouping of resource items. A ResourceGroup has a set of ResourceItem members to simplify resource management and assignments.
ResourceItem	A ResourceItem is metadata about a resource item that is not an executable task. A resource item is either a local or global item of resource that is applied to a targetable item by a task workflow/runbook. An example of a resource item is an operating system patch or a configuration template.
TargetGroup	A TargetGroup is metadata about a logical grouping of target items. A TargetGroup has a set of TargetItem members to simplify task/resource assignments.
TargetItem	A TargetItem is metadata about a device item or container. Target items can have parent/child relationships. A top-level target item is called a 'Container' since it always contains a set of targetable items. Target items can have tasks assigned to be executed on them via automation. Target items can have resources assigned to be applied to them. An example of a target item is a router or a switch.

DATA TYPE	FUNCTION
TaskAssignment	A TaskAssignment is metadata about a task assignment to a targetable item and its current state. Task assignments are sometimes direct assignments of a task to a target item. Task assignments are also generated through assignments of a task map to a container. TMS uses task assignment state to control automation.
TaskAssignmentDependency	A TaskAssignmentDependency is metadata about a task assignment dependency. Each instance defines the parent/child dependency between two task assignments. These dependencies are generated from task map definition dependencies during a task map assignment to a target. These dependencies are used by TMS to control task execution order during automation.
TaskAssignmentUserAction	A TaskAssignmentUserAction is metadata about a user action associated with a task assignment when task assignment state is 'PendingUserAction'.
TaskItem	A Task is metadata about a task workflow. A task is a workflow that is executed by TMS. Tasks are assigned to target items directly or through task maps for coordinating task execution on target items.
TaskMap	Identifies a set of steps, what order they should be performed in, and what target items those steps apply to.
TaskMapAssignment	A TaskMapAssignment is metadata about a task map assignment to a target. A task map assignment contains information about the assignment of a task map to a target item or group of target items.
TaskMapDefinition	A TaskMapDefinition is metadata about a task map definition. A task map definition defines which task item is assigned to which target item. An optional property bag filter can be associated.
TaskMapDefDependency	A TaskMapDefDependency is metadata about a task map definition dependency. Each instance defines the parent/child dependency between two task map definitions. These dependencies are used by TMS to control task execution order during automation.
TaskMapDefFilter	A TaskMapDefFilter is metadata about a task map definition filter. A filter is used to further define which target items a task item is being assigned to by specifying which property bag key/value pairs are present on a target item.
TaskState	A TaskState is an enumeration of task states used for task assignments by the TMS controller.

RPS CMDB Relationships

The detailed design diagram for database entity relationships (ER) on how database entities are related to each other can be found here: [RPS > Trunk > Documents > Architecture > RPSEntityRelationshipDiagram.vsd](#)

The CMDB database tables have the following cardinality in relation to other tables:

- A **Node** can have zero-or-more child Nodes, and zero-or-one parent Node.
- A **Node** can contain zero-or-more TargetItems.
- A **Node** can contain zero-or-more local ResourceItems.
- A **Node** can have zero-or-more ItemProperties.
- A **LocalConfig** can reference one local Node.
- A **TargetItem** can have zero-or-more child TargetItems, and zero-to-one parent TargetItems.
- A **TargetItem** can have zero-or-more ItemProperties.
- A **TargetItem** can belong to one Node.
- A **TargetItem** can belong to zero-or-more TargetGroups.
- A **TargetItem** can belong to zero-or-more ResourceAssignments.
- A **TargetItem** can belong to zero-or-more TaskAssignments.

- A **TargetGroup** can have zero-or-more TargetItems.
- A **TargetGroup** can have zero-or-more ItemProperties.
- A **ResourceItem** can have zero-or-more child ResourceItems, and zero-or-one parent ResourceItem.
- A **ResourceItem** can belong to one local Node or be global.
- A **ResourceItem** can have zero-or-more ItemProperties.
- A **ResourceItem** can belong to zero-or-more ResourceGroups.
- A **ResourceItem** can belong to zero-or-more ResourceAssignments.
- A **ResourceGroup** can have zero-or-more ResourceItems.
- A **ResourceGroup** can have zero-or-more ItemProperties.
- A **ResourceAssignmentStatus** can reference one ResourceItem.
- A **ResourceAssignmentStatus** can reference one TargetItem.
- A **ResourceAssignmentStatus** can have zero-or-more ItemProperties.
- A **TaskItem** can belong to zero-or-more TaskMapDefinitions.
- A **TaskItem** can belong to zero-or-more TaskAssignments.
- A **TaskItem** can have zero-or-more ItemProperties.
- A **TaskMap** can have zero-or-more TaskMapDefinitions.
- A **TaskMap** can have zero-or-more TaskMapAssignments.
- A **TaskMapDefinition** can belong to one TaskMap.
- A **TaskMapDefinition** can be associated with one TaskItem.
- A **TaskMapDefinition** can have zero-or-more dependent child TaskMapDefinitions, and zero-or-more dependent parent TaskMapDefinitions.
- A **TaskMapDefinition** can have zero-or-more TaskMapDefFilters.
- A **TaskMapDefFilter** can be associated with one TaskMapDefinition.
- A **TaskMapAssignment** can reference one TaskMap.
- A **TaskMapAssignment** can have zero-or-more TaskAssignments.
- A **TaskAssignment** can reference one TaskMapAssignment.
- A **TaskAssignment** can reference one TaskItem.
- A **TaskAssignment** can reference one TargetItem.
- A **TaskAssignment** can have zero-or-more dependent child TaskAssignments, and zero-or-more dependent parent TaskAssignments.
- A **TaskAssignment** can have zero-or-more ItemProperties.
- A **TaskAssignmentUserAction** can reference one TaskAssignmentStatus.

RPS CMDB Detail Descriptions

Nodes

Nodes provide a physical location for a set of containers, target items and local resource items that is globally identifiable by the synchronization of data between nodes. Custom node properties are stored in an item property bag.

For example, the defined hierarchy of nodes are assigned containers, then also task and resource assignments. A parent node can automatically provide a child node through synchronization data describing a patch or other release, and the controller can begin to execute the release as soon as the data is available.

Containers – Automations Scope

Containers provide a scoping mechanism to properly 'group' items together for automations purposes. It is defined as the top-most parent in a TargetItem's hierarchy. When a parent/child hierarchy of TargetItems is established, the highest-level item is automatically derived as the "Container". Custom Container properties are stored in an item property bag.

When executing TaskMaps that require more complex execution chains, the container provides the scope that is representative of how to coordinate those tasks across devices. This enables the capability to define many similar sets of devices with different scopes for automation purposes.

Consider a series of systems that are interdependent, such as a set of servers on which different pieces of software reside.

Software or devices may require that certain components are configured before others. The container offers a scoping mechanism for how those dependencies relate.

Other examples include:

- A datacenter containing servers. Representing the datacenter as a TargetItem, then adding child TargetItems.
- A vehicle containing equipment. Representing the vehicle as an TargetItem, then adding computer devices as child TargetItems.

Both of these offer the scope necessary to automate their TargetItems as related to each other.

⚠ IMPORTANT

A TargetItem may only belong to one container. Containers cannot belong to other containers, as they are derived from the top-most parent item in a parent TargetItem / child TargetItem hierarchy.

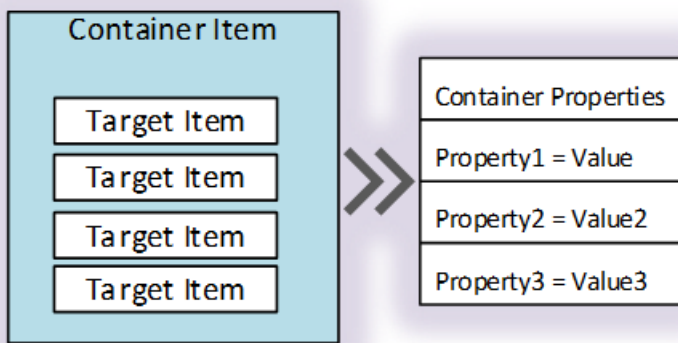


Figure 1 - Container: contains items, has properties

TargetItems – Target Device Data

TargetItems represent the data about an individual item and its properties. A target item is simply a data representation of the device that requires automation. This could be physical or logical. A target item belongs to a container for the purposes of scope and executions. Custom TargetItem properties are stored in an item property bag.

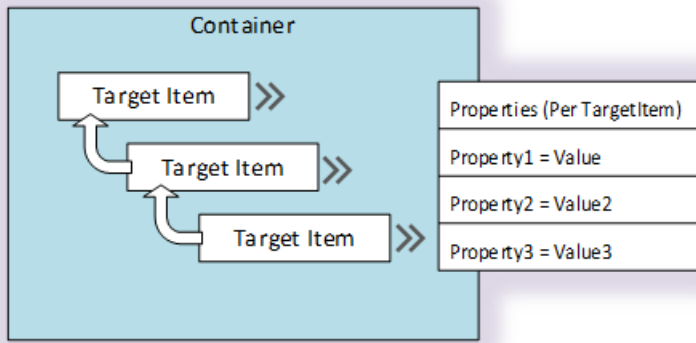
Examples include:

- A computer or server
- A router or switch
- A 'vehicle', logically vehicle may be physical, it is not a 'device' that we automate against, however there may be a model in which that is the case.

i NOTE

TargetItems also maintain a parent-child relationship to each other, leaving the highest-level parent always automatically defined as the container. This offers the ability to structure item relationships around their relationships to other items while maintaining a scope automatically. As this scoping mechanism is logical in nature, it can be defined using whatever logic best suits the business case. (E.g., a "Vehicle" containing vehicles, or a "Datacenter" containing servers, or a "Computer" containing software.) The figure below shows three sample target items, as related to each other. Each has its own separate list of properties, but all exist within the context of the Container, while the container itself is the top-most parent defined in the hierarchy of items.

Figure 2 - Target items with Properties, as related to each other and their Container.



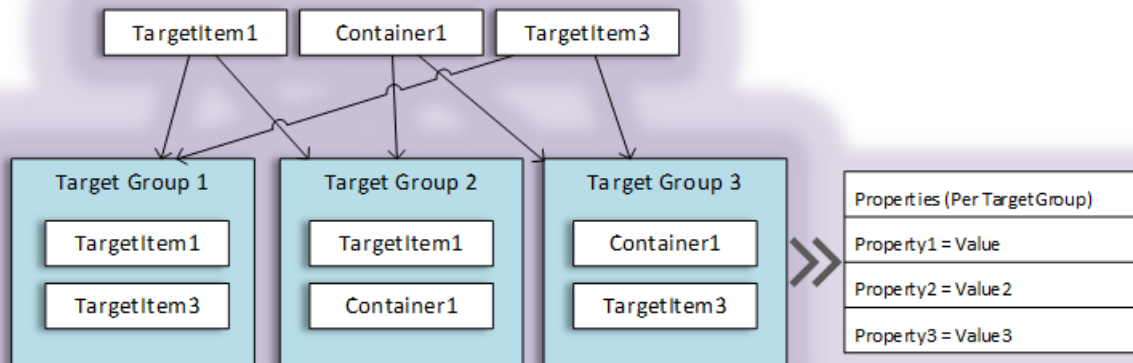
TargetGroups

TargetGroups offer the ability to relate TargetItems into a bucket for administrative purposes. TargetGroups are used to group together all items of a certain type, or all items based on a specific criterion. This provides administrative capabilities to assign tasks to a large number of TargetItems with whatever common criteria are required by the business case. (E.g., "All Servers" or "All Dell devices", etc.) Custom TargetGroup properties are stored in an item property bag.

⚠ IMPORTANT

That TargetGroups do NOT offer the same scoping functionality that Containers do for the purposes of coordinating Tasks. TargetGroups provide an administrative way to manage a bundle of TargetItems. Each is treated individually and iteratively whenever an action is taken against the group, such as assigning tasks.

Figure 3 - Containers and TargetItems within a TargetGroup, which also has properties.



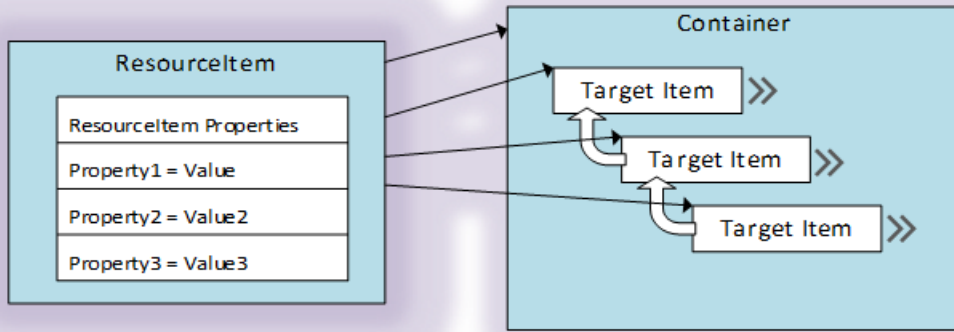
ResourceItems - Associative Resource Data

ResourceItems represent a reusable, associative piece of data for broader consumption than an individual item or container. For example, storing a list of all patches as required to be applied to Windows Servers, then associating that data to all Windows Servers in our TargetItem data. Custom ResourceItem properties are stored in an item property bag.

Resources provide this mechanism by representing the patch data as a ResourceItem (what patch, where the files are located, etc.). These details are the properties of the ResourceItem. This resource can then be associated to all of the relevant devices via a ResourceAssignment. Providing a one-to-many relationship of that single Resource to as many TargetItems as deemed applicable.

The figure below shows a simple ResourceItem and an example of the resource being 'associated' to various TargetItems or Containers.

Figure 4 - Resource item with properties, with an example of association to items/containers

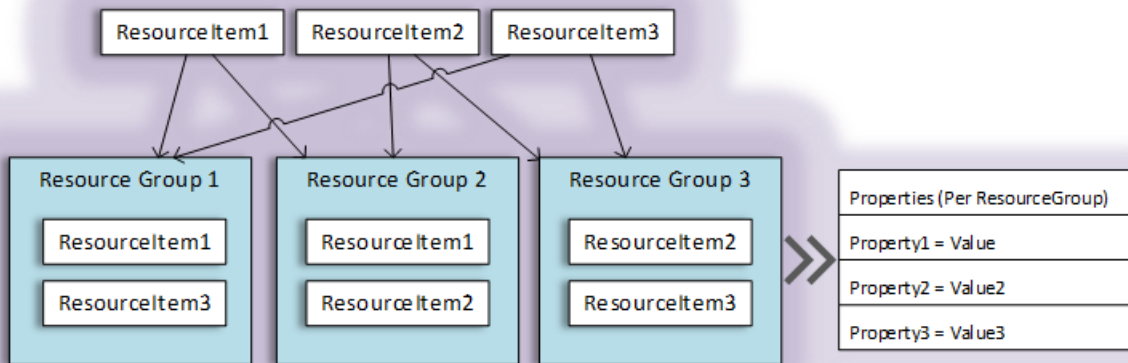


ResourceGroups

ResourceGroups offer identical functionality to TargetGroups in regard to ResourceItems. They provide a logical grouping mechanism for grouping ResourceItems together for use and consumption. As with TargetGroups, the ResourceGroup does not offer any scoping functionalities and is merely an administrative function. Custom ResourceGroup properties are stored in an item property bag.

One benefit to ResourceGroups however, given their flexibility, is that they can be used to piece together configurations from smaller sets of ResourceItems. Configurations can be stored as Resources, then added to various ResourceGroups to create different configuration sets for different purposes.

Figure 5 - ResourceGroups containing Resources, Each group also has properties



ResourceAssignment

ResourceAssignments offer a correlation mechanism between ResourceItems and TargetItems and subsequently store a status about that relationship. For example, defining a patch resource, and assigning it to a device allows for both the existence of that correlation as well as its status. Custom ResourceAssignment properties are stored in an item property bag.

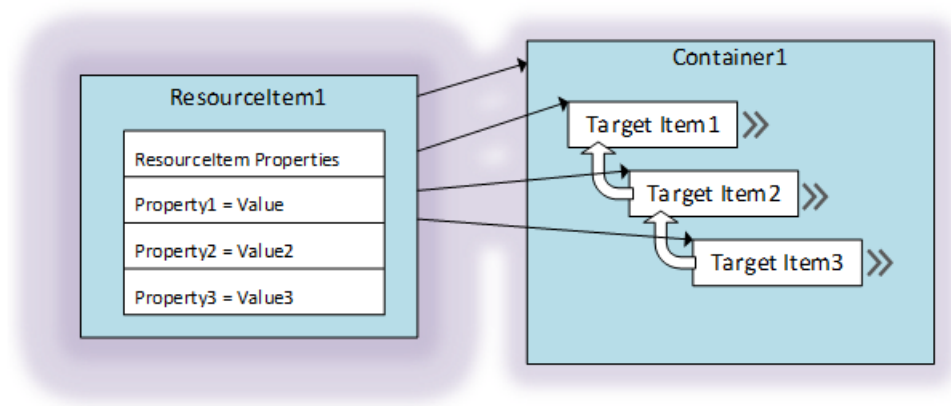
It is important to note however, the action of associating ResourceItems to TargetItems has no outward effect on automation. No execution request is performed, and no action is taken simply by the action of creating this assignment. The assignment instead creates the reference between that TargetItem so it may be leveraged in code. This provides both management of what ResourceItems are assigned to what TargetItems as well as an optional AssignmentStatus for this relationship.

Using this mechanism is optional, as there may be cases where a ResourceItem does not have or require a status of the Assignment. As a result, use of ResourceAssignments are to be leveraged as needed. While one may exist, it may contain null or useless data.

Additionally, a ResourceAssignment is not required to make use of a ResourceItem in code. Storing 'global' information in a ResourceItem without assigning it to anything is an expected use-case.

The diagram below and provides context to the resulting list of ResourceAssignments that would be generated if the displayed associations were created.

Figure 6 - ResourceItem associated to multiple Targets and a container

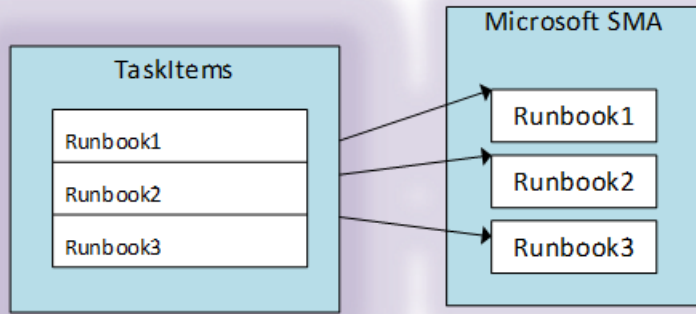


RESOURCEITEM	TARGET	STATUS
ResourceItem1	TargetItem1	Complete
ResourceItem1	TargetItem2	Incomplete
ResourceItem1	Container1	NULL
ResourceItem 1	TargetItem3	MiscStatus

TaskItems

A TaskItem is an instance of a runbook registered to the RPS database. This represents a runbook that has been imported into TMS for execution and is being added to RPS to be made available for coordinated execution by RPS. Custom TaskItem properties are stored in an item property bag.

Runbooks can exist inside of TMS without being registered to RPS for control, coordination, and execution by the automation's solution. This distinction registers an item for coordinated execution by RPS.



Task Maps - Coordinating Pieces of Automations

Automations requirements vary by need from installing software to configuring or installing firmware on devices that don't necessarily run common operating systems. This makes automation tasks in general quite difficult to author due to the complexity of dependencies that exist when analyzing any specific goal. Coding these dependencies in begins to create a complex web of code that can be tedious to manage as well as a difficult to update or change.

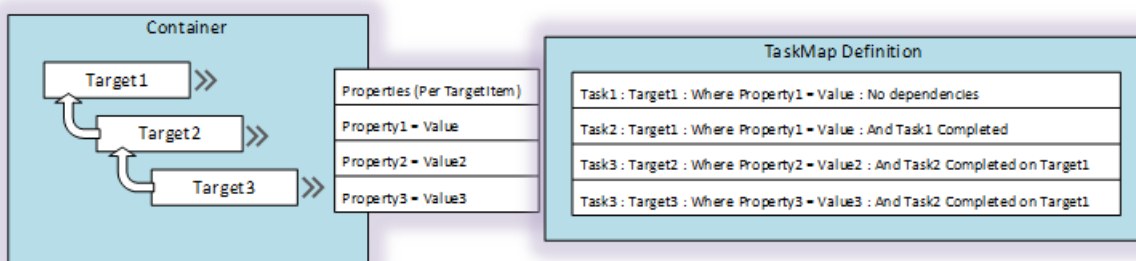
By coordinating componentized bits of code, each task can be isolated, re-used where able, and coordinated in the appropriate order or executed individually if needed. This removes the concept of authoring automations into monolithic scripts by coordinating the smaller pieces.

Each automation action's state is represented by an assignment record (see TaskAssignments) containing information about the TaskItem and TargetItem in the assignment. The status of that action/item relationship controls the flow of execution.

The RPS toolset provides this capability by managing state data about these executions and their intended target device. This allows the code to be implemented and updated quickly, while maintaining this complex web of requirements. Additions and changes become simply managing the order in which these activities are executed or making updates to the pieces that require an update, without fear of breakage elsewhere.

Finally, it is important to note that when executing a TaskMap a scoping mechanism was designed to allow automations that require cross-device dependencies to function. This mechanism is the "Container" as defined in the Device Metadata section of this document. Containers offer the "highest level" relationship between a set of intended targets. When developing TaskMaps which have these types of cross-device dependencies, those dependency checks are always performed within the scope of their parent Container.

Figure 7 - A visual representation of a TaskMap and how the logic is interpreted



TaskMap Definitions – Filtering on Criteria

When defining tasks for execution in a TaskMap, an optional filtering mechanism has been added to support choosing between items of similar types.

TaskMap Definition
Task1 : Target1 : Where Property1 = Value
Task2 : Target1 : Where Property1 = Value : And Task1 Completed
Task3 : Target2 : Where Property2 = Value2 : And Task2 Completed on Target1
Task3 : Target3 : Where Property3 = Value3 : And Task2 Completed on Target1

As shown in the figure above, when defining a TaskItem to be run within a TaskMap, the definition may contain a Target 'type' of device as well as provide criteria from that TargetItem's Properties to isolate specific devices. For example, if multiple "Target1" items existed in a Container, Property filters could be used to choose a specific instance of that item that meets only those criteria.

For example, a Vehicle may contain multiple computing units, but a specific computing unit may be required as a target. This method of target definition allows for control of which TargetItems receive assigned tasks.

Task and TaskMap Assignments

When a TaskItem or TaskMap is assigned to a TargetItem or Container for execution, an assignment record is generated in the RPS Database. This assignment record represents that execution request and persists the status of that request so the TaskMap can be coordinated as it is defined.

Tracking/reacting to this status data provides the core mechanism for how task maps operate as well as basic reporting information on whether or not a task was successful.

TaskMapAssignment records are persists a copy of the originating TaskMap and TaskMapDefinition along with the dependencies between definitions for automation sequencing. The TaskAssignment records are used to identify TaskItem and TargetItem execution status. Custom TaskAssignment properties are stored in an item property bag.

References

RPS Task State Diagram

The detailed design diagram for task states and how they are processed can be found here:

RPS > Trunk > Documents > Operations > **TaskStateDiagram v1.10.vsd**

NOTE

This document is intended to outline how task states are processed by the controller.

RPS CMDB – ERD Database Diagram

The detailed design diagram for database entity relationships (ER) can be found here:

RPS > Trunk > Documents > Architecture > **RPSEntityRelationshipDiagram.vsd**

NOTE

This document is intended to outline how database entities are related to each other.

More Resources

- [RPS Software Design](#)
- [RPS Configuration Management \(DSC\) Software Design](#)

Configuration Changes to a Two Domain Architecture

Last updated on August 3, 2021.

Last Reviewed and Approved on PENDING REVIEW

Data Changes for Architecture of two domains, Unit.domain & Rps.local

Account Changes

New

- An account with the role FileTransferServiceAccount needs to be added to the unit domain (i.e. 45bct.army.mil) that references the rpsadmin account (or Account with permissions to access the ContentStore) on the rps.local domain.

- For example:

```
RpsTransferAccount,ad.unit.domain,rps,rpsadmin,FileTransferServiceAccount,,FALSE,FALSE,FALSE,TRUE,,,
```

- A new account is needed with the Role FileTransferServiceAccount.
 - This account is assigned to all Targets that are not on the rps.local domain.
 - This account allows for non rps.local domain systems to connect to the ContentStore.

Modified

- RpsAdmin needs the role of Domain Admin and only assigned to the systems on the rps.local domain.
- The Following roles need to be changed and created in the rps.local domain only:
 - SqlServiceAccount
 - WebApiServiceAccount
 - TaskManagementServiceAccount
 - GuiServiceAccount
 - ProvisioningServiceAccount
 - CdnServiceAccount
- The following accounts need to be across all servers: the unit domain and the rps.local domain.
 - RpsDomainJoin (2 separate accounts, 1 for unit.domain and 1 for rps.local)
 - DatabaseAccount (this account is the same for both domains)
- There are two accounts with the name administrator for the unit domain:
 - One with the prefix of the server and the role LocalAdmin (AD\Administrator).
 - One with the prefix of the domain and the role DomainAdmin (Unit\Administrator).
 - There needs to be an account with the role LocalAdmin on all other servers on the unit.domain.

Certificates

Changes

- Each server on the unit.domain needs the following certificates:
 - WinRm
 - DscEncryption
 - RpsClientCdn
 - RpsRoot
- All servers with RPS need the following certificates:
 - WinRm
 - DscEncryption
 - RpsSyncSSL
 - RpsWebApiSsl
 - RpsSync

- RpsGuiSsl
 - RpsClientCdn
 - RpsRoot
 - MasterEncryption
 - NodeEncryption
- RPS servers used for provisioning need:
 - ProvisioningSsl

Properties

Changes

- DomainName - This is the property that identifies the domain the server is part of. Currently this is set on the Node level and inherited to the children. This needs to be set on the TargetItems that are different than the Node.
 - In our example, this is set on all TargetItems with type VirtualMachine that are part of the unit.domain.
- The ForestName on the Node is changed on the Node to rps.local (This could be prefixed with the unit).
- The ForestName needs to be added to the unit domain TargetItems so it does not inherit the ForestDomain from the node.
- The DnsServerName on the Node is changed to the RPS server FQDN (App.rps.local).
- Accounts with the following roles need to be added to the Domain Admins group on the RPS domain (rps.local):
 - ServerAdmin
 - GuiServiceAccount
 - TaskManagementAccount
 - WebApiServiceAccount
- The NIC assigned to the RPS server gets the property IsDnsListener and is set to \$true.
- The NIC assigned to all systems on the unit.domain gets the property DnsServer and is set to unit.domain (This could be prefixed with the unit).
- All RPS servers that are not a Domain Controller point their DNS to the Domain Controller server IP and set their local IP as the second DNS Server. This is set on their NIC.

ResourceItems

New

- A new RpsDomain needs to be added, rps.local (This could be prefixed with the unit).
- A new DnsZone is added, rps.local.
- A new partial to install DNS and set up secondary zones needs to be imported.
- All systems that have RPS but are not a Domain Controller need to assign the partial DNS to them.
- There needs to be multiple DNSZones created for each Unit.domain and rps.local:
 - Two rps.local DNS zones
 - 1 DNS zone, rps.local, with all RPS server IPs to include the NOSC RPS server IP.
 - 1 DNS zone, rps.local, with the NOSC RPS Server IP.
 - Two unit.domain DNS zones
 - 1 DNS zone, unit.domain, with all the RPS Server IPs and the unit.domain Domain Controller IP.
 - 1 DNS zone, unit.domain, with unit.domain Domain Controller IP.
- The RPS child systems at the TCN and SNE level get the rps.local and unit.domain DnsZones assigned.

Changes

- SecondaryServers property is added to DnsZones.
 - To be able to transfer the zones to the child RPS servers, we identify them as secondary servers that will host the zone.
 - This will also allow the unit domains to be transferred to RPS, and vice versa.

STIGs

Changed

- RPS at the NOSC level is assigned the STIGs that AD had.
- RPS at the TCN and SNE level are assigned the DNS STIG and some additional skip rules that are conflicting.

Data Validation Schema Definition

Last updated on August 24, 2021.

Document Status: Document Developer Quality Complete.

Introduction

This section describes JSON Schema and their properties.

Schema Definition

JSON Schema is a powerful tool for validating the structure of JSON data. JSON Schema are built around two core data structures: Objects and Arrays.

Object

A JSON object is a collection of properties inside a curly brace. The properties are key-value pairs where the key is always of type string and value can be any data type: string, boolean, or int.

```
{
  "Name" : "TestSchema",
  "IsList" : true
}
```

Array

A JSON array is a list of objects inside a square bracket and separated by a comma.

```
[
  {
    "Name" : "TestSchema1",
    "IsList" : true
  },
  {
    "Name" : "TestSchema2",
    "IsList" : false
  }
]
```

Keywords

The following keywords are used in JSON Schema:

KEYWORD	DESCRIPTION
Name	Describes the name of the schema.
Type	Describes the object type.
Properties	Describes the key-value pairs of an object.
AvailableExtensions	Describes extensions that can be used with the schema.

Data Types

The following data types are used in JSON Schema:

DATA TYPES	DESCRIPTION
System.String	This is enclosed in double quotes.
System.Int32	This is a non-fractional number ranging from 1 - 2147483647.
System.Boolean	This is true/false without quotes.

JSON Schema Example

A sample JSON Schema follows:

```
{
  "Name": "TestSchema",
  "Type": "TargetItem",
  "Properties": [
    {
      "Name": "MemoryMB",
      "Type": "System.Int32",
      "Attributes": [
        {
          "Name": "RequiredAttribute",
          "Values": {}
        },
        {
          "Name": "RangeAttribute",
          "Values": {
            "minimum": 1,
            "maximum": 2147483647
          }
        },
        {
          "Name": "DefaultValueAttribute",
          "Values": { "value": 2048 }
        }
      ]
    }
  ],
  {
    "Name": "BaseImage",
    "Type": "ResourceItem",
    "SchemaReference": "BaseImage",
    "NonValidationMetadata": {
      "DisplayName": "Base Image/Type"
    },
    "Attributes": [
      {
        "Name": "RequiredAttribute",
        "Values": {}
      }
    ]
  }
],
  "AvailableExtensions": [
    "IsDB",
    "IsCDN",
    "IsDC",
    "IsGUI",
    "IsProvisioningNic",
    "IsSync"
  ]
}
```


Getting to Know Properties

The following keywords are used when defining Properties:

KEYWORD	DESCRIPTION
Name	Describes the name of the property.
Type	Describes data types of the property.
SchemaReference	References the child schema.
Attributes	Describes validation attributes.
IsList	Describes whether the property is list or not.
NonValidationMetadata	Describes all the properties which do not require validation.

⚠ WARNING

When adding a new property to the schema, the `Name` keyword cannot be set to a value of *ExtraProperties*. Example:

```
{
  "Name": "ExtraProperties",
  "Type": "System.String"
}
```

ExtraProperties is used internally by RPS as a Data Validation keyword. If *ExtraProperties* is used in a schema, validation errors will occur:

- During item validation in REACTR.
- During the creation of a new Target Item or Resource Item: when validating against the updated schema for the item type that contains the property name *ExtraProperties*.

Attributes

Attributes are used to validate the given property. There are four custom validation attributes implemented. They are described below:

1. *ValidateDateTime* Attribute

```
{
  "Name": "Date",
  "Type": "System.String",
  "Attributes": [
    {
      "Name": "ValidateDateTime",
      "Values": { }
    }
  ]
}
```

ValidateDateTime Attribute validates the property which requires *Datetime* value. Currently, this attribute supports two date-time formats:

- MM/dd/yyyy
- MM/dd/yyyy hh:mm tt

2. *ValidateExpression* Attribute

```
{
  "Name": "Role",
  "IsList": true,
  "Type": "System.String",
  "Attributes": [
    {
      "Name": "ValidateExpression",
      "Values":
        {
          "expression": "$_ -Contains rpsadmin",
          "atLeast": 1
        }
    }
  ]
}
```

ValidateExpression Attribute validates the property by making sure the expression is true.

In this particular example, the expression is true if the name of the role contains *rpsadmin*. The value for the *atLeast* property is making sure the expression is true for at least 1 item in the list.

3. *ValidateIpAddress* Attribute

```
[
  {
    "Name": "IpAddressA",
    "Type": "System.String",
    "Attributes": [
      {
        "Name": "ValidateIpAddress",
        "Values":
          {
            "LowerBoundIp": "10.0.0.0",
            "UpperBoundIp": "10.0.0.20"
          }
      }
    ]
  },
  {
    "Name": "IpAddressB",
    "Type": "System.String",
    "Attributes": [
      {
        "Name": "ValidateIpAddress",
        "Values": {}
      }
    ]
  }
]
```

ValidateIpAddress Attribute validates the property type of the IP Address. A custom IP range can be set by setting *LowerBoundIp* and *UpperBoundIp*.

NOTE

If no value is assigned for *LowerBoundIp* and *UpperBoundIp*, then the attribute will assign default values of 0.0.0.0 to *LowerBoundIp* and 255.255.255.255 to *UpperBoundIp*.

In this particular case, it is not required to explicitly declare in schema.

Alternatively, a *Subnet* can be provided in CIDR range instead of a *LowerBoundIp* and *UpperBoundIp*. This will calculate the *LowerBoundIp* and *UpperBoundIp* based on the provided *Subnet*.

```
[
  {
    "Name": "IpAddressA",
    "Type": "System.String",
    "Attributes": [
      {
        "Name": "ValidateIpAddress",
        "Values": {
          "Subnet": "10.0.0.0/8"
        }
      }
    ]
  },
  {
    "Name": "IpAddressB",
    "Type": "System.String",
    "Attributes": [
      {
        "Name": "ValidateIpAddress",
        "Values": {}
      }
    ]
  }
]
```

In the example above, 10.0.0.0/8 would translate to a *LowerBoundIp* of 10.0.0.0 and *UpperBoundIp* of 10.255.255.255.

If neither a *Subnet* nor a *LowerBoundIP* and *UpperBoundIP* are assigned, the attribute will assign default values of 0.0.0.0 to *LowerBoundIp* and 255.255.255.255 to *UpperBoundIp*.

4. *ValidateMacAddress* Attribute

```
{
  "Name": "Date",
  "Type": "System.String",
  "Attributes": [
    {
      "Name": "ValidateMacAddress",
      "Values": { }
    }
  ]
}
```

ValidateMacAddress Attribute validates that the property type is a MAC Address.

For more information on additional attributes such as *RequiredAttribute*, *RangeAttributes*, and *DefaultValueAttribute*, refer to Microsoft docs link: <https://docs.microsoft.com/en-us/dotnet/api/system.componentmodel.dataannotations?view=net-5.0>

Schema Reference

In the TestSchema below, we have a property with the *Name* "BaselImage". It has *SchemaReference* property with value "BaselImage". BaselImage is a schema with its own sets of all JSON data types and properties. TestSchema will inherit all the properties of BaselImage schema.

```
{
  "Name": "BaseImage",
  "Type": ResourceItem,
  "SchemaReference": "BaseImage",
  "NonValidationMetadata": {
    "DisplayName": "Base Image/Type"
  },
  "Attributes": [
    {
      "Name": "RequiredAttribute",
      "Values": {}
    }
  ]
}
```

Create, Update and Delete the Schema

Currently, all schemas must be manually created, updated, and deleted.

Data Validation Integration in RPS

Last updated on August 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

This section describes the integration of the Data Validation service in RPS.

Data Validation in RPS

The Data Validation service validates RPS object types (ResourceItem, ResourceGroup, TargetItem and TargetGroup) on all create, update, and delete operations. During this process, Data Validation validates that all properties on the item are in alignment with the schema for that object's type. New-RPSResourceItem, New-RPSTargetItem, New-RPSTargetGroup, and New-RPSResourceGroup are some of the commands that trigger Data Validation.

For more information on schemas and their validation attributes, refer to: [Data Validation Schema Definition](#).

Schema and Log File Location

- Validation log files are located in `C:\ContentStore\DataValidation\logs\`
- Schema are placed in folders based on their types:
 - Schema of ResourceItem types are located in `C:\ContentStore\DataValidation\Schemas\ResourceItem\`
 - Schema of ResourceGroup types are located in `C:\ContentStore\DataValidation\Schemas\ResourceGroup\`
 - Schema of TargetItem types are located in `C:\ContentStore\DataValidation\Schemas\TargetItem\`
 - Extensions are located in `C:\ContentStore\DataValidation\Schemas\Extensions\`

Throwing Errors & Logging with Data Validation

Data Validation can be configured to behave in one of two ways:

- Log errors to the validation log file.
- Throw errors and halt the operation.

The RPS settings key *LogSchemaValidation* controls this Data Validation configuration.

NOTE

By default, *LogSchemaValidation* is set to null, which is considered "true".

If the value of *LogSchemaValidation* is true (or null/default), Data Validation will log any errors to the validation log file and will complete the operation.

If the value of *LogSchemaValidation* is false, Data Validation will throw any errors and halt the operation.

Get *LogSchemaValidation* Value

Settings key *LogSchemaValidation* value can be obtained using the following command:

```
Get-RpsStorageValue -Key LogSchemaValidation
```

Set *LogSchemaValidation* Value

NOTE

LogSchemaValidation value can only be set by users with the SuperAdmin or RpsAdmin roles.

Settings key *LogSchemaValidation* value can be set with the following command:

```
Set-RpsStorageValue -Key LogSchemaValidation -Value false
```

In this example, *LogSchemaValidation* value is set to false.

Data Validation Example

This example creates a new resource item of type VMKernelAdapter (which targets the VMKernelAdapter schema). **VlanId** is required field in the schema.

- When *LogSchemaValidation* value is true:
 1. Set the *LogSchemaValidation* value to true.

```
Set-RpsStorageValue -Key LogSchemaValidation -Value true
```

2. Create property object without **VlanId** and run `New-RpsResourceItem` command.

```
$vmKernelAdapterProps = @{  
    VMKernelAdapterName = "VMKernelAdapterName"  
    IPAddress = "192.168.1.2";  
    Subnet = "255.255.255.0";  
}  
  
New-RpsResourceItem -Type VMKernelAdapter -Name "vmka" -Properties $vmKernelAdapterProps
```

In this scenario, the resource item is created because the rps setting value is set to true. The expected error is:

```
VMKernelAdapter data validation error: The VlanId field is required.
```

It is logged in the Validation log file at `C:\ContentStore\DataValidation\logs\`.

- When *LogSchemaValidation* value is false:
 1. Set the *LogSchemaValidation* value to false.

```
Set-RpsStorageValue -Key LogSchemaValidation -Value false
```

2. Create property object without **VlanId** and run `New-RpsResourceItem` command.

```
$vmKernelAdapterProps = @{  
    VMKernelAdapterName = "VMKernelAdapterName"  
    IPAddress = "192.168.1.2";  
    Subnet = "255.255.255.0";  
}  
  
New-RpsResourceItem -Type VMKernelAdapter -Name "vmka" -Properties $vmKernelAdapterProps
```

In this scenario, the resource item will not be created because the rps setting value is set to false. The command will throw the error `VMKernelAdapter data validation error: The VlanId field is required.`

It is logged in the Validation log file at `C:\ContentStore\DataValidation\logs\`.

RPS Certificate Management Technical Design

Last updated on May 05, 2021.

Last Reviewed and Approved on PENDING REVIEW

Table of Contents

- [Introduction](#)
- [Certificate Management Overview](#)
- [Rolling Certificate Process](#)
- [Certificate data in RPS](#)

Introduction

This document describes the Certificate Management feature and its technical design. The technical design was established based on requirements gathered from the customer, constraints of existing systems, and an analysis of benefits and tradeoffs of various design decisions.

Certificate Management Overview

RPS Certificate Management describes the approach to storing and managing certificates in the RPS configuration management database (CMDB). The data stored in the CMDB can then be used to deploy the certificates to Targets also stored in the CMDB. Certificates are represented and managed in the RPS CMDB as resource items that describe the properties and attributes of the certificate. For a list of detailed certificates properties, please see [Certificate data in RPS](#). For more information about specific certificate usage in RPS, please see [Certificate Usage](#)

Rolling Certificate Process

RPS Certificate Management provides the ability to roll certificates. The term "roll" a certificate, refers to replacing one certificate with another certificate of the same type. A few potential reasons for rolling certificates include, a security breach, certificate expirations, or moving from self-signed to Certificate Authority signed.

The process of rolling RPS certificates is orchestrated by invoking a task map that consists of the following runbooks:

RUNBOOK NAME	DESCRIPTION
Refresh-RpsSelfSignedCerts	The request disposition message returned by the Certificate Authority after a certificate request submission.
Update-MasterKeyProtection	Imports the master key certificate and encrypts the master key with the new certificate.
Set-EncryptionSettings	Imports and configures the certificate used for PowerShell Desired State Configuration (DSC) encryption and decryption.
Set-WinRmSettings	Installs and configures the certificates used for Windows Remote Management (WinRm).
Publish-DscConfiguration	Using data in the CMDB, compiles and publishes DSC configurations to targets. DSC will install and configure certificates across the various application and RPS components.

Certificate data in RPS

Certificates are represented in the RPS database as Resource Items of type Certificate. As of RPS 4.0, additional properties are supported on Certificate resource items to support the rolling process. The following tables describes each property used for RPS certificates.

PROPERTY NAME	DESCRIPTION
DispositionMessage	The request disposition message returned by the Certificate Authority after a certificate request submission.
ExpirationDate	Certificate expiry date.
FriendlyName	Certificate Friendly Name.
GenericRole	Generic role used to provision the certificate based on an RPS certificate template.
IssuedBy	Issuer of the certificate.
Password	Password used to export the certificate's private key.
PrivateBase64Content	Base 64 encoded private key.
PublicBase64Content	Base 64 encoded public key.
PublicKeyPath	Location of the Public Key certificate on the file system.
RequestDisposition	The request disposition flag returned by the Certificate Authority after a request submission.
RequestId	The ID of the certificate request returned by the Certificate Authority.
RequestStatus	The status of the certificate request. Complete/Pending/Incomplete.
Role	The certificate role name that defines its RPS use case.
RoleValidForRegen	Specifies whether the certificate can be rolled.
SigningType	Type of certificate. RpsSigned/CASigned
SubjectAlternativeName	Certificate Subject Alternative Name (SAN).
SubjectName	Certificate Subject Name.
TemplateName	The name of the template to use when requesting a certificate from a Certificate Authority.
Thumbprint	Thumbprint of the certificate.

Certificate Usage

Last updated on August 30, 2021.

Document Status: Document Developer Quality Complete.

Introduction

The future security needs of the RPS security infrastructure are currently planned to depend on a Public Key Infrastructure (PKI). However, the current landscape of development for the project does not allow for the full implementation of PKI. In its absence, a Self-Signed Certificate strategy has been developed as a temporary measure to provide improved security over plain text secrets and ease the future adoption of full PKI.

By default, RPS includes a variety of certificates (even self-signed/RPS-signed) to showcase functionality, and it is expected that these development or test certificates will be replaced with appropriate secure and trusted certificates to perform the various functions using the roles indicated.

⚠ IMPORTANT

Each certificate must have a certificate root that is trusted by the local host (i.e., Trusted Root Certification Authorities).

⚠ WARNING

Use of self-signed or untrustworthy certificates presents a security risk for all assets and functions "secured" by said certificates.

ℹ NOTE

The .pfx file is capable of storing both public and private keys whereas the .cer file is generated from the .pfx and contains only the public key.

Certificate Roles and Functions

The tables below map certificates in the ContentStore, as well as certificates generated by the deployment, to their role and corresponding function.

RPS Specific Roles

The following table describes all the RPS Certificate Roles. For details on how each certificate's requirements, see corresponding Generic Role row in the [Generic Role Templates](#) table below.

RPS ROLE NAME	GENERIC ROLE NAME	SCOPE	OTHER NOTES
CertManager	ClientAuthentication	Hosts that will access the Certificate Manager plugin	Used for certificate authentication with Certificate Manager Plugin
DscEncryption	DscEncryption	All computers	Credential encryption in DSC .mof files
DscPullServer	DscPullServer	All computers configured for DSC Pull mode	Used for certificate authentication with DSC Pull Server
MasterKeyEncryption	MasterKeyEncryption	Computers where RPS protected properties will need to be accessed	Used for decryption of RPS Master Key

RPS ROLE NAME	GENERIC ROLE NAME	SCOPE	OTHER NOTES
NodeEncryption	DscEncryption	Provisioning hosts	Used for encryption of exported RPS Node data
ProvisioningSSL	ProvisioningSSL	Provisioning hosts	RPS Provisioning endpoint
RdtSsl	SSL	Computers where RDT is installed	RDT UI HTTPS binding
RpsApi	ClientAuthentication	Hosts that will access the RpsApi plugin	Used for certificate authentication with RPS Api Plugin
RpsGuiSSL	SSL	RpsGui hosts	RPS Gui HTTPS binding
RpsPackage	ClientAuthentication	All Computers	Used for certificate authentication with RPS Package Manager Plugin
RpsRoot	Root	All Computers	Used to sign initial RPS Certificates
RpsSync	ClientAuthentication	RpsSync hosts	Cert:\CurrentUser\My for Sync account
RpsSyncSSL	SSL	RpsSync hosts	RpsSync HTTPS endpoint
RpsWebApiSsl	SSL	RPS Web API hosts	RPS Web API HTTPS endpoint
WinRm	ServerAuthentication	All computers	PowerShell HTTPS endpoint

Generic Role Templates

The following table describes the specific certificate attributes required by each generic role. The Key Usages and Enhanced Key Usages should be used for referenced when creating certificate templates. The signature algorithm and key length columns indicate the default values for certificates signed by RPS root certificate. All RPS certificate roles support Elliptical Curve Cryptography based algorithms and larger key lengths, with the exception of DscEncryption. The certificate used for DscEncryption only support RSA algorithm.

GENERIC ROLE NAME	KEY USAGES	ENHANCED KEY USAGES	SIGNATURE ALGORITHM	KEY LENGTH
ClientAuthentication		Client Authentication (1.3.6.1.5.5.7.3.2)	SHA256	2048
DscEncryption	Key Encipherment, Data Encipherment (30)	Document Encryption (1.3.6.1.4.1.311.80.1)	SHA256	2048
DscPullServer	Digital Signature (80)	Client Authentication (1.3.6.1.5.5.7.3.2)	SHA256	2048
ProvisioningSSL	Data Encipherment, Key Encipherment (e0)	Server Authentication (1.3.6.1.5.5.7.3.1)	SHA256	2048
Root	Certificate Signing, Off-line CRL Signing, CRL Signing (06)		SHA256	4096

GENERIC ROLE NAME	KEY USAGES	ENHANCED KEY USAGES	SIGNATURE ALGORITHM	KEY LENGTH
ServerAuthentication	Digital Signature, Non-Repudiation, Key Encipherment (e0)	Server Authentication (1.3.6.1.5.5.7.3.1)	SHA256	2048
SSL	Digital Signature, Non-Repudiation, Key Encipherment (e0)		SHA256	2048

Generating Certificates

Certificates can be generated as part of the installer process or supplied from an external PKI. By default, the `New-RpsNodeConfiguration.ps1` script will generate self-signed certificates for each role and server using the existing configuration data. If external certificates will be used, the certificate data file located at

`{ContentRoot}\Setup\Configuration\MNCertificateData.psd1` will need to be updated to store the certificate role and password information. The certificates themselves must also be stored in the following path: `{ContentRoot}\Certificates`. The naming convention required for each certificate file should be as follows: `{TargetItemName}_{CertificateRole}.pfx/cer`.

Set-RpsCertificate

As part of the Rps-Encryption PowerShell module, the `Set-RpsCertificate` function generates a certificate based on Rps template and imports it into the CMDB. If the certificate already exists at the path specified, it will only import the certificate into the CMDB.

For detailed documentation on this function from PowerShell, run `Get-Help Set-RpsCertificate`.

Example:

```
$properties = @{
    SigningCertificate = @{
        Name      = 'RpsRoot.pfx'
        Password = 'ExamplePasswordHere'
    }
    CertificateFolderPath = 'C:\ContentStore\Certificates'
    'Member.Unit.Domain' = @{
        RpsSync = 'ExamplePasswordHere'
    }
}
$targetItem = Get-RpsTargetItem -Name 'Member.Unit.Domain' -Type 'VirtualMachine'
Set-RpsCertificate -Role RpsSync -Target $targetItem -Properties $properties
```

New-RpsCertificate

Also part of the Rps-Encryption module, the `New-RpsCertificate` function allows you to create template driven certificates. The function will generate certificates but do not import the certificate into an existing Rps session.

For detailed documentation on this function from PowerShell, run `Get-Help Set-RpsCertificate`.

Example:

```

$parameters = @{
    Type = 'SSL'
    SubjectName = 'Member'
    SubjectAlternativeName = 'member.unit.domain'
    FriendlyName = 'Member.unit.domain RpsWebApiSSL'
    OutputPath = 'C:\ContentStore\Certificates\Member.unit.domain_RpsWebApiSSL.pfx'
    Password = ConvertTo-SecureString 'ExamplePassword' -AsPlainText -Force
    NotBefore = Get-Date
    NotAfter = (Get-Date).AddYears(2)
    SigningCertificatePath = 'C:\ContentStore\Certificates\RpsRoot.pfx'
    SigningCertificatePassword = ConvertTo-SecureString 'ExamplePasswordHere' -AsPlainText -Force
}

New-RpsCertificate @parameters

```

Import-RpsCertificate

As part of the Rps-Installer module, the `Import-RpsCertificate` function allows you to import an existing certificate into the Rps CMDB.

For detailed documentation on this function from PowerShell, run `Get-Help Set-RpsCertificate`.

Example:

```

# Get the target item to assign the certificate to.
$targetItem = Get-RpsTargetItem -Name 'Member.Unit.Domain' -Type 'VirtualMachine'
$password = ConvertTo-SecureString 'ExamplePasswordHere' -AsPlainText -Force
Import-RpsCertificate -Name 'Member.unit.domain_RpsWebApiSSL' -Path
'C:\ContentStore\Certificates\Member.unit.domain_RpsWebApiSSL.pfx' -Password $password -AssignTo $targetItem
-Role RpsWebApiSsl

```

The `New-RpsCertificate` function implements the `New-RpsSelfSignedCertificate` function in the Rps-Encryption Module. The `New-RpsSelfSignedCertificate` function is generic and allows the configuration of many different certificate settings.

PostgreSQL Encryption

SSL connections encrypt all data sent across the network: the password, the queries, and the data returned. The `pg_hba.conf` file allows administrators to specify which hosts can use non-encrypted connections (`host`) and which require SSL-encrypted connections (`hostssl`). Also, clients can specify that they connect to servers only via SSL. Stunnel or SSH can also be used to encrypt transmissions.

RPS Database Encryption

RPS is configured to use SSL connections for the RPS CMDB using DSC. The certificate used to secure the DEK is generated automatically with DSC, is called `RpsDatabaseCertificate.crt`, and is backed up to disk (by default in `C:\Backups\Certificates`). The server's master key is backed up to `RpsDatabaseMasterKey.crt` using the password supplied for the RPS Configuration.

WARNING

The compromise of the certificates could allow malicious users to retrieve unencrypted data. Follow proven certificate management and backup practices to mitigate security vulnerabilities while preserving the ability for a legitimate administrator to restore the RPS CMDB or TMS databases if needed.

Certificate Requirements for Linux Clients

Last updated on January 28, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

This document describes the certificate requirements to leverage Patch Management in Rapid Provisioning System (RPS) for Linux clients. Additionally, this document also provides instructions on installing the certificate required.

Document Overview

Patch Management in RPS requires communication via HTTPS. The certificate authority (CA) that signed the webserver's certificate must be trusted by the Linux client or patches will not be downloaded. This is done by installing the public certificate of the CA. This document is considered a living document and subject to change.

Installing the RPS CA Public Certificate

1. Copy the RPS CA public certificate to Linux machine.
 - a. The RPS CA public certificate is located in `\ContentStore\Certificates\RpsRoot.cer`
2. Convert to .pem file with openssl tool.
 - a. `openssl x509 -inform der -in certificate.cer -out certificate.pem`
 - b. If you receive a 0D0680A8 and 0D07803A error, the certificate is already in the correct format. The only change needed is to change the certificate's file extension from .cer to .pem
3. Rename RpsRoot.cer to RpsRoot.pem
 - a. `mv RpsRoot.cer RpsRoot.pem`
4. Once the certificate has the .pem extension copy certificate to:
 - a. `/etc/pki/ca-trust/source/anchors/`
5. Import the certificate with the following command:
 - a. `update-ca-trust extract`
 - b. The certificate will be added to the `/etc/pki/ca-trust/extracted/pem/tls-ca-bundle.pem` file.
6. Verify the certificate imported with the following command:
 - a. `cat /etc/pki/ca-trust/extracted/pem/tls-ca-bundle.pem | grep RPS`

Certificate Request Plugin Configuration

Last updated on September 9, 2021.

Document Status: Document Developer Quality Complete.

Introduction

The Certificate Request REST endpoint is a plugin that runs on the RPS Web API. The plugin serves as a Certificate Authority (CA) interface for RPS, but can operate independently of RPS.

Requirements

The following sections describe the requirements for configuring the Certificate Request plugin.

RPS Settings

RPS Settings are encrypted settings stored in `%APPDATA%\Rps\RpsSettings.txt`. The settings can be configured using `Set-RpsStorageValue` cmdlet and retrieved using `Get-RpsStorageValue` from the Rps-API module. The following settings are required for the RPS Web API host to run correctly:

SETTING NAME	PURPOSE
RpsWebApiThumbprint	Thumbprint of the certificate that will be used for RPS Web API host SSL binding.
RpsWebApiUrl	The URL the RPS Web API host will listen on. For example: https://member.unit.domain:8080

RpsWebApi Files

FOLDER NAME	PURPOSE
C:\Source\RpsWebApi	Contains the files required to run the RPS Web API host.
C:\Source\RpsWebApi\Plugins\Rps.CertificateRequest.RpsPlugin	Contains the plugin required to run the Certificate Request REST endpoint.

RpsWebApi Service

RpsWebApi should be configured as a service with the executable path set to: `C:\Source\RpsWebApi\Rps.Web.Api.exe`.

RpsWebApi Service Account

The Windows account running the RpsWebApi service needs the following:

- Granted the *Log on as a service* User Rights Assignment.
- The certificate specified by thumbprint in the RpsWebApiThumbprint setting, will need to be installed in the Cert:\CurrentUser\My certificate store of the RpsWebApi service account.
- Registered Service Principal Name (SPN) in the domain. For example, from a Windows command prompt using the `setspn` command, the following code snippet would register the service account named `RpsWebApiSvc` for HTTP service running on a server named `member.unit.domain`:

```
setspn -S HTTP/member.unit.domain RpsWebApiSvc
```

Certificate Request Process

Last updated on September 9, 2021.

Document Status: Document Developer Quality Complete.

Introduction

The goal of this article is to describe the workflow to request and retrieve certificates from a Certification Authority (CA) using RPS. It is a data-driven process that uses RPS CMDB to take the existing certificate resource items and generate certificate signing requests (CSRs) and then retrieve the certificates from the CA. Throughout the process, the CMDB is updated with the responses from the CA.

The RPS certificate request process pairs with Certificate Rolling and should be performed prior to certificate rolling in order to roll with CA signed certificates. See [Certificate Rolling](#) for more details on rolling certificates.

Prerequisites

In order for the Certificate Request Process to work, the following prerequisites must be in place. Please see [RPS Certificate Management Technical Design](#) for additional details on RPS Certificate Management architecture.

- RPS Web API host running on a node, for example NOSC, (see `DSC Configuration ContentStore\Dsc\PartialConfigurations\RpsWebApi.ps1`), with the following plugins present under `ContentStore\RpsWebApi\Plugins`:
 - `.\Rps.Api.Rest.RpsPlugin`
 - `.\Rps.CertificateManager.RpsPlugin`
- RPS Web API host running on a server with the following:
 - Network access to the intended Active Directory Certificate Services (AD CS) Certification Authority (CA).
 - See [Certificate Request Plugin Configuration](#) for details on `Rps.CertificateRequest.RpsPlugin` configuration.
- Active Directory Certificate Services Certification Authority installed and configured, for example DCA.
- AD CS CA certificate templates created for the supporting RPS certificates. See the [Generic Role Templates section in Certificate Usage](#) for details on template requirements.

CA Enabled Certificate Resource Items

The RPS Certificate Request process will take action on ResourceItems of Type `Certificate` and have a property `SigningType = 'CaSigned'`. RPS Certificate Manager will take appropriate action based on the `RequestStatus` property value of each ResourceItem, as described in the table below. RPS Certificate Manager will set the values to `Pending` and `Complete` when appropriate, but new ResourceItems with `SigningType = 'CaSigned'` should have their `RequestStatus = 'NotRequested'`.

REQUESTSTATUS	ACTION	DESCRIPTION
NotRequested	Create a CSR.	This should be the initial value.
Pending	Retrieve the certificate public key from the CA.	This value will be set after a CSR has been successfully submitted to the CA.
Complete	No action taken.	Once the public key has been successfully retrieved from the CA, this value will be set.

New-RpsCASignedResource

The recommended method to create the CA Enabled Certificate ResourceItems is using the `New-RpsCASignedResource` function from Rps-Encryption module. The function will use from existing certificate ResourceItems to generate a CA Enabled certificate ResourceItem that will be signed by a certificate authority. The created resource item will be based on an existing certificate resource item properties that would be needed to create a CSR to fulfill an appropriate certificate for the respective role.

Example

The following example would take all Certificate ResourceItems that are not `CASigned` or have the role `RpsRoot` and generated the necessary CA Enabled Certificate ResourceItems:

```
$certificates = Get-RpsResourceItem -Type $Rps.ResourceTypes.Certificate | Where-Object -FilterScript
{$_ .SigningType -ne 'CASigned' -band $_.Role -ne 'RpsRoot'}
$certificates | New-RpsCASignedResource
```

Process Certificates REST Endpoint

The certificate request process is initiated by using a REST client to invoke the `ProcessCertificates` endpoint on the RPS Certificate Manager plugin. Any REST client can be used, but this article will be using the PowerShell cmdlet `Invoke-RestMethod` to make the REST request.

REST Parameters

NAME	TYPE	DESCRIPTION
NodeId	Guid	The node ID to request certificates for.
CaName	String	The Certification Authority name to submit requests to. Example format of a CA Name would be <code>AD.unit.domain\TPKI-LAB-DCA-CA</code> .

Example

The following example shows how to request certificates from the NOSC for a child node like TCN.

```
$node = Get-RpsNode -Name NOSC
$childNode = Get-RpsNode -Name TCN

# Get the active client authentication certificate for the NOSC CertManager REST Endpoint
$targetItem = Get-RpsTargetItem -Type VirtualMachine -Name nosc.rps.local
$clientAuthCertificate = Get-RpsResourceItem -Type Certificate -Role 'CertManager' -TargetItem $targetItem -
IsActive $true

$processCertificatesParameters = @{
    # The URI and parameters for the certificate manager endpoint.
    Uri = "https://nosc.rps.local:777/CertManager/v1.0/CertificateManager/ProcessCertificates/?
nodeId=${$childNode.Id}&caname=${$node.CertificateAuthorityName}"
    # The certificate for this corresponding thumbprint must be installed in either the Computer or User MY
cert store.
    CertificateThumbprint = $clientAuthCertificate.thumbprint
}

# Use Invoke-RestMethod cmdlet to initiate the certificate request process
Invoke-RestMethod @processCertificatesParameters
```


Rolling Certificates

Last updated on February 12, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

Certificates can be rolled "replaced" through both PowerShell and the RPS Web UI.

The certificate rolling process is executed through the assignment of the `UpdateNodeCertificates` TaskMap (which is of TaskMap type `CertificatesManagement`). This TaskMap contains the instructions, or more specifically the Task Map steps, responsible for correctly publishing the updated certificate configuration. Additionally, it also creates and activates new RPS signed and CA signed certificates, along with publishing Desired State Configuration (DSC), configuring Windows-Remote Management (WinRM) encryption settings, and other RPS certificate roles.

Through PowerShell, the `UpdateNodeCertificates` TaskMap can be utilized by either making an assignment to a specific TargetItem, or to a TargetGroup. However, in this documentation, we will focus on rolling certificates to all targets (using the dynamic/smart group) under a specified Node. The process of making the assignment between the `UpdateNodeCertificates` TaskMap and the dynamic group is automated, and subjectively much simpler, through the Web UI.

Dynamic groups are essentially auto-generated TargetGroups, based on a set of filters and conditions. For more detailed documentation on dynamic groups, please reference [Creating a Dynamic Group](#). The dynamic group which we will be primarily concerned with for certificate rolling is of type `ManagedCertificate_Targets`, and will be named based on the following syntax `{Name}-ManagedCertificate` (where **Name** is the name of the Node).

Step-by-step instructions on how to roll certificates, through your method of choice, can be found below.

PowerShell

1. You will need to obtain and store the `UpdateNodeCertificates` TaskMap.

```
$taskMap = Get-RpsTaskMap -Type $Rps.TaskMapTypes.CertificatesManagement -Name  
$Rps.TaskMapNames.UpdateNodeCertificates
```

NOTE

If you already know the name of the Node for which you would like to roll certificates, you may skip this step.

2. We will be getting the Node by Id, and storing its name.

```
$node = Get-RpsNode -Id "1a38129b-b8ac-4523-be79-94cfc929ba4b"  
$nodeName = $node.Name
```

3. Get and store the dynamic group 'TargetGroup' by name, using the following syntax:

```
$dynamicGroup = Get-RpsTargetGroup -Name "$nodeName-ManagedCertificate"
```

4. Finally, we will create the TaskMapAssignment between the previously saved TaskMap and dynamic group. This will initiate the certificate rolling process.

```
New-RpsTaskAssignment -TaskMap $taskMap -TargetGroup $dynamicGroup -NodeIdToRunOn $node.Id
```

Web UI

1. Through the Web UI, certificate rolling is performed through the **Certificate Management** web page (accessible via the Distribution drop-down). Please navigate to this page.

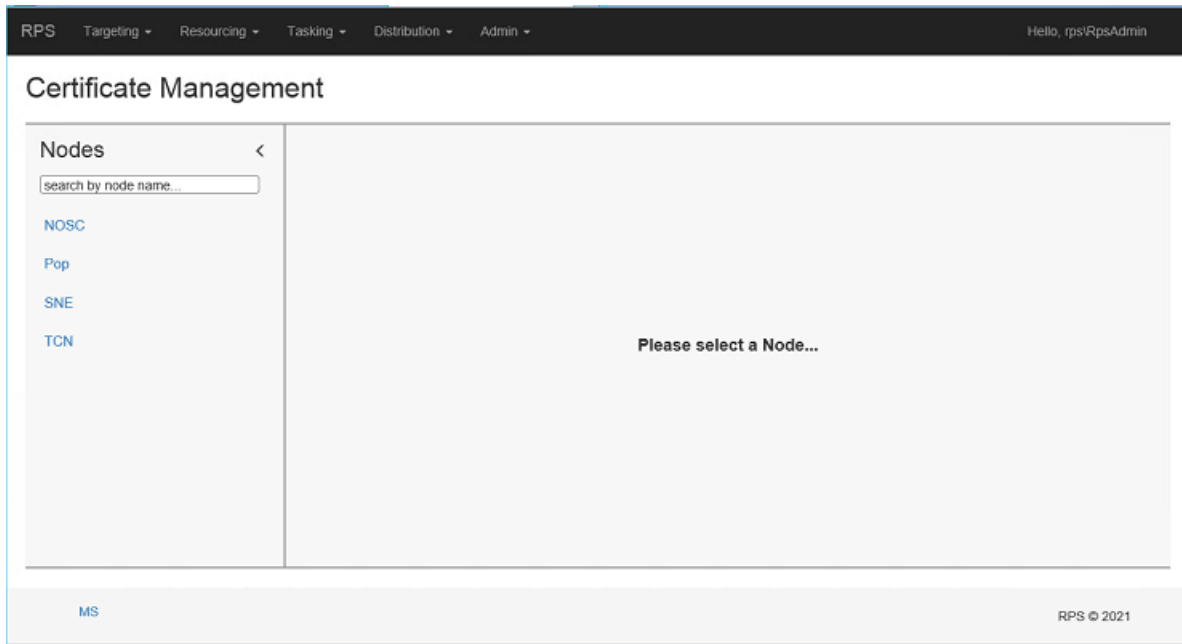


Figure 1: Certificate Management page

1. Using the panel on the left-hand side of the screen, search for and then select the Node for which you would like to roll/deploy certificates to.

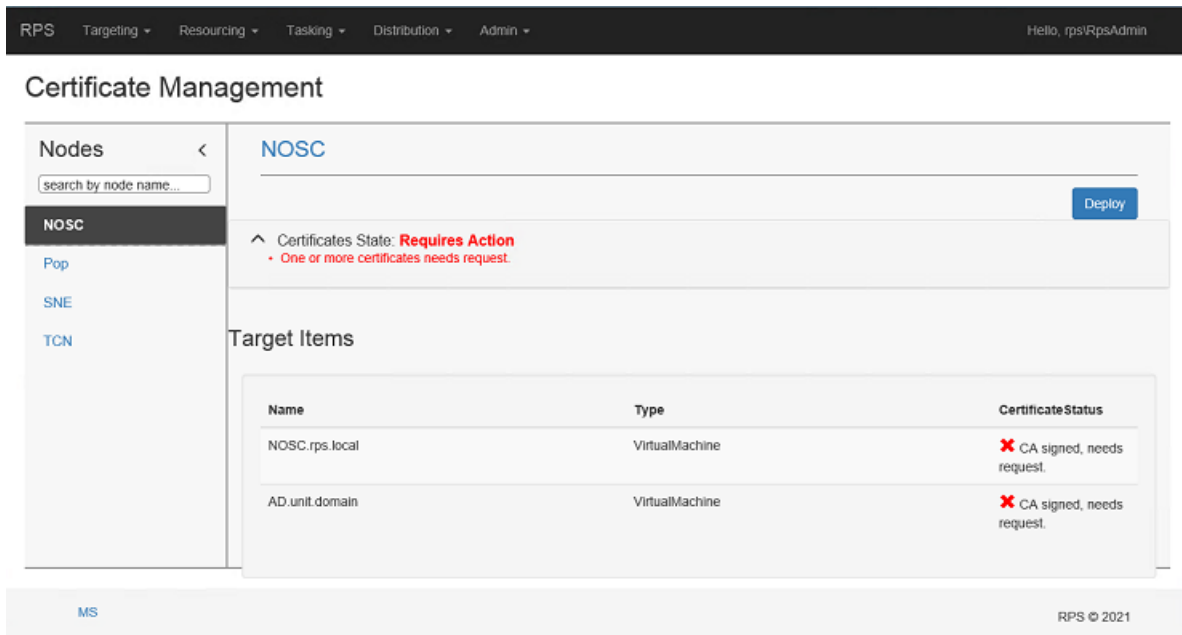


Figure 2: Selecting Node

1. When you are ready, click on the `Deploy` button for the selected Node on the **Certificate Management** page. This will assign the `UpdateNodeCertificates` TaskMap to the dynamic group, and begin the certificate rolling process.

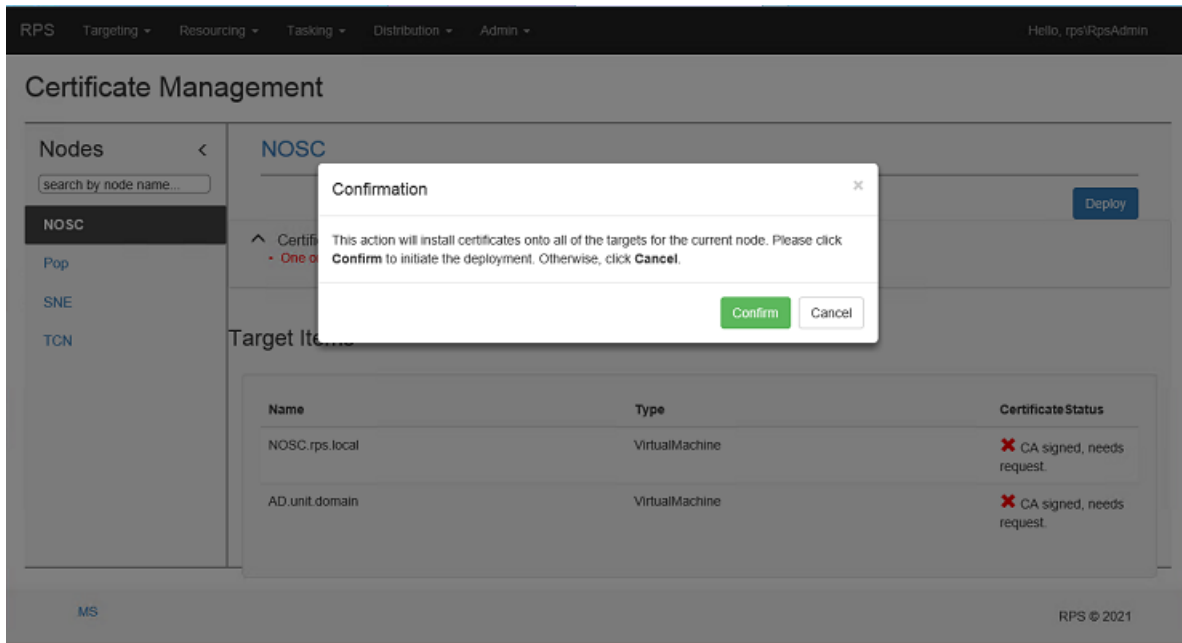


Figure 3: Confirm Install

1. To check the status of the certificate rolling processes, navigate to the **Assignments** page by selecting Assignments from the Tasking drop-down on the menu bar.

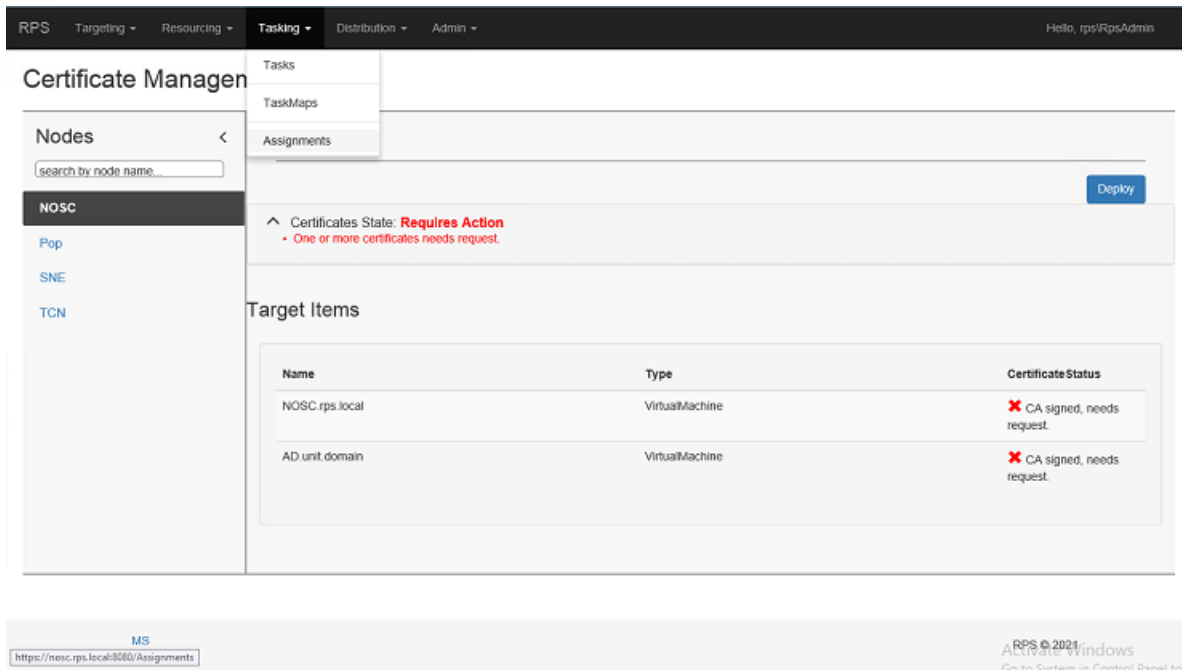


Figure 4: Checking Rolling Status

1. On the left side of the **Assignments** page you can filter by Target Group or by Status. You can view the Status of your deployment within the filtered results in the center of the page. If any, you can view applicable messages on the right side of the **Assignments** page under **Message**

RPS Targeting Resourcing Tasking Distribution Admin Hello, rps/RpsAdmin

Auto Refresh List

Filters

All Active InActive

Name
nos

Status
Any ...

Assignments

Target Item ↑	Target Group	Workflow	Task Map	Status	Assignment Date	Start Date	Message
NOSC.rps.local	NOSC-ManagedCertificate	Export-NodeData	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:39:32.030575	Completed
NOSC.rps.local	NOSC-ManagedCertificate	Publish-DscConfiguration	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:40:32.496321	Completed F DscConfigur
NOSC.rps.local	NOSC-ManagedCertificate	Set-NodeCertificateThumbprint	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:35:29.520359	Completed S NodeCertific
NOSC.rps.local	NOSC-ManagedCertificate	Set-WinRMSettings	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:38:31.461111	Completed S on NOSC.rp
NOSC.rps.local	NOSC-ManagedCertificate	Set-EncryptionSettings	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:37:30.912558	Completed S EncryptionS NOSC.rps.ic
NOSC.rps.local	NOSC-ManagedCertificate	Update-MasterKeyProtection	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:36:30.00922	Completed U MasterKeyP
NOSC.rps.local	NOSC-ManagedCertificate	Refresh-RpsSelfSignedCerts	UpdateNodeCertificates	Completed	2021-08-06T17:33:42.953109	2021-08-06T17:34:28.83182	Completed F RpsSelfSign NOSC.rps.ic

MS RPS © 2021

Figure 5: Checking Rolling Status

Authoring RPS DSC Partial Configurations

Last updated on March 9, 2020.

Last Reviewed and Approved on PENDING REVIEW

DSC Partial Configurations (Partial Configs) are used to organize and apply a set of configuration settings to a target computer. Multiple partial configs can be applied and combined to form the full DSC configuration applied to a target computer. RPS-enabled Partial Configs are ones that adhere to specific guidelines which allow them to be used by RPS to configure and publish configurations to computers on an RPS Node.

This document contains the guidelines to create RPS-enabled Partial Configurations.

Outline

1. RPS-Enabled Partial Config
2. Common Parameters
3. RPS-Mapped Parameters
4. DependsOn
5. Importing Partial Configs into RPS
6. Testing Partial Config data with RPS
7. Publishing from RPS

RPS-Enabled Partial Config

What is "RPS-Enabled"?

"RPS-Enabled" indicates that a Partial Config is capable of being automatically published by RPS. In order to do this, a partial must adhere to certain rules:

1. Partial must contain baseline parameters used for secure publishing.
2. Partial must describe its dependencies and required inputs.
3. Partial and DSC Modules must be imported into RPS.

Sample Partial Config

This sample partial configuration shows the common elements of an RPS-enabled partial.

```

using namespace Rps.Api.PowerShell
using namespace Rps.Api.Constants

[DependsOn(DscPartialName = 'OSCore')]
Param
(
    [Parameter(Mandatory = $true)]
    [ValidateScript({[IPAddress]::Parse($_)})]
    [string]
    $IPAddress,

    [Parameter(Mandatory = $true)]
    [ResourceItemMapping(EntityType = [ResourceTypes]::Certificate, Role = 'DSCEncryption')]
    [Hashtable]
    $DSCEncryptionCertificate,

    [Parameter(Mandatory = $true)]
    [string]
    $OutputPath
)

Configuration SampleConfiguration
{
    Import-DscResource -ModuleName 'PSDesiredStateConfiguration' -ModuleVersion 1.1
    Import-DscResource -ModuleName 'SampleDscModule' -ModuleVersion 1.0

    Node $IPAddress
    {
        File SampleFile
        {
            Ensure = 'Present'
            SourcePath = "sourcepath"
            DestinationPath = "destinationPath"
            Force = $true
        }

        SampleDscResource SampleResourceName
        {
            Ensure = 'Present'
            Properties = Values
        }
    }
}

$certificatesPath = (Get-Item "$PSScriptRoot\..\..\Certificates").FullName

$ConfigData = @{
    AllNodes = @(
        @{
            # DSC Encryption common data
            CertificateFile = ConvertTo-RootedPath -Filename $DSCEncryptionCertificate.PublicKeyPath -
FolderPath $certificatesPath
            Thumbprint = $DSCEncryptionCertificate.Thumbprint

            NodeName = $IPAddress
            PSDscAllowDomainUser = $true
        }
    )
}

$null = SampleConfiguration -ConfigurationData $ConfigData -OutputPath $OutputPath

```

Common Parameters

All RPS Partial Configs must define the following parameters:

1. **IPAddress** - Accessible IPAddress of the computer we'll publish DSC Configuration to.
2. **DSCEncryptionCertificate** - Information about the certificate used to encrypt the mof (configuration). The LCM is set to use this certificate and any partials that are not secured will not run on a target.
3. **OutputPath** - Location to temporarily store the mof file once its compiled.

RPS-Mapped Parameters

The main advantage of using RPS-enabled partials is that they can be dynamically built from data within RPS' CMDB. When published through RPS, RPS will supply a partial config with values for all parameters from the CMDB. This system of mapping is based on a few simple conventions and parameter attributes.

For more information on RPS-Mapped Parameters see [How to configure Rps-Mapped Parameters](#)

RPS Native Parameters

RPS supports another parameter type, `[Rps.Api.TargetItem]`. If you supply a parameter of this type, RPS will return the native **TargetItem** you are publishing to. From there, you are free to use the Rps-Api module and objects to access any data.

Partial Config Dependencies

Some Partial Configs rely on others, and to optimize publishing, they need to be compiled and published in the right order.

To indicate to RPS a dependency, use the `[DependsOn]` attribute above the param block:

```
[DependsOn(DscPartialName = 'OSCore', Mandatory = $true)]
```

- DscPartialName is the name of the partial which is depended on.
- Mandatory indicates that the dependent partial is required on this target. If `$false`, this dependency is ignored if the dependent partial isn't assigned to the target.

Importing Partial Configs into RPS

Partial Configs must be imported into RPS in order to be published via RPS. When a partial is imported, RPS parses the file and stores the metadata about the partial, its parameters and its dependencies as ResourceItems. Once imported, the partial is able to be assigned to a computer, represented as a TargetItem.

Importing a Partial

- Save Partial configs in the `\DSC\PartialConfigurations\` folder.
- Save any referenced DSC Resources to the `\DSC\Modules\` folder.
- Import the Partial via the `Import-DscPartial` cmdlet, found in the Rps-Dsc module.
- Specify a folder or file for `-Path`

Example: Import all DSC Partials

```
Set-Location $ContentStore
Import-Module .\Modules\Rps-Api
Import-Module .\Modules\Rps-Dsc

Enter-RpsSession

Import-DscPartial -Path .\DSC\PartialConfigurations
```

Testing a Partial

You can test the RPS supplied configuration data for a partial by using the `Get-DscPartialParam` cmdlet. This is useful for understanding how configuration data from Targets and Resources is passed to the DSC Partial.

The cmdlet returns a `Hashtable` that can be splatted for use directly with the `Partial`.

Example: Get Config Data for OSCore

```
Set-Location $ContentStore
Import-Module .\Modules\Rps-API
Import-Module .\Modules\Rps-Dsc

Enter-RpsSession

# import partial for OSCore
$partialPath = ".\dsc\PartialConfigurations"
Import-DscPartial -Path $partialPath
$osCorePartial = Set-RpsResourceItem -Type $Rps.ResourceTypes.DscPartial -Name OSCore

# create target item
$computer = Set-RpsTargetItem -Type $Rps.TargetTypes.Computer -Name "TestComputer1" `
    -Properties @{ ComputerName = "DEMO"; IPAddress = "10.0.0.1"; JoinDomain = "true" }

# set node properties
$containerNode = Get-RpsContainerNode -TargetItem $computer
$containerNode.SystemTimeZone = "UTC"
$containerNode.OutputPath = "c:\temp"
$containerNode.DomainName = "rps.master"
Set-RpsNode -Node $containerNode

# assign certificate
$dscCert = Set-RpsResourceItem -Type $Rps.ResourceTypes.Certificate -Name "DSCCert" `
    -Properties @{ Role = "DscEncryption"; Path = "test"; Password = "something" }
$null = New-RpsResourceAssignment -ResourceItem $dscCert -TargetItem $computer

# assign partial
$assignment = New-RpsResourceAssignment -ResourceItem $osCorePartial -TargetItem $computer

# get DSC Params
Get-DscPartialParam -PartialAssignment $assignment
```

Output

```
PS > Get-DscPartialParam -PartialAssignment $assignment
[12:55:30 INF] No credential resolved on 4ca80061-7951-4ecc-bf40-b7afcd297119: DomainAdmin
[12:55:30 INF] No value resolved on 4ca80061-7951-4ecc-bf40-b7afcd297119: ServerAdmin mapping ResourceItem in
OSCore didn't yield one match.
```

Name	Value
-----	-----
SystemTimeZone	UTC
LocalAccount	{}
DomainName	rps.master
ComputerName	DEMO
DomainAdmin	
DscEncryptionCertificate	{Path, Password, Role, Name...}
NetworkConfig	{}
IPAddress	10.0.0.1
JoinDomain	True
IsDC	False
OutputPath	c:\temp\6c94d0db-235c-418b-ad48-f40944899960\OSCore
ServerAdmin	

Assigning a Partial

Once imported, the Partial Config is saved as a `ResourceItem` in RPS. To publish a configuration from RPS, you must first assign one or more `ResourceItems` representing partials to the desired computer `TargetItem`.


```
Import-Module .\Modules\Rps-Api
$vm = New-RpsTargetItem -Type $Rps.TargetTypes.VirtualMachine -Name DemoVM1 -Properties { IPAddress =
"10.0.0.17" }
$softwarePartial = Get-RpsResourceItem -Type $Rps.ResourceTypes.DscPartial -Name "SoftwareDistribution"
New-RpsResourceAssignment -ResourceItem $softwarePartial -TargetItem $vm
```

Publishing a Partial

RPS Publishes DSC Partials using the **Publish-RpsConfiguration** runbook. This runbook is triggered from a fresh RPS Install, from an assigned Task or TaskMap, or via RPS Web.

Additional Tips

1. Using statements at the top improve readability. Without them, you must fully qualify Rps specific objects.

```
using namespace Rps.Api.PowerShell
using namespace Rps.Api.Constants

[DependsOn(DscPartialName = 'OSCore')] # instead of [Rps.Api.PowerShell.DependsOn(...)]
```

2. ConfigurationName - Use a meaningful name for the configuration. This name identifies the Partial Config in RPS and the `DependsOn` attribute.
3. For all resources you use in your partial you will need to import the module inside the configuration block. Include the `-ModuleVersion` in your import statements to avoid issues with multiple module versions.

```
Import-DscResource -ModuleName 'RPS_RPSApi' -ModuleVersion 1.0
```

1. `$certificatesPath` indicates the location where certificates are stored. Do not modify its value.
2. configData - This is the basic configuration for a partial. You can add parameters to this HashTable but this is the minimum. This sets up the mof encryption for the target.

Authoring RPS DSC Resources

Last updated on March 15, 2019.

Last Reviewed and Approved on PENDING REVIEW

The Rapid Provisioning System use Desired State Configuration to manage itself and other target computers to configure and prevent the drift from discrete states. A major piece of implementing DSC is the development and maintenance of Resources. This document will outline the steps for authoring and consuming DSC resources within RPS.

Using the Resource Within A Partial

To use a configuration of a resource within a partial you simply need to import the DSC resource.

For example, when needing to use the RPS_RpsApi resource add the below line within your configuration.

```
Configuration RPSPrivilegedAccount
{
    Import-DscResource -Module 'RPS_RpsApi'

    Node $TargetName
}
```

TERM	DEFINITION
DSC	Desired State Configuration.

Adding the resource to the RPS CMDB

The entire RPS CMDB data structure used by DSC can be found within the RPS Configuration Management (DSC) Design document. This document can be found at [\\$/Documents/Operations/RPS Configuration Management \(DSC\) Design.docx](#).

DSC resources are added to the CMDB as RPS resource items.

Resource Item

Name: <DSC Resource Name>

Type: DscResource

Path = <Path to resource>

Resources used by partials are added by name as a comma-delimited list to the partial configuration Resource Item.

Resource Item

Name: <Partial Config Name>

Type: DscPartial

PartialConfigurationName =
<PartialConfigurationName>

Path = <Path to Partial Configuration>

Resources = <Comma seperated list of
resource names>

Using the resource within the initial Install

To add a DSC resource to the RPS offline build, add the new resource to the New-RpsXmlConfiguration script within the Setup directory matching the structure described in the section above.

Introduction to DSC Pull Server

Last updated on May 27, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

A DSC Pull Server is an endpoint that you can configure clients to connect to in order to have clients pull configurations from a central location. When the client connects, it registers with the Pull Server, downloads configurations for the client, downloads the modules needed for the configuration, and, if configured, sends reports back to the Pull Server.

Plugin

The DSC Pull Server that is part of RPS is based off of [tug](#). It has been modified to be part of the RPS Web API plugin architecture and uses the RpsWebApi. This makes the plugin dependent on the RPS Web API.

The plugin uses the same port as the RPS Web API (default: 777).

The plugin is located in the ContentStore\Plugins\Rps.DscPullServer.RpsPlugin. When RPS is deployed, the RpsWebApi partial copies the plugins to ContentStore\RpsWebApi\Plugins.

The plugin uses RPS Settings to set the path for the var folder that the plugin will use. The property used to set this is DscServerVarPath. This setting is set in the RPSWebApi partial. If this setting is not used, it defaults to the root of the Rps.DscPullServer.RpsPlugin folder.

Logs generated by the Pull Server plugin will be generated in the Application Logs in the Event Viewer under Windows Logs. These logs will have a Source of Rps.Web.Api.

Report Server

The Report Server is part of the DSC Pull Server Plugin. The Report Server is an endpoint where the LCM can send the JSON reports that are created when the LCM runs a job. These are stored in the var\DscService\Reports.

LCM - Local Configuration Manager

In order for the LCM to connect to the Pull Server, there are settings that need to be configured.

Basic Settings

Other than specifying pull service endpoints/paths and partial configurations, all of the properties of the LCM are configured in a Settings block. The following setting is what is changed to set it to Pull.

PROPERTY	DESCRIPTION
RefreshMode	This has to be set to Pull.

Configuration Server Block

To define a web-based configuration server, a ConfigurationRepositoryWeb block must be created. Below are the settings that RPS sets for the Pull Server.

PROPERTY	DESCRIPTION
----------	-------------

PROPERTY	DESCRIPTION
ConfigurationNames	An array of names of configurations to be pulled by the client.
RegistrationKey	A GUID that registers the node with the pull service.
ServerURL	The URL of the configuration service.

Report Server Block

To define a report server, a ReportServerWeb block must be created. Below are the settings that RPS sets for the Report Server.

PROPERTY	DESCRIPTION
Registration Key	A GUID that identifies the node to the pull service.
ServerURL	The URL of the report service.

Certificates

A certificate is needed for all communication with the DSC Pull Server and for registration. Every time the LCM attempts to register with the Pull Server, it creates a self-signed certificate. Registration fails if the self-signed certificate is not trusted by the Pull Server.

RPS has a DscPullServer certificate that is used. This certificate is installed on the clients that will use the Pull Server. This certificate chain must be trusted by the Pull Server.

NOTE

To account for the self-signed certificate, every time the LCM settings are set for the Pull Server, RPS will remove the self-signed certificate.

RPS Settings for DSC Pull Server

Last updated on May 27, 2021.

Last Reviewed and Approved on PENDING REVIEW

RPS Required Settings

Node Properties

PROPERTY	DESCRIPTION
DscServerVarPath	Path to where the var folder is located. This folder holds registrations, configurations, resources, and reports.
DscConfigurationUrl	URL to the DSC Pull Server. Value is used to set the Configuration Server Block for LCM on the targets under the node.
DscReportServerUrl	URL to the DSC Report Server. Value is used to set the Report Server Block for LCM on the targets under the node.

TargetItem Properties

PROPERTY	DEFAULT	DESCRIPTION	VALUES
DscRefreshMode	Push, if property does not exist.	DscRefreshMode that the LCM is set to.	Push, Pull

ResourceItems

Each RPS Target that has the DscRefreshMode set to Pull will need a ResourceItem with Type Certificate with the role of DscPullServer.

At deployment, or when a target is changed to Pull, a ResourceItem with Type Certificate will be generated with SigningType RpsSigned. In addition, another ResourceItem with Type Certificate will be created that has SigningType CASigned. The second ResourceItem is used to enable certificate rolling from RpsSigned to CASigned for the DscPullServer certificate. To learn more about certificate rolling, see: [Rolling Certificates](#)

DscPullServer

PROPERTY	REQUIREMENT	EXAMPLE
SubjectName	Must be the URL of the Pull Server.	CN=https://APP.rps.local:777/DscPullServer/v1.0/DscPullServer
SubjectAlternativeName	Must be the URL of the Pull Server.	https://APP.rps.local:777/DscPullServer/v1.0/DscPullServer
Role	Must be DscPullServer.	DscPullServer

Runbooks and TaskMaps for DSC Pull Server

Last updated on May 27, 2021.

Last Reviewed and Approved on PENDING REVIEW

DSC Pull Server Runbooks

The following runbooks have some functionality that the DSC Pull Server depends on.

Compress-DscModules

The Compress-DscModules runbook finds all the modules in the ContentStore\DSC\Modules and then compresses them to the location specified at DSCServerVarPath. It names the zip files according to the naming scheme needed by the LCM.

Switch-DscClientRefreshMode

The Switch-DscClientRefreshMode runbook will change the DscRefreshMode property on the TargetItem. If it is set to Push, it will change it to Pull. If it is Pull, it will change it to Push.

If the TargetItem does not have a DscRefreshMode property, this runbook will set the DscRefreshMode on that target to 'Push'.

Update-DscRegistrationFile

The Update-DscRegistrationFile runbook will update the Registration.txt file that is used to verify a client's Registration Key during the registration process. It looks at the CMDB, finds all computers (physical or virtual) that have DscRefreshMode set to Pull, and updates the file. The file is located from the DscServerVarPath → DscService\Authz.

Install-DscPullClientCertificate

The Install-DscPullClientCertificate runbook installs the certificate needed for DSC Client Auth for the Pull Server on Windows targets. If a ResourceItem Type certificate and role of DscPullServer does not exist, it will create the certificate prior to installing it on the target.

Install-LinuxDscPullClientCertificate

The Install-LinuxDscPullClientCertificate runbook installs the certificate needed for DSC Client Auth for the Pull Server on Linux targets. If a ResourceItem Type certificate and role of DscPullServer does not exist, it will create the certificate prior to installing it on the target.

Publish-DscConfiguration

The Publish-DscConfiguration runbook will create the MOF file(s) for the target by looking up all the partials assigned to the target in the CMDB. If the target has DscRefreshmode set to Pull, it will merge the partial configuration MOFs into a single MOF named according to the target IP and place in the DscServerVarPath → DscService\Configurations. Next, this runbook will set the LCMConfiguration Server Block to have the configuration name that was created, and apply to the target.

Set-DscReportServer

The Set-DscReportServer runbook will set the Report Server Block on the targets LCM. This enables reports to be sent to the DSC Pull Server and saved to DscServerVarPath → DscService\Reports.

NOTE

A property on the node, `DscReportServerUrl`, is required to set the ReportServer on a target. If this property is NOT on the node, it will remove the ReportServer from a target.

DSC Pull Server TaskMaps

These taskmaps were created to streamline the process of changing from Push to Pull, and Pull to Push, for DSC. To see how to assign taskmaps, reference the Task Map Assignment [RPS Tasking Guide](#).

Set-DscPushToPull

The Set-DscPushToPull taskmap will make all the changes required to change a target from Push to Pull.

The following runbooks make up this taskmap:

1. Compress-DscModules
2. Switch-DscClientRefreshMode - Filter {DscRefreshMode = Push}
3. Update-DscRegistrationFile
4. Install-DscPullClientCertificate - Filter {OSType = Windows}
5. Install-LinuxDscPullClientCertificate - Filter {OSType = Linux}
6. Publish-DscConfiguration
7. Set-DscReportServer

Set-DscPullToPush

The Set-DscPullToPush taskmap will make all the changes required to change a target from Pull to Push.

The following runbooks make up this taskmap:

1. Switch-DscClientRefreshMode - Filter {DscRefreshMode = Pull}
2. Update-DscRegistrationFile
3. Copy-DscModules
4. Publish-DscConfiguration
5. Set-DscReportServer

Create a Host Through Rapid Provisioning System (RPS)

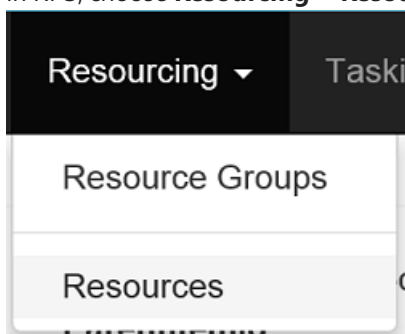
Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

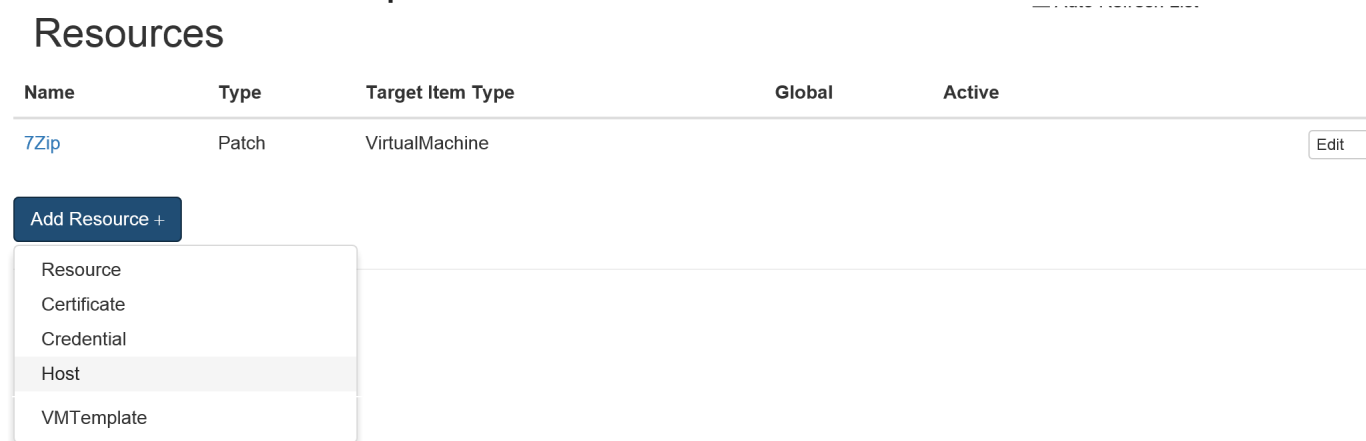
This guide shows the process to create a new Host for RPS to use during Virtual Machine deployment.

Create a Resource Item that represents the Host

1. Open a web browser to the RPS Website, for example: <https://localhost:8080>
2. In RPS, choose **Resourcing** > **Resources**



3. Choose **Add Resource** > **VMTemplate**



4. Set the requested information, and then click **Save**

VMTEMPLATE SETTINGS	DESCRIPTIONS
Name (Required)	The desired name for Host in RPS
Type (Required)	Must be Host . Pre-populated
ComputerName (Required)	The name of the server acting as the Virtual Machine Host.
HostType (Required)	The type of host, e.g. Hyper-V, ESXi
Path (Required)	The path to save the virtual machines

i NOTE

Click **Add Property** to add custom fields for your Credential.

More Resources

- [Create a Hyper-V Virtual Machine Through Rapid Provisioning System \(RPS\)](#)
- [Create RPS Credentials Through Rapid Provisioning System \(RPS\)](#)
- [Create a Virtual Machine Template Through Rapid Provisioning System \(RPS\)](#)

Create RPS Credentials through Rapid Provisioning System (RPS)

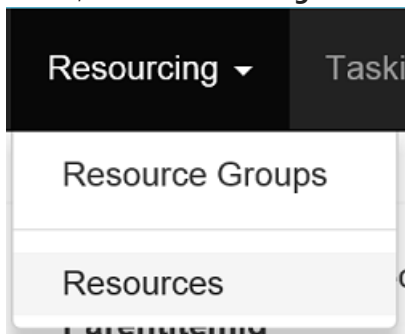
Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

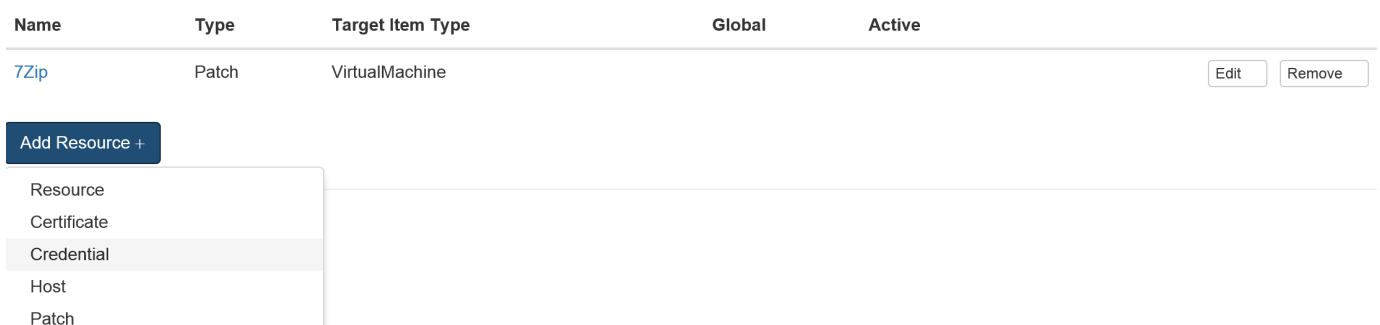
This guide shows the process to create new credentials to use in RPS.

Create a Resource Item that Represent the Credential

1. Open a web browser to the RPS Website, for example: <https://localhost:8080>
2. In RPS, choose **Resourcing** > **Resources**



3. Choose **Add Resource** > **Credential**



4. Set the requested information, and then click **Save**

VMTEMPLATE SETTINGS	DESCRIPTIONS
Name (Required)	The name you want to give the Credential in RPS
Type (Required)	Must be Credential . Pre-populated
UserName (Required)	The UserName of the credential
Password (Required)	The Password of the credential. This password can be automatically generated by pressing the Generate Password button.
Roles (Required)	The role the credential will be used for. Multiple roles can be added, separated by a ' '

NOTE

Click **Add Property** to add custom fields for your Credential.

More Resources

- [Create a Host Through Rapid Provisioning System \(RPS\)](#)
- [Create a Hyper-V Virtual Machine Through Rapid Provisioning System \(RPS\)](#)
- [Create a Virtual Machine Template Through Rapid Provisioning System \(RPS\)](#)

Create a Virtual Machine Template Through Rapid Provisioning System (RPS)

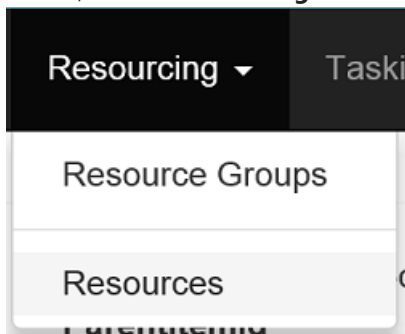
Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

This guide shows the process to create a new Virtual Machine template for RPS to use during Virtual Machine deployment.

Create a Resource Item that Represents the Virtual Machine

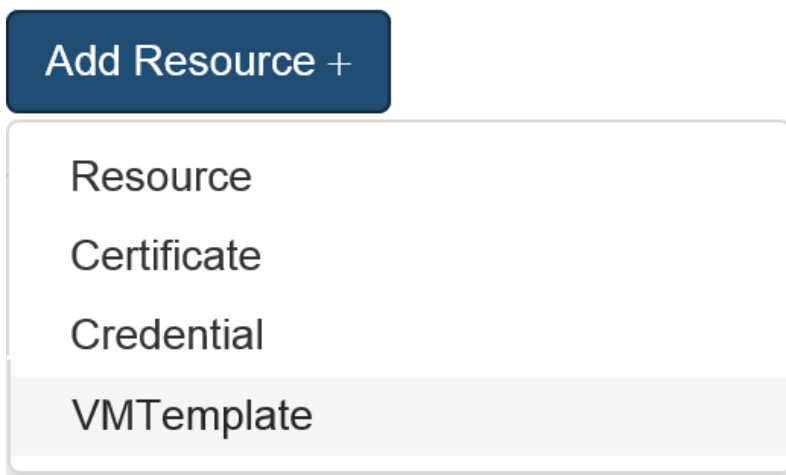
1. Open a web browser to the RPS Website, for example: <https://localhost:8080>
2. In RPS, choose **Resourcing** > **Resources**



3. Choose **Add Resource** > **VMTemplate**

Resources

Name	Type	Target Item Type
7Zip	Patch	VirtualMachine



4. Set the requested information, and then click **Save**.

VMTEMPLATE SETTINGS	DESCRIPTIONS
Name (Required)	The desired name for the VMTemplate in RPS
Type (Required)	Must be VMTemplate . Pre-populated
Path (Required)	The path to the .vhdx file, relative to Content Store

NOTE

Click **Add Property** to add custom fields for your Virtual Machine Template.

More Resources

- [Create a Host Through Rapid Provisioning System \(RPS\)](#)
- [Create RPS Credentials Through Rapid Provisioning System \(RPS\)](#)
- [Create a Hyper-V Virtual Machine Through Rapid Provisioning System \(RPS\)](#)

Create a Hyper-V Virtual Machine through Rapid Provisioning System (RPS)

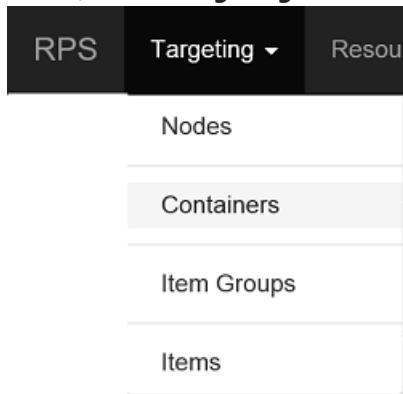
Last updated on March 26, 2021.

Last Reviewed and Approved on PENDING REVIEW

This guide shows the process to create a new Target Item to deploy as a Hyper-V Virtual Machine.

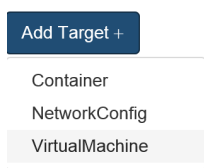
Create a Target Item that represents the Virtual Machine

1. Open a web browser to the RPS Website, for example: <https://localhost:8080>
2. In RPS, choose **Targeting > Containers**



3. Choose **Add Target > Virtual Machine Containers**

Name	Type	# Of Children	Pending Actions	Errors	Active
AD	VirtualMachine	1			Edit Remove
CH	VirtualMachine	1			Edit Remove
Child	VirtualMachine	1			Edit Remove
DB	VirtualMachine	1			Edit Remove
SMA	VirtualMachine	1			Edit Remove



4. Set the requested information, and then click **Save**

VIRTUAL MACHINE SETTINGS	DESCRIPTIONS
Name (Required)	The desired name to give the Target Item (Virtual Machine) in RPS
Type (Required)	Must be VirtualMachine . Pre-populated
ComputerName (Required)	The name of the computer as it is in Active Directory
MemoryMB (Required)	The amount of memory the machine needs in MB
OSType (Required)	The Virtual Machines operating system, e.g. Windows

VIRTUAL MACHINE SETTINGS	DESCRIPTIONS
OSVersion (Required)	The version on the operating system, e.g. 8.1 for Windows Server 2012 R2
Architecture (Required)	The architecture of the Virtual Machine, e.g. x86, x64
IsCDN	[Boolean] Set to true if this Virtual Machine will have RPS role Content Delivery Network
IsTaskManagement	[Boolean] Set to true is this Virtual Machine will have RPS role Task Management Service
IsDB	[Boolean] Set to true if this Virtual Machine will have RPS role PostgreSQL Database
IsDC	[Boolean] Set to true if this Virtual Machine will have RPS role Active Directory Domain Controller

NOTE

Click **Add Property** to add custom fields for the Virtual Machine.

Create a child Target Item that represents the Network Configuration

1. Click on the new Virtual Machine
2. Under All Items, choose **Add Child Target Item**.

All Items 0				Details
Name	Type	IsActive	ID	
				Add Child Target Item +

3. Set the requested information, then click **Save**.

NOTE

To add properties, click **Add Property** button. The required fields must be added to create a New Hyper-V Virtual Machine. The next update will use a template to pre-populate the fields.

NETWORK CONFIGURATION SETTINGS	DESCRIPTIONS
Name (Required)	The desired name to give the Network Adapter in RPS
Type (Required)	Must be NetworkConfiguration
Alias (Required)	The name of the Network Adapter on the Host you are creating the Virtual Machine
IpAddress (Required)	The IP Address of the Virtual Machine
Subnet (Required)	The subnet of the Virtual Machine
DnsServer (Required)	The IP Address of the DNS Server
MacAddress	The desired MAC Address for the Virtual Machine. If left blank, the Virtual Machine will get a dynamic MAC Address

NETWORK CONFIGURATION SETTINGS	DESCRIPTIONS
DHCP	The IP Address of the DHCP server

NOTE

Click **Add Property** to add custom fields for your Virtual Machine.

Add the Hyper-V Virtual Machine Template

- Under **Resource Assignments**, click **Add Resource Assignment**.

Resource Assignments								Details
Resource Name	Type	Target Item Type	Resource Group	Target Group	Global	State	Status Changes	
								Add Resource Assignment

- Click on the **Resource Assignment** dropdown, choose the Resource Item that represents the Virtual Machine Template. Click **Save**.

NOTE

The Resource Item will have a type of **VMTemplate**. If the desired Virtual Machine template is not visible, then see: [Create a Virtual Machine Template Through RPS Guide](#).

Add the desired Hyper-v Host

- Under **Resource Assignments**, click **Add Resource Assignment**.

Resource Assignments								Details
Resource Name	Type	Target Item Type	Resource Group	Target Group	Global	State	Status Changes	
								Add Resource Assignment

- Click on **Resource Assignment** drop down, choose the Resource Item that represents the Host. Click **Save**.

NOTE

Resource Item will have a type of **Host**. If you do not see the Host you want to use then you can create your own following the [How to Create a Host Guide](#).

Add Local Administrator Credentials

- Under **Resource Assignments**, click **Add Resource Assignment**.

Resource Assignments								Details
Resource Name	Type	Target Item Type	Resource Group	Target Group	Global	State	Status Changes	
								Add Resource Assignment

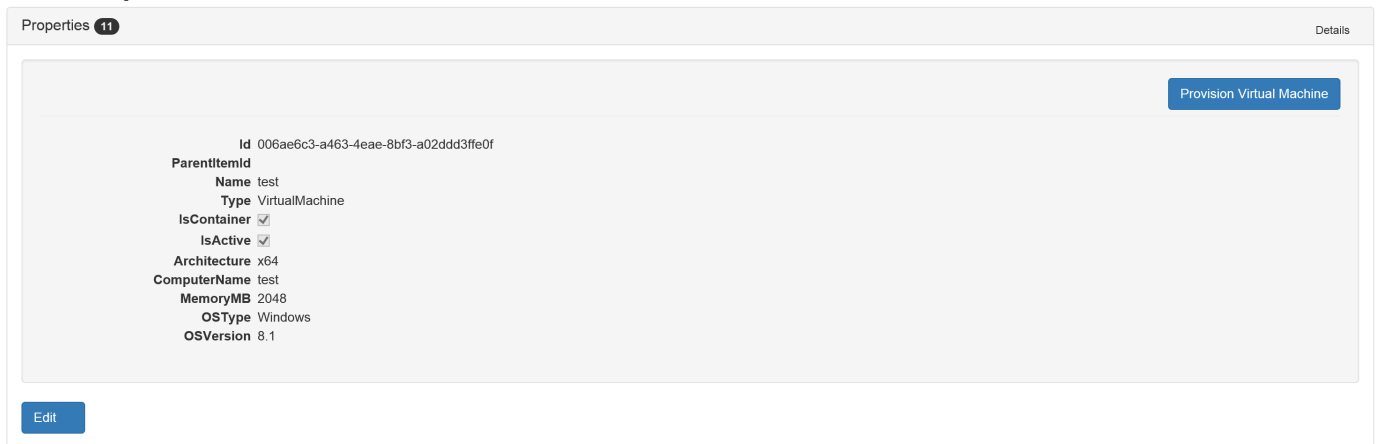
- Click on **Resource Assignment** drop down, choose the Resource Item that represents the Local Admin Credentials. Click **Save**

NOTE

Resource Item will have a type of **Credential**. If you do not see the Credential you want to use then you can create your own following the [Create RPS Credentials Guide](#).

Provision the Virtual Machine

1. Under **Properties**, click **Provision Virtual Machine**.



Properties **11** Details

[Provision Virtual Machine](#)

Id 006ae6c3-a463-4eae-8bf3-a02ddd3ffe0f

ParentItemId

Name test

Type VirtualMachine

IsContainer

IsActive

Architecture x64

ComputerName test

MemoryMB 2048

OSType Windows

OSVersion 8.1

[Edit](#)

NOTE

This action creates a new Task to create the VM on the assigned Hypervisor Host. RPS will begin processing the task in the background. Refresh the view to see the status of the task.

More Resources

- [Create a Host Through Rapid Provisioning System \(RPS\)](#)
- [Create RPS Credentials Through Rapid Provisioning System \(RPS\)](#)
- [Create a Virtual Machine Template Through Rapid Provisioning System \(RPS\)](#)

RPS Building iPXE ROMs

Last updated on March 15, 2019.

Last Reviewed and Approved on PENDING REVIEW

This guide shows how to configure an iPXE ROM with the specific options/features needed to be used within RPS.

Process Overview

The following steps are required to build a iPXE ROM that is trusted by the RPS root certificate:

- Install prerequisites
- Download iPXE source
- Build ROM

Prerequisites

- iPXE source
- Admin Workstation
- Linux workstation such as Ubuntu OR WSL (Windows Subsystem for Linux)
- Linux packages
- GCC
- binutils
- make
- syslinux (required for building ISOs)
- genisoimage (required for building ISOs)
- liblzma

- RPS public root certificate, Base64 encoded

NOTE

Most of the tools required to build an iPXE ROM are Linux based and can only be executed from Bash (Unix Shell). However, a Unix operating system is not required. The process in this document has been completed leveraging WSL (Windows Subsystem for Linux). WSL is available on a machine running 64-bit version of Windows 10 Anniversary Update build 14393 or later. If the development machine that the ROM is being built from does not have the required prerequisites an internet connection will be required to install them.

Installing the Tools in Bash

1. Update the package list, **sudo apt update**
2. Install GCC, **sudo apt install gcc**
3. Install binutils, **sudo apt install binutils**
4. Install make, **sudo apt install make**
5. Install syslinux, **sudo apt install syslinux**
6. Install genisoimage, **sudo apt install genisoimage**
7. Install liblzma, **sudo apt install liblzma-dev**
8. Install git, **sudo apt install git**

Download iPXE source and Build ROM that Trusts the RPS CA Root Certificate

1. Navigate to the directory the source will be downloaded to and clone iPXE repository: **git clone git://git.ipxe.org/ipxe.git**

2. After source is downloaded navigate to ipxe/src: **cd ipxe/src**
3. To build a ROM that trusts the RPS root certificate run the following:

```
make bin-x86_64-efi/ipxe.efi TRUST=<path to certificate> CERT=<path to certificate>
```

b. Here is an example of building an iPXE, EFI compatible, ROM with the RPS certificate in the local folder.

```
make bin-x86_64-efi/ipxe.efi TRUST=RPSbase64Ca.cer CERT=RPSbase64Ca.cer
```

NOTE

The ROM will be in the bin-x86_64-efi folder

iPXE ROM Build Command Examples

All the examples are executed from the ipxe/src folder.

1. Create an iPXE bootable ISO that trust the RPS root cert

```
make bin/ipxe.iso TRUST=RPSbase64Ca.cer CERT=RPSbase64Ca.
```

2. Create iPXE ROMs for all the compatible ESXi network adapters

```
make bin/8086100f.mrom bin/808610d3.mrom bin/10222000.rom bin/15ad07b0.  
certutil.exe -encode CaRootCert.cer base64Ca.cer
```

NOTE

Ensure the RPS root cert is Base64 encoded. If not run the following command from a Windows environment:

More Resources

- [RPS PXE Document](#)
- [Configuring ESXi VMs to Use iPXE](#)

RPS PXE

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

A major function of the Rapid Provisioning System (RPS) is the ability for system users to perform bare metal provisioning of computing devices through the use of a Pre-Boot Execution Environment (PXE). The decision to use a basic installation of Windows Deployment Services (WDS) and Microsoft Deployment Toolkit (MDT) was made to allow for the smallest possible footprint, as well as for ease of use.

Additionally, many deployed environments that utilize RPS are very unpowered and over-provisioned, where the use of a major component such as System Center Configuration Manager (SCCM) would be impractical.

MDT and creating a WIM

Two components are required for setting up the Microsoft Deployment Toolkit (MDT) environment: 1) the Deployment Toolkit itself, and 2) the Windows Assessment and Deployment Kit (ADK), which has all of the necessary tools required to allow a user to create and customize a deployment environment using MDT.

- The MDT can be downloaded from the following link: <https://www.microsoft.com/en-us/download/details.aspx?id=50407>
- The ADK can be downloaded from the following link: <https://www.microsoft.com/en-us/download/details.aspx?id=39982>

The ADK needs to be installed first, and then MDT can be installed. Once it is installed, open the program called "Deployment Workbench". The interface looks like the following:

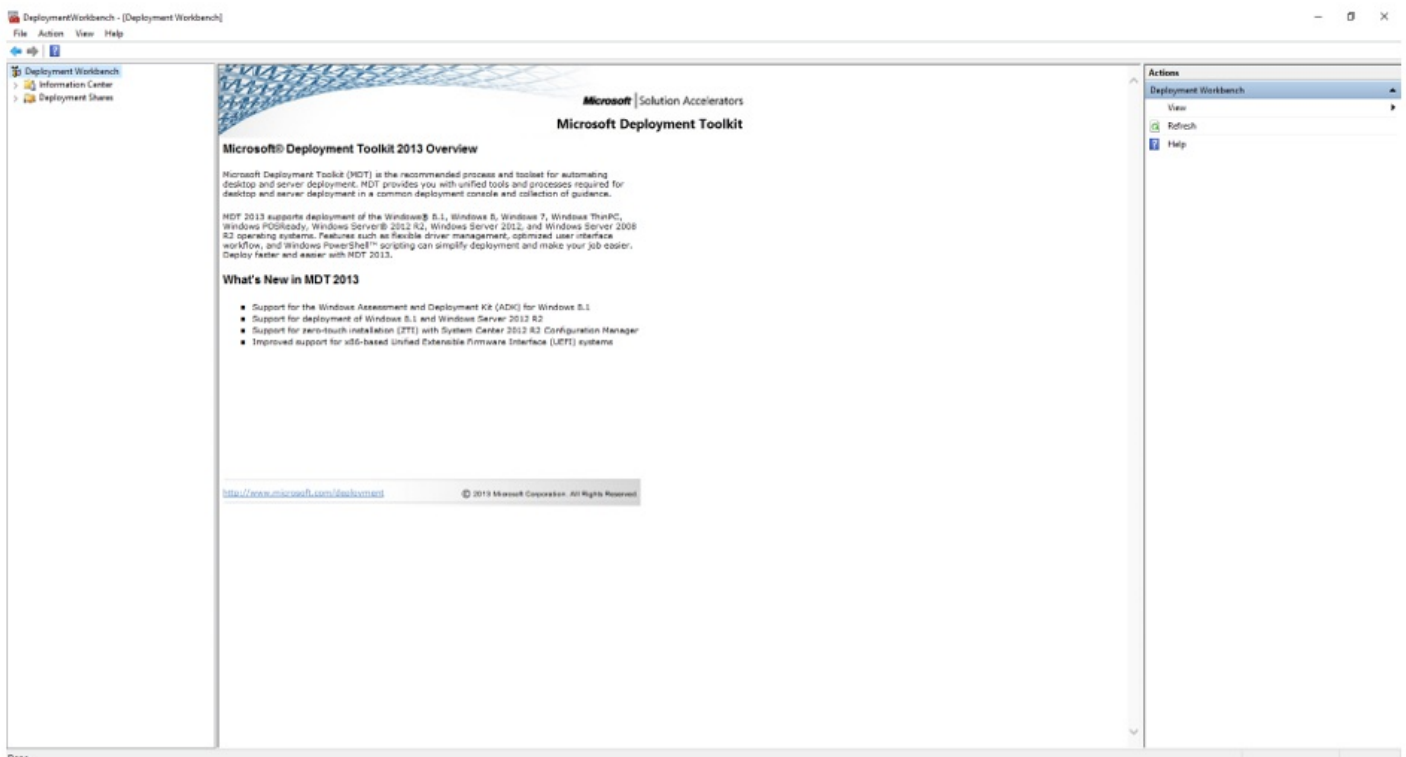


Figure 1 Deployment Workbench

Creating a Deployment Share

The first step in setting up the environment is to create a new deployment share. To do this, right click on "Deployment Shares" in the left pane and select "New Deployment Share". The following wizard will appear:

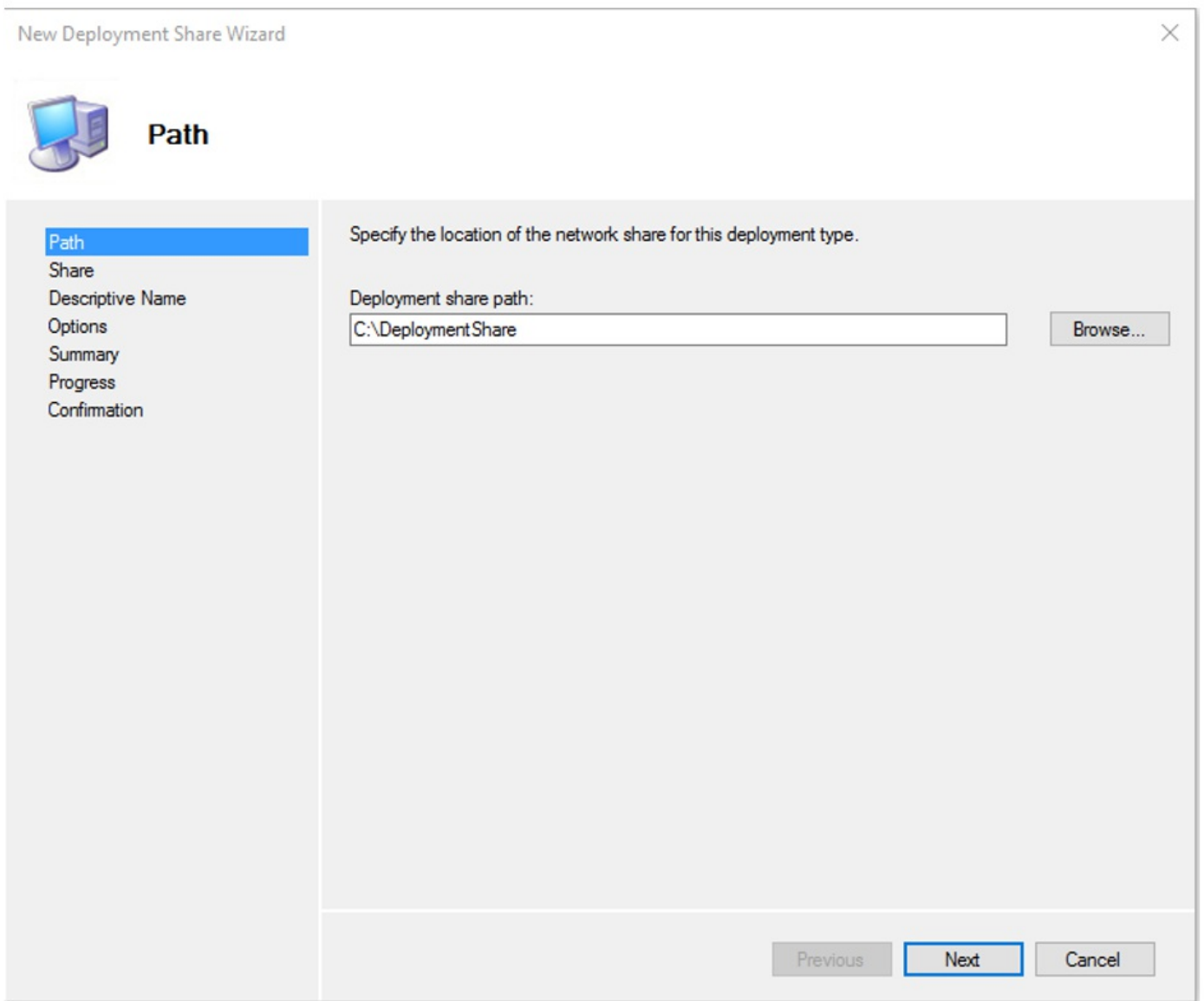


Figure 2 New Deployment Share

Continue through the wizard, and at the options screen, uncheck all of the options. The summary screen before the Deployment Share gets created should look like this:



Summary

Path
Share
Descriptive Name
Options
Summary
Progress
Confirmation

All of the necessary details have been specified. Please review the values below.

Details:

Path:	C:\DeploymentShare1
Upgrade:	False
ShareName:	DeploymentShare\$
Description:	MDT Deployment Share
Ask about Backup:	False
Ask for Product Key:	False
Ask for Admin Password:	*****
Ask about Image Capture:	False
Ask about BitLocker:	False

Click next to execute the requested action.

Previous Next Cancel

Figure 3 Deployment Share Options

Customizing a Deployment Share

1. Once the Deployment Share gets created, it will appear in the left pane under "Deployment Shares." To customize it, right-click your deployment share, and click properties. The deployment Share Properties Screen will appear.

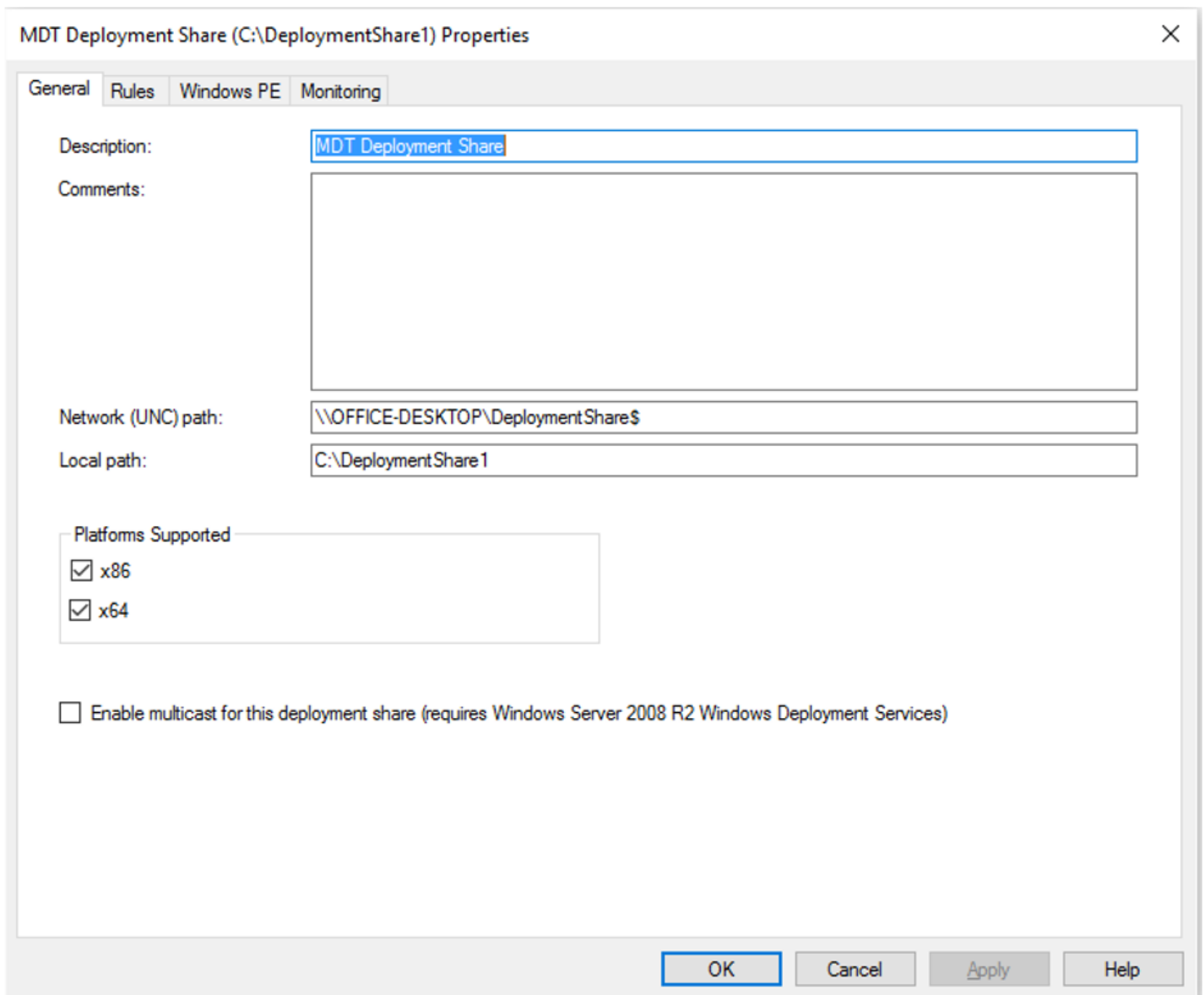
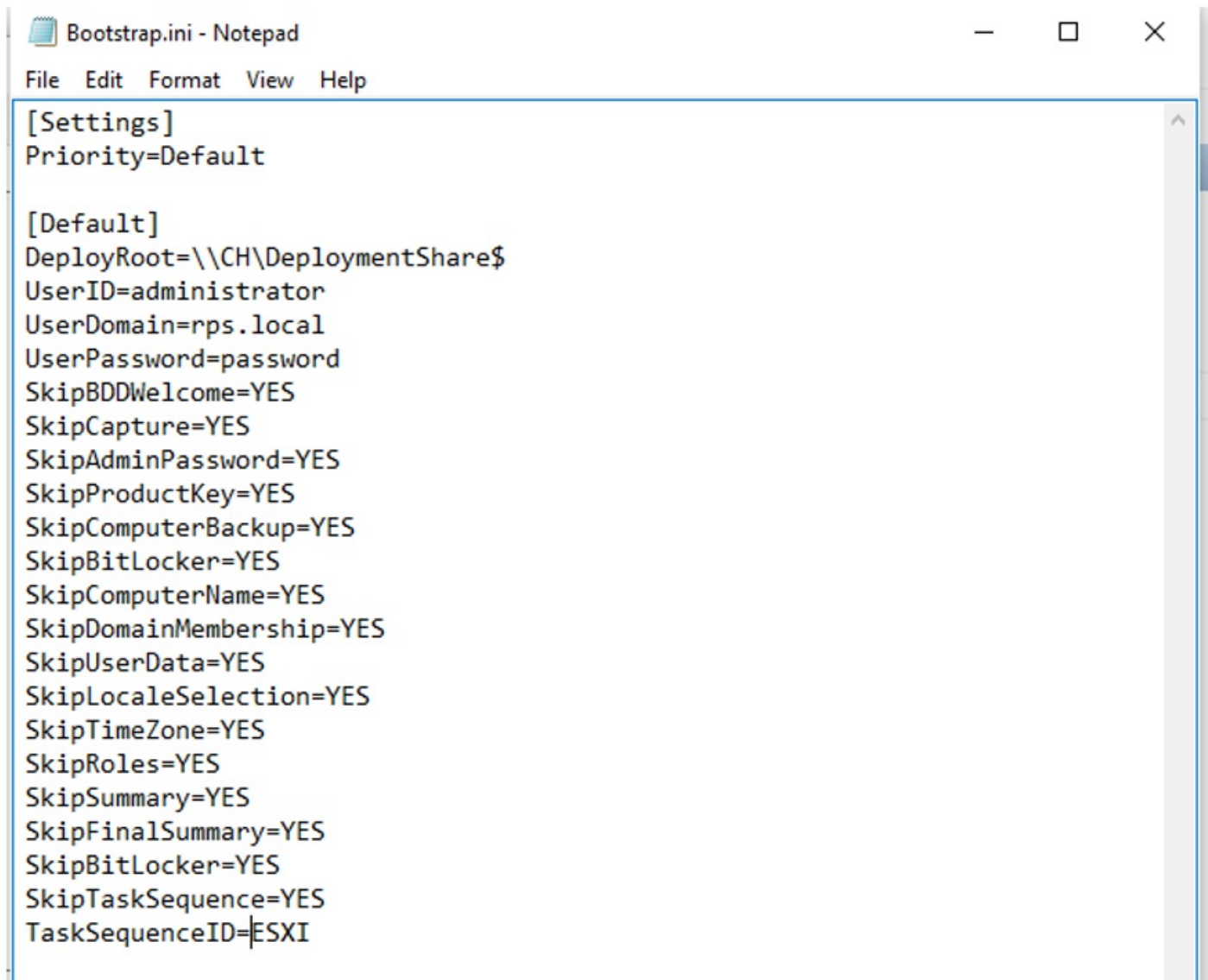


Figure 4 Deployment Share Properties

2. On the general tab, select which platforms you will support. Generally, you only need to select "x86" because then you only need to maintain one image, and it will work both on x86 and x64 platforms. On the Rules tab, click the "Edit Bootstrap.ini" button, and edit the "DeployRoot" path to be the path where the deployment share will exist in your deployment environment. In the same file, edit the UserID field to be the user which will access the deployment share from WinPE, the UserDomain to be the User's domain, the UserPassword to be the User's password, and the TaskSequenceID to be the ID of the Task Sequence that will be used in order to run deployments. This will be covered in a later step.



```
[Settings]
Priority=Default

[Default]
DeployRoot=\\CH\DeploymentShare$
UserID=administrator
UserDomain=rps.local
UserPassword=password
SkipBDDWelcome=YES
SkipCapture=YES
SkipAdminPassword=YES
SkipProductKey=YES
SkipComputerBackup=YES
SkipBitLocker=YES
SkipComputerName=YES
SkipDomainMembership=YES
SkipUserData=YES
SkipLocaleSelection=YES
SkipTimeZone=YES
SkipRoles=YES
SkipSummary=YES
SkipFinalSummary=YES
SkipBitLocker=YES
SkipTaskSequence=YES
TaskSequenceID=ESXI
```

Figure 5 Bootstrap.ini

3. Save the changes you make to the Bootstrap.ini file, and continue on to the Windows PE tab. Navigate to the features tab, and select the DISM Cmdlets, .NET Framework, and Windows PowerShell check boxes.

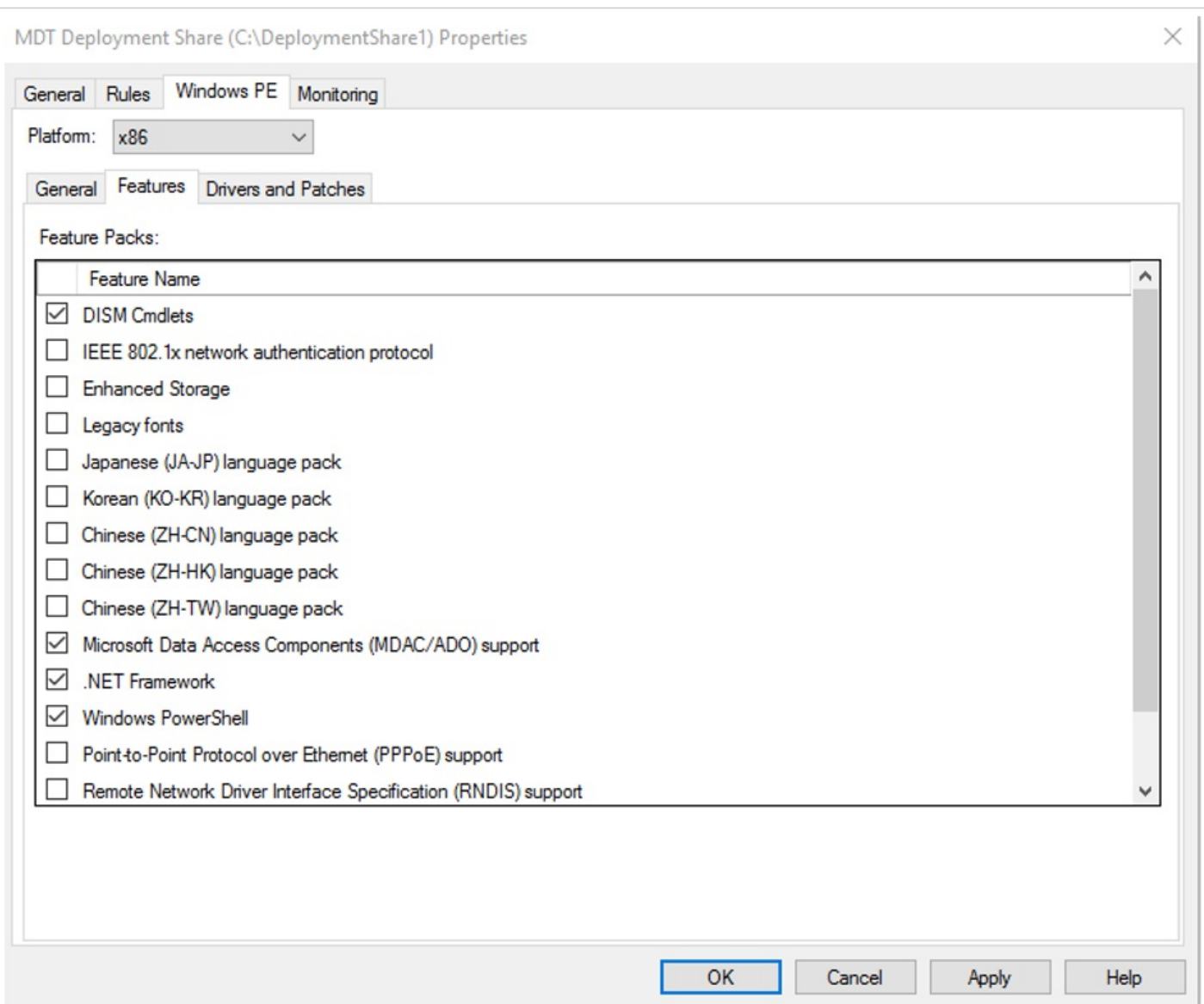


Figure 6 Windows PE Features

Generating Boot Media

Prior to completing these steps for generating boot media, you need to perform steps 1-3 for customizing a deployment share.

4. The last step is to generate the Windows Imaging Format (WIM) Boot media that will be used to run deployments. To do this, right click on your deployment share, select "Update Deployment Share", select the "Completely regenerate boot media" Radio button, and complete the wizard. The final output will look like the following:



Confirmation

Options
Summary
Progress
Confirmation



The process completed successfully.

```

=== Making sure the deployment share has the latest x86 tools ===

=== Processing LiteTouchPE (x86) boot image ===

Building requested boot image profile.
Determining if any changes have been made in the boot image configuration.
No existing boot image profile found for platform x86 so a new image will be created.
Calculating hashes for requested content.
Changes have been made, boot image will be updated.
Windows PE WIM C:\Program Files (x86)\Windows Kits\8.1\Assessment and Deployment Kit\Windo
WIM file mounted.
Set Windows PE system root.
Set Windows PE scratch space.
Added component winpe-hta
Added component winpe-scripting
Added component winpe-wmi
Added component winpe-securestartup
Added component winpe-fmapi
Added component winpe-mdac
Copy: C:\DeploymentShare1\Control\Bootstrap.ini to C:\Users\johny\AppData\Local\Temp\MDTUp
  
```

Save Output...

View Script

Previous

Finish

Cancel

Figure 7 Generating Boot Media

5. The WIM that is generated will be located in your deployment share folder, under the "Boot" Folder. The last step is that this WIM that is generated needs to be imported into Windows Deployment Services (WDS) on the server that is going to be responsible for running deployments. To do this, open the WDS console, expand Servers, expand your WDS server, right click on Boot Images, select "Import Boot Image", and then navigate to your deployment share and select the WIM that you created.

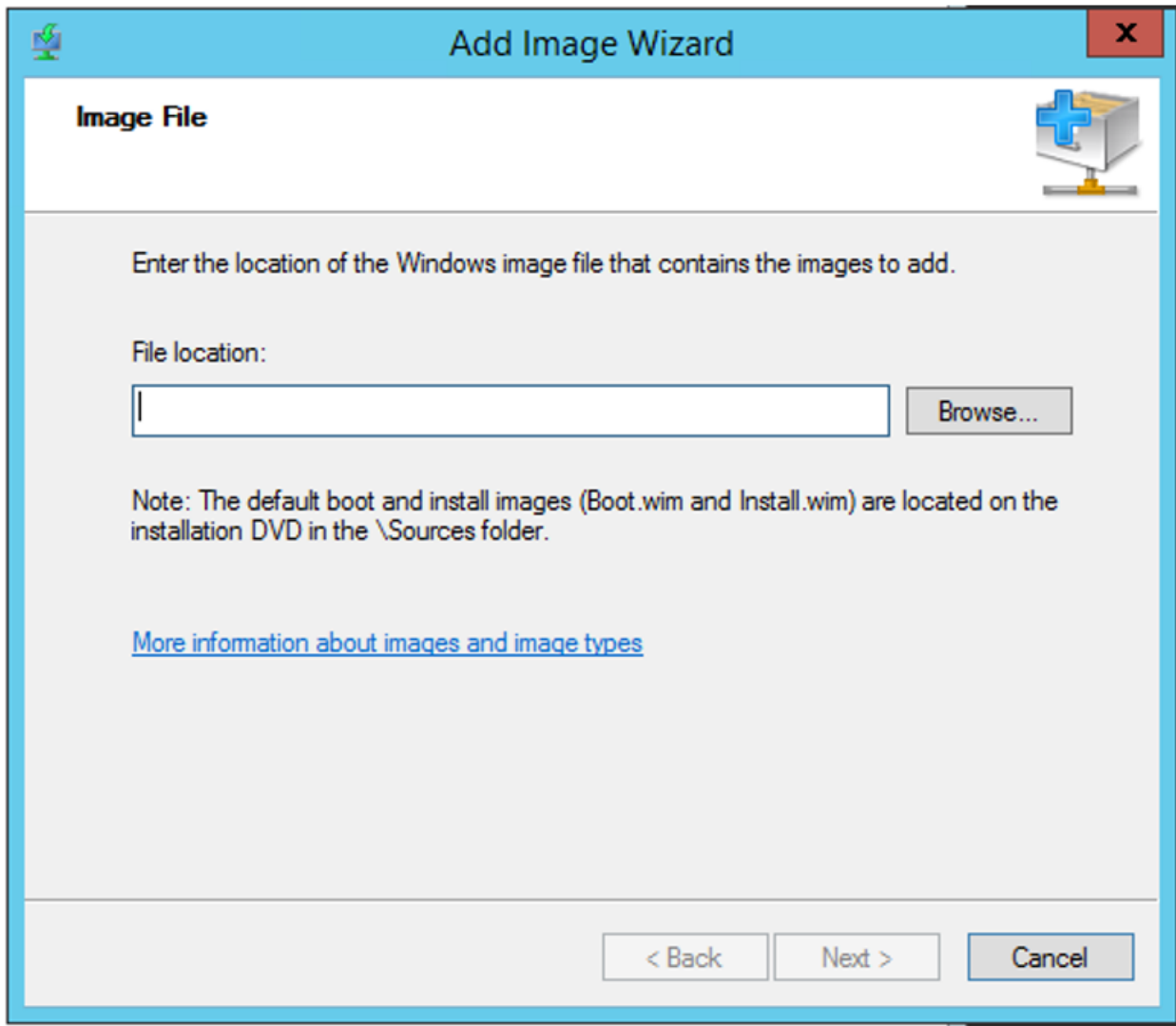



Figure 8 WIM Import

Adding Drivers to WIM

There are cases where the default network and storage drivers built into the WIM are not sufficient for running deployments, and the client machine is either unable to communicate to the deployment server over the LAN or cannot access its storage. In these cases, additional drivers need to be injected into the WIM so that connectivity can be restored.

Adding Drivers to MDT

In order to inject drivers into your WIM, you need to have the drivers downloaded and available on the machine where MDT is installed. To inject the drivers, open the deployment workbench, expand your deployment share, right-click on "Out-of-Box Drivers", and then select "Import Drivers." The following wizard will appear:



Specify Directory

The specified folder and all subfolders will be scanned looking for drivers. Each directory containing a driver (found by looking for INF files) will be added.

Driver source directory:

Import drivers even if they are duplicates of an existing driver.

Specify Directory
Summary
Progress
Confirmation

Previous Next Cancel

Figure 9 Import Drivers

Select the browse button, navigate to the folder where your third-party drivers are located, select the folder and then complete the wizard. If the drivers are imported successfully, they will appear in the middle pane of the deployment workbench as follows:

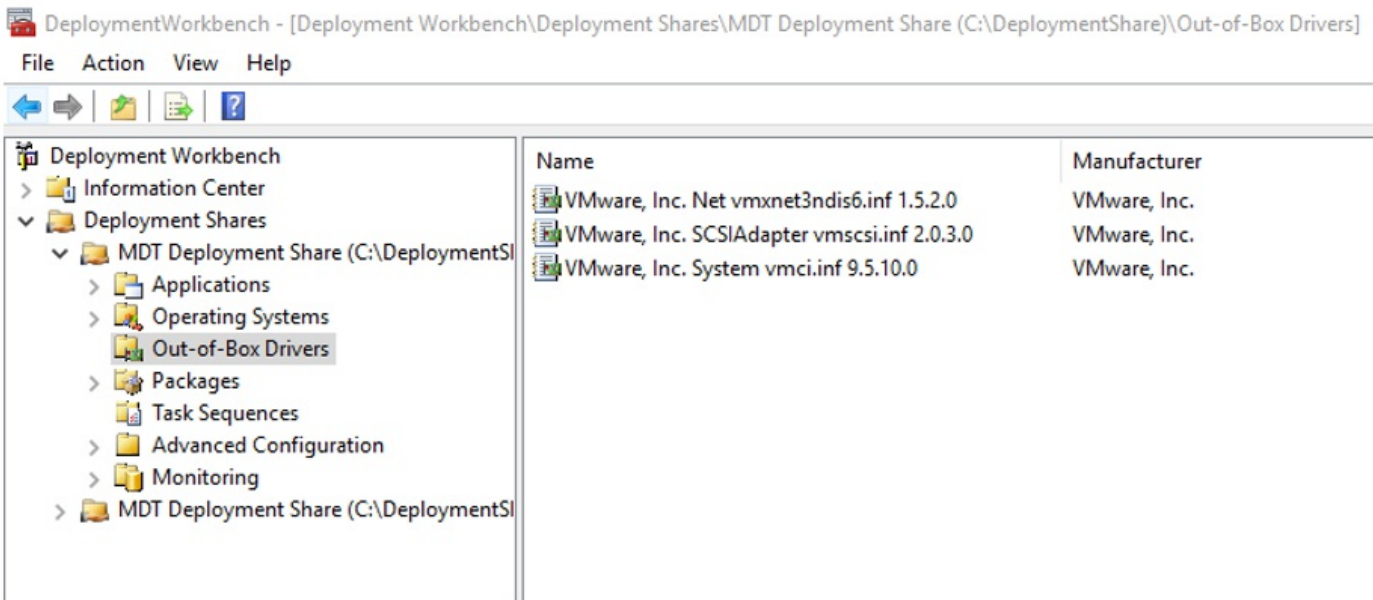


Figure 10 Third Party Driver

Configuring MDT to Use Third Party Drivers

This process just makes the drivers available in your deployment workbench but has not yet injected the drivers into the WIM. To do so, right-click on your deployment share, select Properties, and navigate to the Windows PE tab, and then the Drivers and Patches tab. Select which type of drivers you want added to the WIM (you can either select all drivers that are available in your deployment workbench, or specify a specific type over driver, and then click OK.

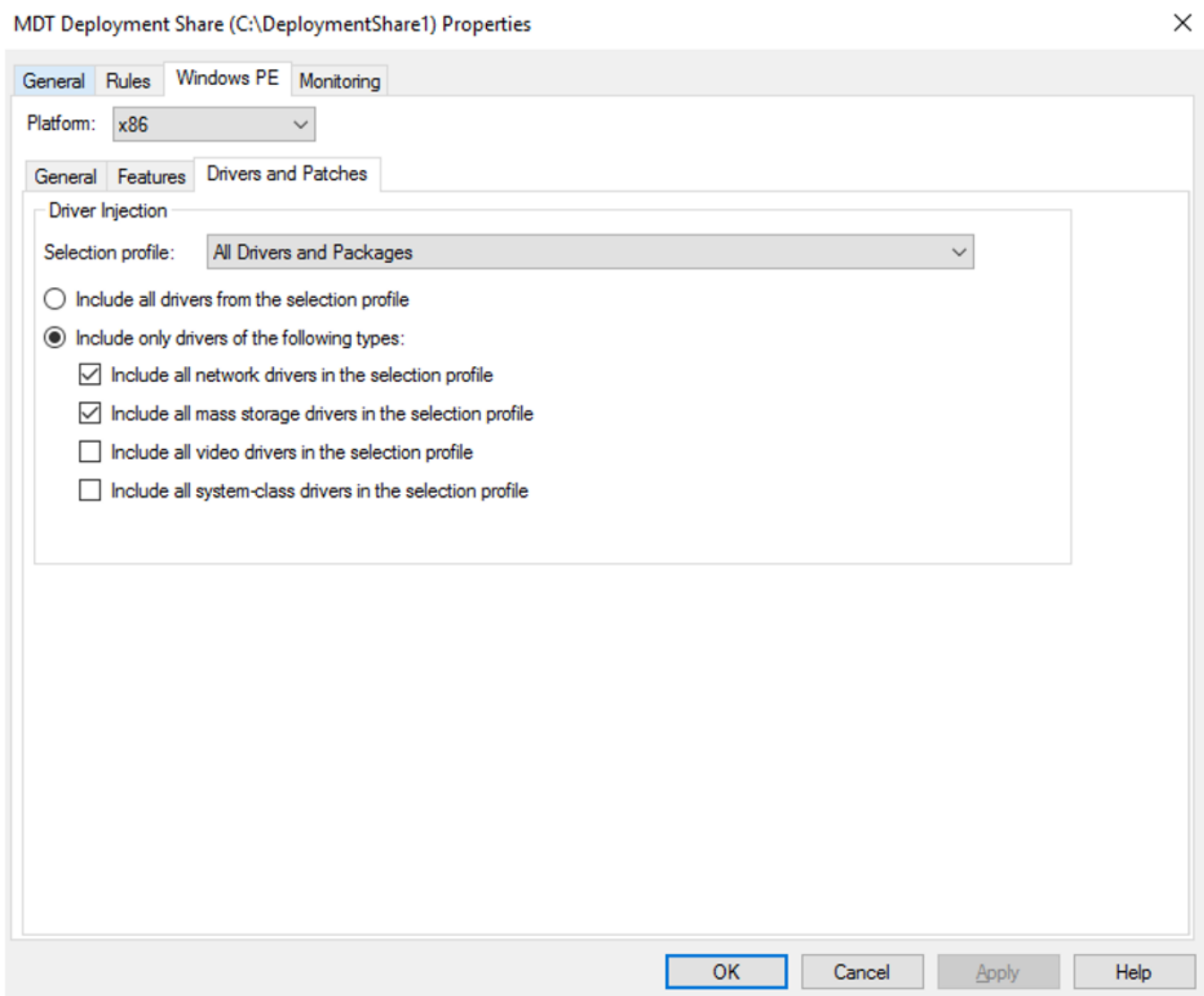


Figure 11 Driver Settings

Once you determine your settings, you need to regenerate the WIM so that the newly added drivers are injected into it. To do so, follow the process documented above in the WIM creation process.

Adding Additional Components to WIM

There are cases where additional components need to be added to your WIM, that cannot be done using the deployment workbench. One such case is adding the environment variable connection string to the registry of the WIM, so that clients that are in the Windows Pre-Boot Environment (WinPE), can import the RPS DAC dll and interface with the CMDB.

Mounting the WIM

To do this, create an empty folder on the server with MDT installed, in a location that is easily accessible. A good example of this would be something like "C:\mount". Next, open an administrator command prompt window, and enter the following command:

```
Cd C:\Program Files (x86)\Windows Kits\8.1\Assessment and Deployment Kit\Deployment Tools\x86\DISM
```

Once in the DISM directory, issue the following command:

```
dism /mount-image /imagefile:C:\DeploymentShare\Boot\LiteTouchPE_x86.wim /index:1 /mountdir:C:\mount
```

Make sure to replace the imagefile path to the location of your WIM, and the mountdir path to where you created that folder.

Loading the WIM Registry

Navigating to the empty folder you created will now show you the contents of your WIM. The connection string that is used by the RPS API is a PowerShell environment variable, which cannot be set inside of WinPE, unless already exists in the registry of the WIM file. In order to do this, open the registry editor on the machine where you mounted the WIM, select HKEY_LOCAL_MACHINE, then click File>Load Hive. Navigate to your mount directory, then to Windows, System32, config, and select the file called "System".

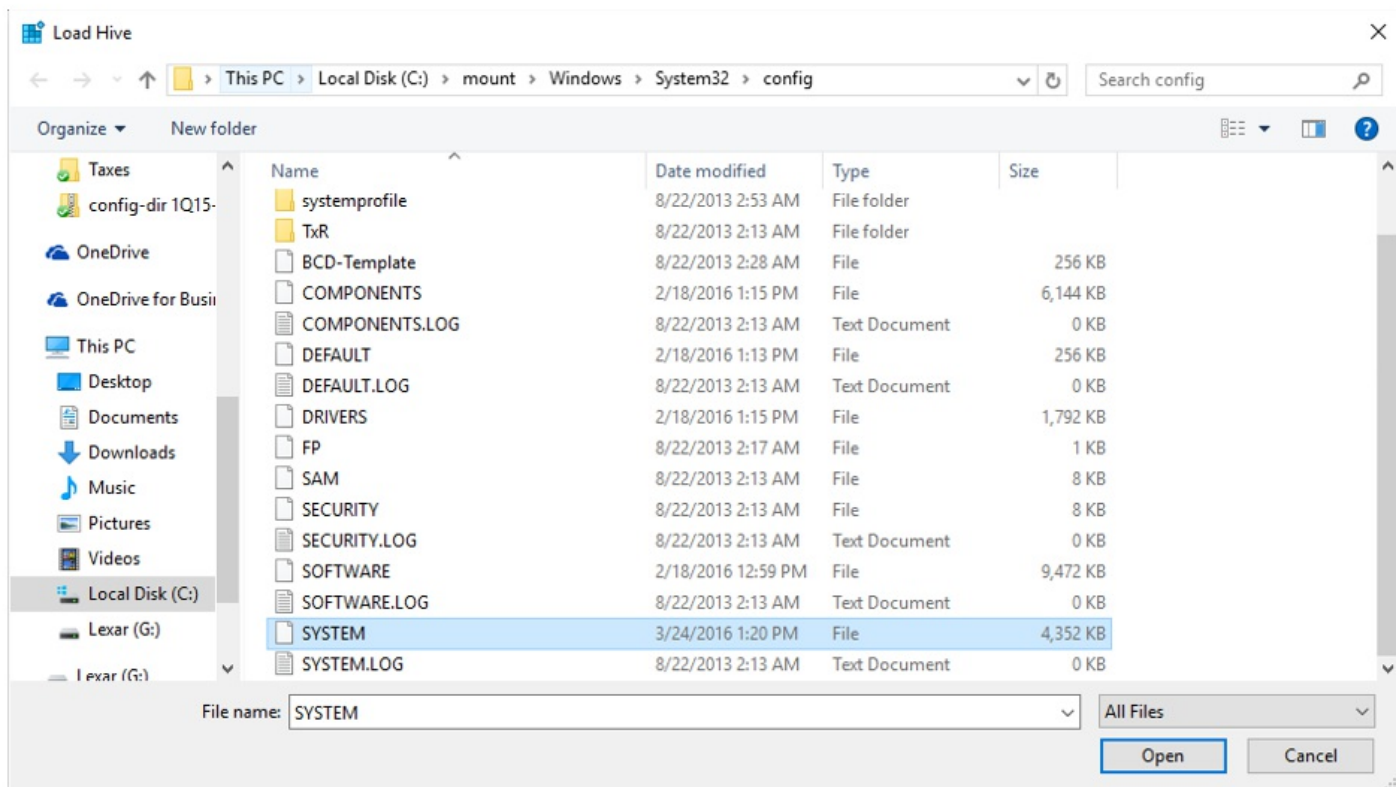


Figure 12 WIM Registry

Adding new Registry Keys to WIM Registry

After selecting Open, the registry editor will prompt for a key name, enter "WIM", and then select Ok. Expand WIM>ControlSet001>Control>Session Manager>Environment. Right Click on the Environment folder and select New>String Value. For the name of the registry key enter "RPSDbAddress" and for the Value, enter the following connection string:

```
Data Source=DB\RPS;Initial Catalog=Rps.Database;
MultipleActiveResultSets=True;user=user;password=password
```

Make sure the Data Source is the name of your SQL Server\InstanceName, the catalog is the name of the RPS CMDDB Database, and the user and password are the credentials of a local SQL Account with access to read and write to the database. It is important to note that the user provided in the connection string must be a local SQL account, as domain accounts are not supported when using SQL connection strings and supplying the username and password as part of the string. Once you have entered the connection string, click OK to save the registry key. The key should now appear under the Environment folder.

```
Data Source = DB\RPS; Initial Catalog = Rps.Database;
MultipleActiveResultSets = True; user = user; password = password
```


Editing a Task Sequence

Once the task sequence has been created, you can begin editing it to suite your environment and the steps that you want to clients to execute in order to deploy an operating system to your clients. To being editing your Task Sequence, right-click on the Task Sequence in the Deployment Workbench and select "Properties", the navigate to the "Task Sequence" tab. The following window will appear. Note: This is an example of a task sequence where task sequence variables and steps have already been supplied.

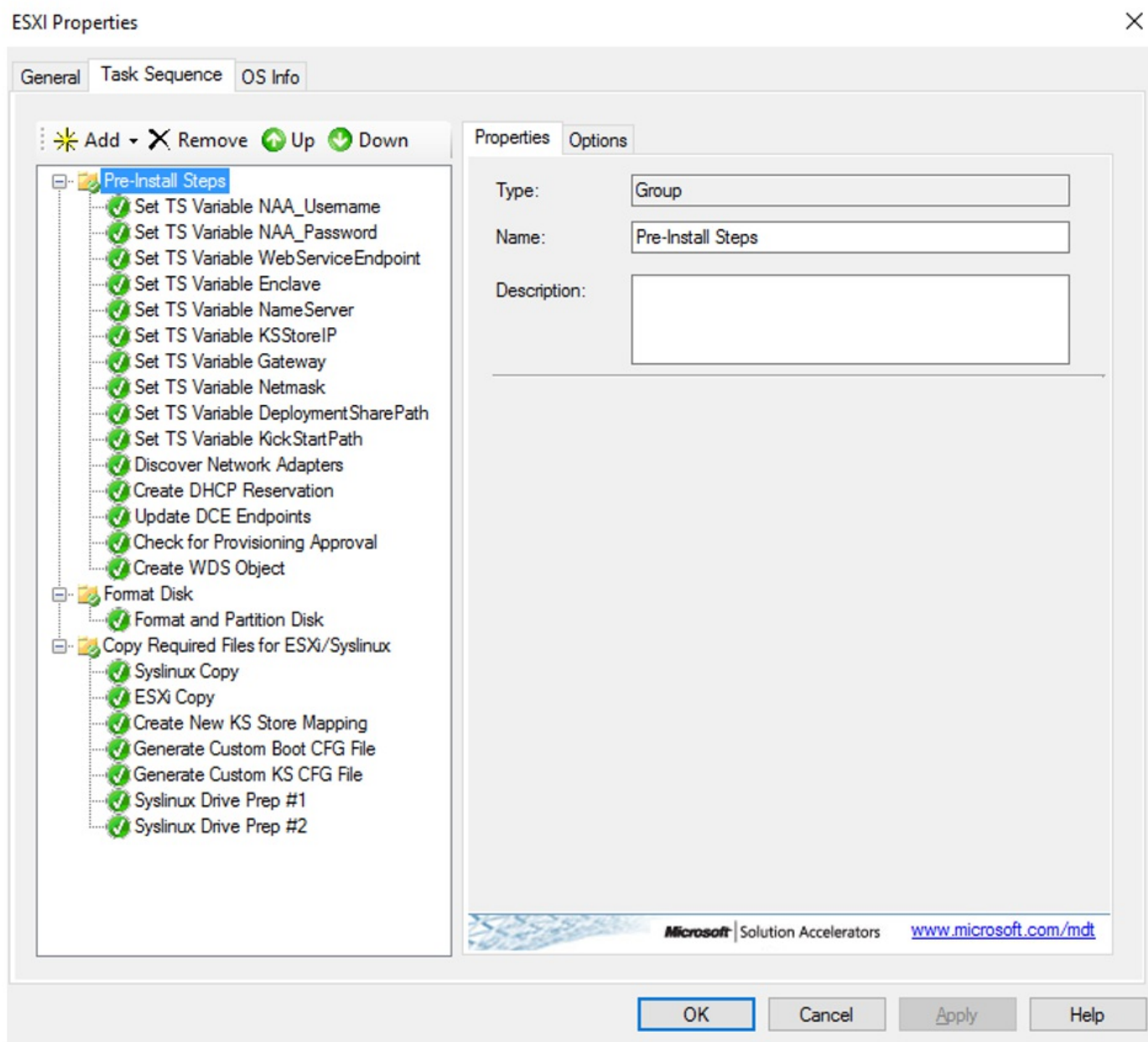


Figure 15 Task Sequence

In order to add steps to your Task Sequence, click on the "Add" button at the top, select "General" and then select "Set Task Sequence Variables." Task Sequence Variables are pieces of information that clients will need to perform various functions. For example, if the client is going to need to interface with RPS, it will need to know the Web Service Endpoint of the RPS server. Instead of hardcoding that address into the script that will interface with RPS, it can be supplied as a variable that can be consumed by the client, should it ever need to change.

Generally, it is recommend that you supply all of your necessary Task Sequence Variables as the first steps in the Task Sequence, so that they are available to the client before any actual deployment logic is performed. After you have inputted all of your necessary Task Sequence Variables, you can proceed to adding steps that require PowerShell Scripts to be executed, as well as any other options that are required. All the available options are available under the "Add" button. Note: If you are adding PowerShell scripts as steps in your task sequence, for the value of the "PowerShell script" field, the name of the script should be entered in the following manner:

```
%SCRIPTROOT%\FilenameOfScript.ps1
```

All of the scripts required by your task sequence should be placed in the "Scripts" folder of your Deployment Share.

Credential Masking

Clients that boot into WinPE are not domain joined, and usually need domain account credentials in order to perform various functions throughout the Task Sequence. One such example is if the client needs to interface with (Job Host Server TBD) in order to start a Runbook. Storing the password as clear text is not recommended, and one method of getting around this is to use PowerShell Obfuscation.

Encoding

Credentials within PowerShell are stored and consumed as Secure Strings. In order to do this, open a PowerShell session and issue the following command, replacing "Password" with the password you want to secure:

```
$pass = "Password" | ConvertTo-SecureString -AsPlainText -Force
```

Converting to Masked Credential

If you return the \$pass variable created above, you will note that it does not return the value of your "Password", but instead returns a PowerShell Object of Secure String. What we want to return is the hash that is generated from securing the password. In order to do this, in the PowerShell session that you have open, issue the following command:

```
$hash = ConvertFrom-SecureString $pass -Key (1..16)
```

Now if you return \$hash, a very long string of characters will be returned, which is the hashed version of your secure password. You can then take this hashed value and store it as a Task Sequence Variable in your task sequence, to be consumed and utilized by clients booting into WinPE.

Converting Masked Credential Back to a Secure String

In order to convert the hashed value back into a Secure String, the following command is used:

```
$pass = $hash | ConvertTo-SecureString -Key (1..16)
```

Workflow Examples

The following section will provide some very basic examples of how to perform certain operations in PowerShell scripts that run within WinPE, how to interact with the RPS API, and how to interface with the Controller.

Using Masked Credential

The previous section described how to create a password hash to be consumed as a Task Sequence Variable. The following syntax is used to read the task sequence variable and use it to create a credential object for PowerShell to utilize within WinPE. Note the name of the Task Sequence Variable in question is "Password" and the value is the hash that was created by following the steps from the previous section.

```
$MS_ConfigMgr_Env = New-Object -ComObject Microsoft.SMS.TSEnvironment  
$Password = $MS_ConfigMgr_Env.Value ("Password")  
$Username = $MS_ConfigMgr_Env.Value ("Username")  
$cred = $Password | ConvertTo-SecureString -key(1..16)  
credentials = New-Object System.Management.Automation.PSCredential ($UserName, $cred)
```

Importing RPS API

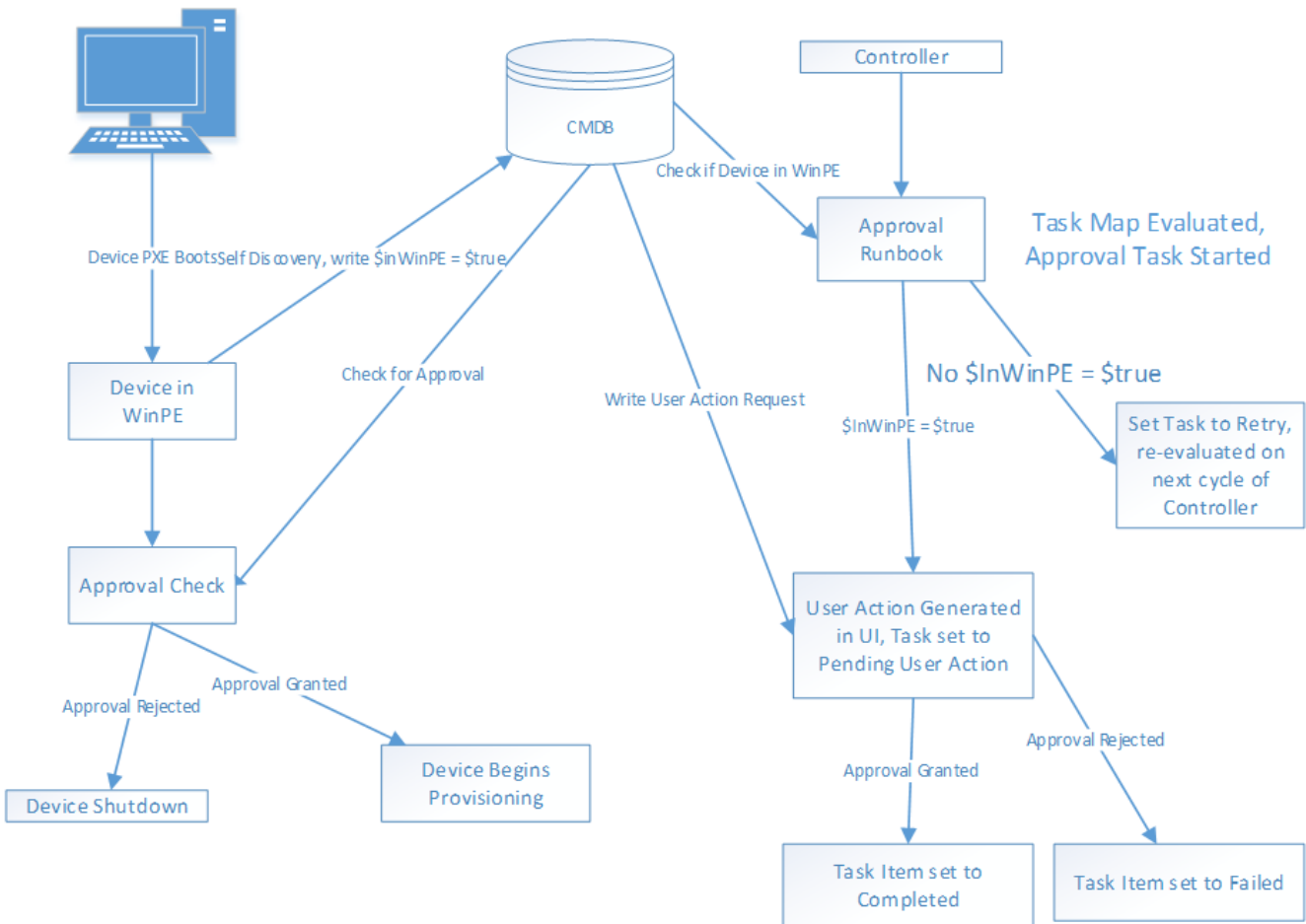
The following syntax is used to import the RPS API so that clients within WinPE can interact with the DAC and CMDB. Note, the

RPS API should be placed in the Deployment Share, under the Applications Folder, and then in its own folder called DAC.

```
$MS_ConfigMgr_Env = New-Object -ComObject Microsoft.SMS.TSEnvironment
$deploymentSharepath = $MS_ConfigMgr_Env.Value("DeploymentSharePath")
Import-Module "$deploymentSharePath\Applications\DAC\Rps.Api.dll"
```

Bare Metal Provisioning Approval

On headless systems, administrators using RPS need a way to approve provisioning of client devices before their hard drives are wiped and a new Operating System installed on the hard drive. The following set of steps give an example of how users utilizing the RPS toolset would be able to perform an example approval process. The diagram below is an example overview of this process.



Client generates flag that it is in WinPE

First, the client needs to set a flag for its Target Item that it has booted into WinPE, so that the controller is aware and can generate the approval. For the first PowerShell script that runs in the Task Sequence, some kind of code similar to the following should be used. In short, the client is querying the CMDB for its Target Item, and setting a property called "InWinPE" to \$true.

```
# find the Client

$Client = Get-RpsTargetItem -Name $ComputerName

if($Client)
{
    $Client.IsActive = $true
    $Client.InWinPE = $true
    Update-RpsTargetItem -TargetItem $Client
}
}
```

Approval Runbook

The following is an example Runbook that is run by the Controller to check if a client has booted into WinPE, and if so, generate an approval task for an administrator either Approve or Reject. Note, this should be the first step in the TaskMap associated with the client device, so that no automations proceed on the client until this approval has been granted.

```
workflow New-BareMetalApprovalTask
{
    param
    {
        [parameter(Mandatory = $true)][string]$taskAssignmentId
    }

    # Disables serialization of objects in certain circumstances allowing for method calls.
    $PSDisableSerializationPreference = $true

    $taskassignment = Get-RpsTaskAssignment -Id $taskAssignmentId
    $targetitem = Get-RpsTargetItem -Id $taskassignment.TargetItemId

    if($targetitem.InWinPE -eq $true)
    {
        New-RpsTaskAssignmentUserAction -TaskAssignmentStatusId $taskAssignmentId
        $null = ($targetitem.ApprovalAssignmentId = "$taskAssignmentId")
        $null = Update-RpsTargetItem -TargetItem $targetitem
        $null = ($taskassignment.StatusMessage = "Approval Generated")
        $null = ($taskassignment.TaskState = "PendingUserAction")
        $null = Update-RpsTaskAssignment -TaskAssignment $taskassignment
    }
    else
    {
        $null = ($taskassignment.TaskState = "Retry")
        $null = ($taskassignment.StatusMessage = "Target Item is not in WinPE")
        $null = Update-RpsTaskAssignment -TaskAssignment $taskAssignment
    }
}
```

Client Checks for Approval

Prior to the step in the Task Sequence where the client wipes its hard drive and lays down the new Operating System, it needs to verify that approval has either been received or denied. To do this, a PowerShell script similar to the following example should be created and placed on the Deployment Share, and then called from the task sequence.

```

# Obtain TS vars
$MS_ConfigMgr_Env = New-Object -ComObject Microsoft.SMS.TSEnvironment
$deploymentSharePath = $MS_ConfigMgr_Env.Value("DeploymentSharePath")

# Set Computer Name
$ComputerName = "$ClientComputerName"

# Import DAC
Import-Module "$deploymentSharePath\Application\DAC\Rps.Api.dll"

# find the client in CMDB
$Client = Get-RpsTargetItem -Name $ComputerName

# find the Task Assignment
$TaskAssignment = $Client.Properties | where Name -eq "ApprovalAssignmentId"
while($TaskAssignment)
{
    Start-Sleep -Seconds 5
    $Client = Get-RpsTargetItem -Name $ComputerName
    $TaskAssignment = $DCE.Properties | where Name -eq "ApprovalAssignmentId"
}
$TaskAssignment = $TaskAssignment.Value

# Monitor for Approval/Rejection
if ($TaskAssignment)
{
    do
    {
        Start-Sleep -Seconds 5
        $Status = (Get-RpsTaskAssignment -Id $TaskAssignment).TaskState
    }
    while (($Status -ne "Completed") -and ($Status -ne "Canceled"))

    if($Status -eq "Completed")
    {
        $DCE.InWinPE = $null
        Update-RpsTaskAssignment -TaskAssignment $DCE
    }

    if($Status -eq "Canceled")
    {
        $DCE.InWinPE = $null
        Update-RpsTaskAssignment -TaskAssignment $DCE
        wpeutil shutdown
    }
}
}

```

Glossary

TERM	DEFINITION
ADK	Assessment and Deployment Kit.
CMDB	Configuration Management Database.
MDT	Microsoft Deployment Toolkit.
RPS	Rapid Provisioning System – A Toolset used to perform automations.
Runbook	A "task" executed within TMS, may contain multiple workflows.

TERM	DEFINITION
Runbook Worker	The TMS service that processes and executes Runbooks.
SCCM	Microsoft System Center Configuration Manager.
TMS	Task Management Service - hosts & runs runbooks, used by RPS.
TaskMap	Identifies a set of steps, what order they should be performed in, and what target items those steps apply to.
Web Service	A web-based receiver that enables connectivity from other applications.
WDS	Windows Deployment Services – A windows service used to perform provisioning of clients.
WIM	Windows Imaging Format – A Bootable file allowing a client to enter WinPE.
WinPE	Windows Pre-Boot Environment – An Environment that runs in memory on the client that is used to provision the client.

More Resources

- [RPS Building iPXE ROMs](#)
- [Configuring ESXi VMs to Use iPXE](#)

Configuring ESXi VMs to Use iPXE

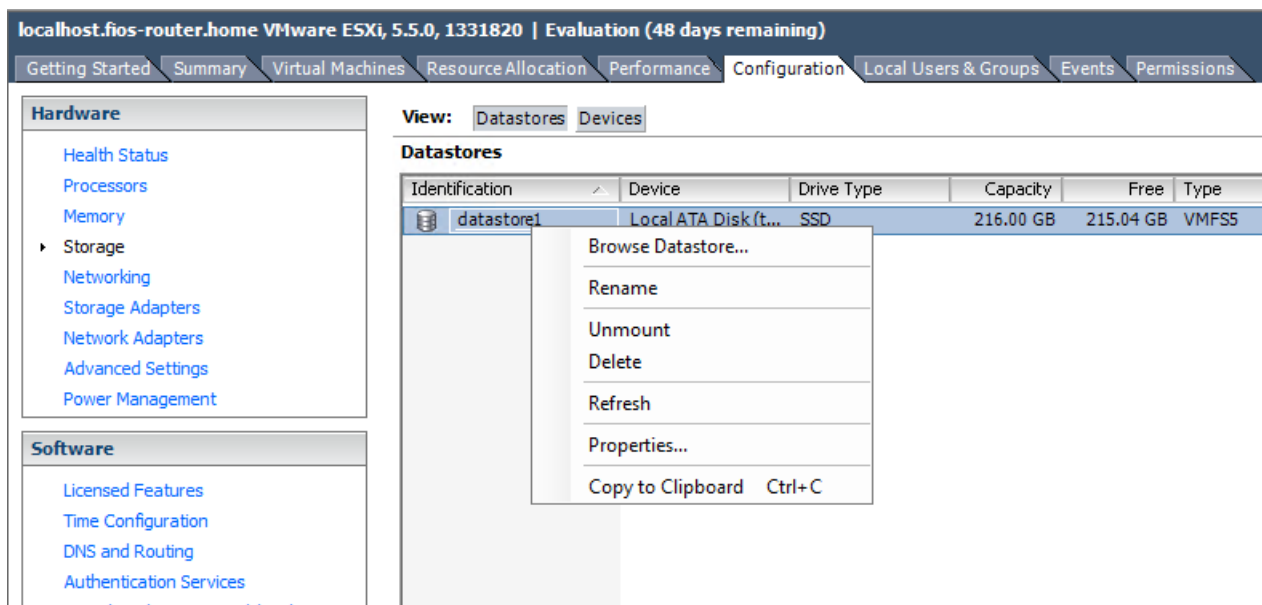
Last updated on July 31, 2018.

Last Reviewed and Approved on PENDING REVIEW

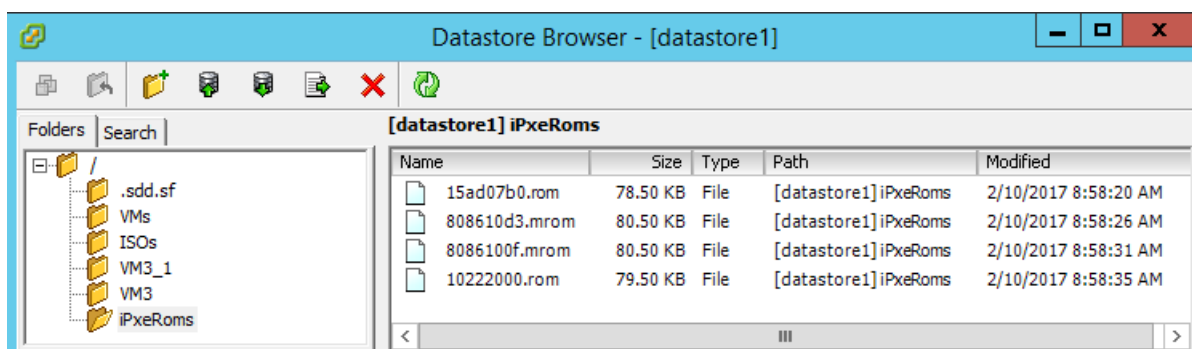
Updating ESXi VM to Use iPXE

The following steps are required to update an ESXi VM to use iPXE:

1. Select the ESXi host.
2. Select the Configuration tab.
3. Right click on the datastore, the ROMs will be copied to and select Browse Datastore...



4. The Datastore Browser window will appear.



5. Create a folder if desired and select the Upload files to this datastore button to copy the iPXE ROMs to the ESXi host. The above screen shot shows all supported iPXE ROMs copied to the iPxeRoms folder.
6. The following commands show PowerCLI being used to copy the iPXE ROMs to the ESXi host.

```
PowerCLI C:\> $ds = Get-Datastore -Name datastore1
PowerCLI C:\> Copy-DatastoreItem -Item C:\temp\iPxeRoms -Destination $ds.DatastoreBrowserPath -Recurse
```

NOTE

The Update-VmxiPxeSetting.ps1 script will perform all the necessary actions to update the VMs network adapter with the iPXE rom. The only parameters needed are the VM's name (VmName), ESXi host's name (VlServer), and credentials to connect to the ESXi host (Credential). The script requires PowerCLI to be installed.

Example Update

Here is an example of updating VM3 to leverage an iPXE ROM:

```
PS C:\>$cred = Get-Credential root
```

```
PS C:\> .\Update-VmxiPxeSetting.ps1 -VmName VM3 -VIMServer 10.0.0.7 -Credential $cred
```

More Resources

- [RPS Building iPXE ROMs](#)
- [RPS PXE Document](#)

Using the Provisioning Service

Last updated on March 15, 2019.

Last Reviewed and Approved on PENDING REVIEW

Introduction

The RPS Provisioning Service is an HTTPS-based Web API hosted in IIS for use in brokering information from the RPS CMDB to a pre-execution environment such as iPXE for installation of a defined image and configuration. For instance, iPXE can be configured to "point to" the Provisioning Service which will return a boot script file for the MAC address requested. In this manner, iPXE will download and boot the image according to the script which has been defined in the CMDB.

Provisioning Service reads of RPS CMDB data

The Provisioning Service uses the RPS CMDB to query for several entity types and records:

1. Target Item (i.e. **MACAddress** property on a **NIC**)
2. Target Item's parent (i.e. **Computer** that *owns* the **NIC**)
3. Resource Item that is of type **BaselImage**
4. Resource Assignment that assigns the **BaselImage** to the Target Item
5. The **iPxeScript** property on the **BaselImage** Resource Item
6. The **{CustomProperty}** property on the **BaselImage** Resource Item

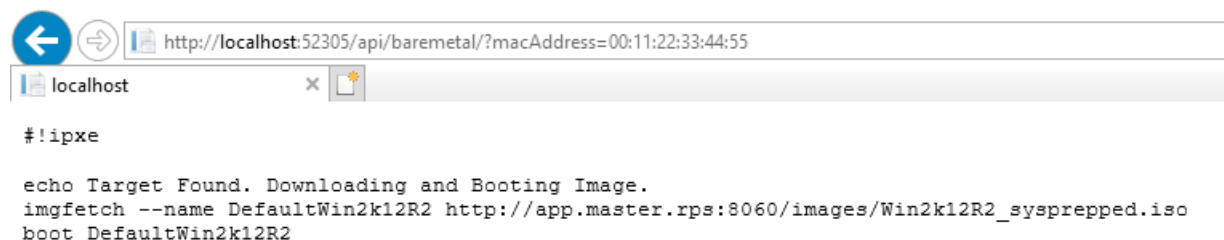
Provisioning Service writes to CMDB

In order to facilitate Target Item retrieval later in the provisioning process (including scenarios where certain environments are unable to perform requests using a URL query string with the MAC Address value), the Provisioning Service can take other inputs to be saved as properties on the parent Target Item. For instance, when iPXE first requests with its MAC Address, it can also send its **SMBiosProductName**, **SMBiosCurrentSpeed**, and **provisionIPAddress**, and potentially other key-value pairs. In this manner, later queries (e.g. through a Runbook) for the target without the MAC Address may be possible to uniquely identify said target.

Baremetal Provisioning Scenarios

Target found

The ideal scenario consists of all records and properties being found (e.g. **NIC** and **iPxeScript** on the **BaselImage**). When the **NIC** Target Item, the **Computer**, and the Resource Assignment that assigns the **BaselImage** to the **Computer** are found, the **iPxeScript** property of **BaselImage** will be returned from the service:



```
#!ipxe
echo Target Found. Downloading and Booting Image.
imgfetch --name DefaultWin2k12R2 http://app.master.rps:8060/images/Win2k12R2_sysprepped.iso
boot DefaultWin2k12R2
```

NOTE

The examples include screenshots from a local iisexpress instance. When it is deployed in the RPS solution on IIS, the server and port name will ultimately differ.

In the case where the required records/properties are found, the **iPxeScript** stored in the CMDB is returned to the requesting client, verbatim.

When the script is executed by iPXE, the environment will attempt to download and install the image from specified in the iPXE Script.

Target found and additional query parameters added

If the Target Item is found, we can also specify additional parameters to add as properties to the parent Target Item. This example shows a different browser and a different *Target Found* iPXE Script, and also shows where **SMBiosProductName**, **SMBiosCurrentSpeed**, and **provisionIPAddress** were added to the CMDB through the Provisioning Service, and then a **newDynParam** added as well to showcase dynamic parameters.

The screenshot shows a browser window with the URL `http://localhost:52305/api/targeting/?physicalAddress=00%3A11%3A22%3A33%3A44%3A55&smBiosProductName=Phoenix&smBiosCurrentSpeed=2.9&provisionipAddress=192.168.1.25&newDynParam=1234`. Below the browser is a terminal window with the following content:

```
#!/ipxe
echo Target Found. Downloading and Booting Image.
imgfetch --name DefaultWin2k12R2 http://localhost:52305/images/Win2k12R2_aysprepped.iso
boot DefaultWin2k12R2
```

The PowerShell terminal shows the execution of `Find-RpsTargetItem -Name dce` twice, returning JSON objects with various properties:

```
PS C:\Src\MM\Core\Utilities\Demos\FY19P11\Sprint1> Find-RpsTargetItem -Name dce

Id                : 36741e84-00b8-4e95-a293-3196ff9b573d
ParentItemId      :
ParentItem        :
NodeId            : adc351ec-5565-4f95-94da-e9ada945851e
Name              : DCE
Type              : VirtualMachine
IsActive          : True
Container         : VirtualMachine - DCE
ContainerNode     : BENDRAK-SB2
IsContainer       : True
ChildItems        : (DCE-NIC)
ParentGroups      : {}
TaskAssignments   : {}
TaskMapAssignments : {}
Properties         : {[MACAddress, 00:11:22:33:44:55]}

PS C:\Src\MM\Core\Utilities\Demos\FY19P11\Sprint1> Find-RpsTargetItem -Name dce

Id                : 36741e84-00b8-4e95-a293-3196ff9b573d
ParentItemId      :
ParentItem        :
NodeId            : adc351ec-5565-4f95-94da-e9ada945851e
Name              : DCE
Type              : VirtualMachine
IsActive          : True
Container         : VirtualMachine - DCE
ContainerNode     : BENDRAK-SB2
IsContainer       : True
ChildItems        : (DCE-NIC)
ParentGroups      : {}
TaskAssignments   : {}
TaskMapAssignments : {}
Properties         : {[ProvisionIpAddress, 192.168.1.25], [SMBiosProductName, Phoenix], [SMBiosCurrentSpeed, 2.9]}

PS C:\Src\MM\Core\Utilities\Demos\FY19P11\Sprint1>
```

NOTE

The **macAddress** can be encoded with a **%3A** value in place of **:**. This is acceptable, as are **:** or **-**, so long as **macAddress** is explicitly specified in the URL query string.

Target not found

If the MAC address is not found as a property of any Target Item, a default iPXE script will be returned to the client. This script prints a message, will sleep for 30 s and then reboot:

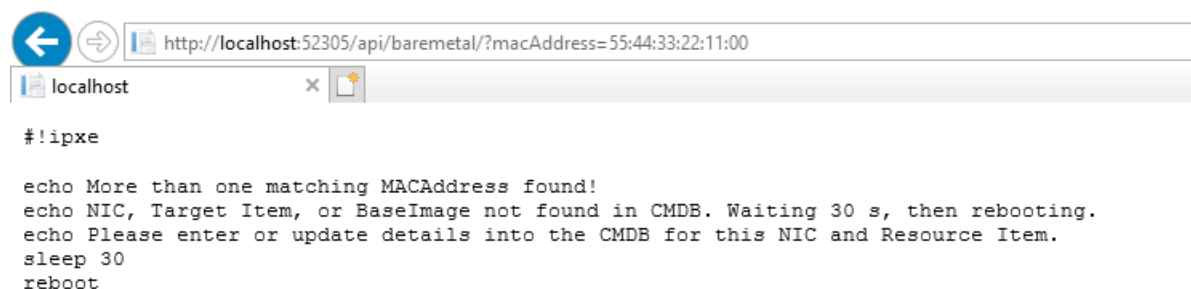
The screenshot shows a browser window with the URL `http://localhost:52305/api/baremetal/?macAddress=00:11:22:33:44:56`. Below the browser is a terminal window with the following content:

```
#!/ipxe

echo NIC, Target Item, or BaseImage not found in CMDB. Waiting 30 s, then rebooting.
echo Please enter or update details into the CMDB for this NIC and Resource Item.
sleep 30
reboot
```

Duplicate MAC found

If more than one of the same MAC address is found in the CMDB, a default iPXE script will be returned to the requesting client:



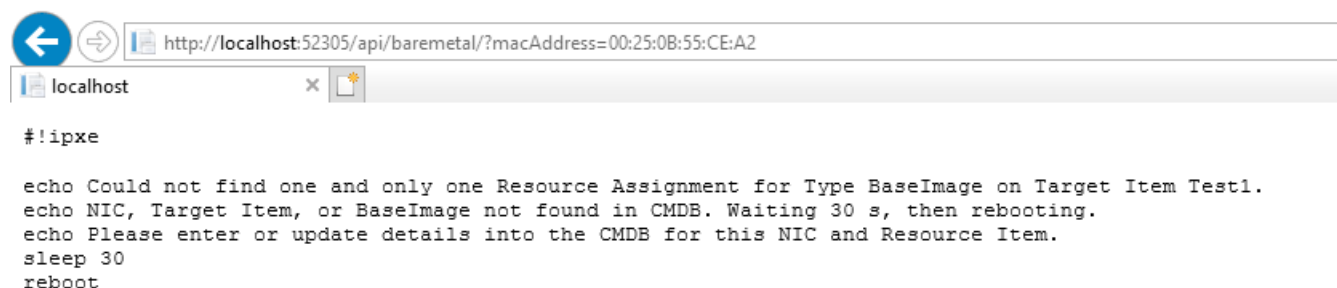
The screenshot shows a browser window with the address bar containing the URL `http://localhost:52305/api/baremetal/?macAddress=55:44:33:22:11:00`. The browser tab is titled 'localhost'. The main content area displays the following iPXE script output:

```
#!/ipxe

echo More than one matching MACAddress found!
echo NIC, Target Item, or BaseImage not found in CMDB. Waiting 30 s, then rebooting.
echo Please enter or update details into the CMDB for this NIC and Resource Item.
sleep 30
reboot
```

Resource Assignment cannot be determined

If there is not one (and only one) Resource Assignment, a default iPXE script will be returned to the requesting client:



The screenshot shows a browser window with the address bar containing the URL `http://localhost:52305/api/baremetal/?macAddress=00:25:0B:55:CE:A2`. The browser tab is titled 'localhost'. The main content area displays the following iPXE script output:

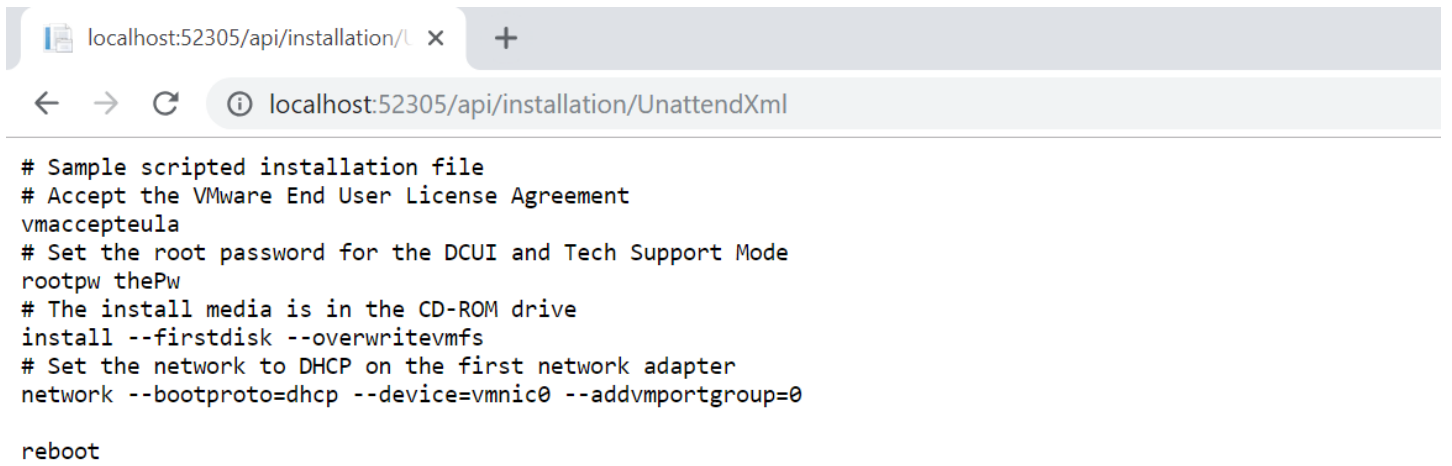
```
#!/ipxe

echo Could not find one and only one Resource Assignment for Type BaseImage on Target Item Test1.
echo NIC, Target Item, or BaseImage not found in CMDB. Waiting 30 s, then rebooting.
echo Please enter or update details into the CMDB for this NIC and Resource Item.
sleep 30
reboot
```

Installation Provisioning Scenarios

The subsequent iPXE operations, the Provisioning Service can be used to receive another scripted file for unattended installations which is commonly referred to as a *Kickstart* file. This script is to be hosted in the CMDB on the **BaseImage**, similar to the **iPxeScript**. Since the environment in this phase of the installation is not equipped to use dynamic parameters (such as the **macAddress** resolved by iPXE), the Provisioning Service will attempt to use the client's IP Address which ought to have been saved to the Target Item (e.g. DCE) during the baremetal provisioning sequence.

If the custom script is already stored on the **BaseImage**, the **ResourceAssignment** has a **ResourceStatus** of **Approved** and the **provisionIpAddress** on the Target Item matches that of the client, the client will receive this custom script, like:



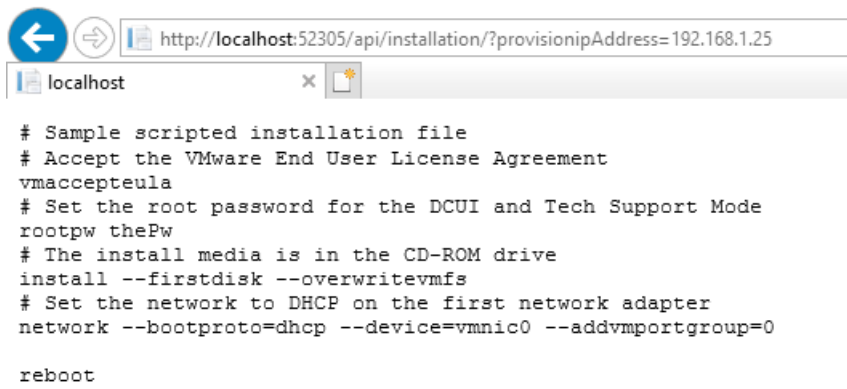
```
# Sample scripted installation file
# Accept the VMware End User License Agreement
vmaccepteula
# Set the root password for the DCUI and Tech Support Mode
rootpw thePw
# The install media is in the CD-ROM drive
install --firstdisk --overwritevmfs
# Set the network to DHCP on the first network adapter
network --bootproto=dhcp --device=vmnic0 --addvmpportgroup=0

reboot
```

NOTE

The Provisioning service can return any custom script. You can do this by calling <https://localhost/api/installation/UnattendXml>. This will return the value of the property **UnattendXml** stored on the **BaseImage**.


Of course, the IP Address can be specified explicitly, like:



```
# Sample scripted installation file
# Accept the VMware End User License Agreement
vmaccepteula
# Set the root password for the DCUI and Tech Support Mode
rootpw thePw
# The install media is in the CD-ROM drive
install --firstdisk --overwritevmfs
# Set the network to DHCP on the first network adapter
network --bootproto=dhcp --device=vmnic0 --addvmpportgroup=0

reboot
```

The **GetAll** action can also be used to see all Target Items with a **provisionIpAddress**:



```
Target Item Name: MDA; ProvisionIpAddress: 192.168.10.107
Target Item Name: DCE; ProvisionIpAddress: 192.168.1.25
```

Summary

The RPS Types have been updated to account for expected **Computer** definitions (e.g. MDA, DCE) as well as the Resource Item

type for **BaseImage**. These definitions help facilitate the creation of appropriate records in the CMDB, including the customization of what is contained in the iPXE script. Without a matching Target Item (and parent Computer), and assigned Resource Item (**BaseImage**) in the CMDB, the Provisioning Service will default to functionality which informs, then reboots the client.

More Resources

- See the ProvisioningServiceDemo.ps1 in `Utilities\Demos\Provisioning` for examples when using RPS

How to Add Runbooks to RPS

Last updated on March 30, 2021.

Last Reviewed and Approved on PENDING REVIEW

Purpose

The purpose of this document is to provide an overview of how to add new runbooks to RPS.

Runbook Pre-Requisites

The following items are required to create a valid runbook:

1. The file must end in `.ps1`.
2. The file name cannot be a duplicate of another runbook.
3. The runbook script must have a mandatory `TaskAssignmentId` parameter.

```
[Parameter(Mandatory = $true)]  
[System.Guid]  
$TaskAssignmentId
```

4. Upon successful completion, Task Assignment status should update to "Completed" status using the `Update-RpsTaskAssignment` PowerShell snippet.

```
Update-RpsTaskAssignment -TaskAssignment $taskAssignment -TaskState Completed -StatusMessage $message
```

NOTE

For more information on other available statuses, see [Task Assignment States](#).

Add a New Runbook

Use the following steps to add a new runbook:

1. Create a runbook script in PowerShell. For more information, see [Authoring RPS Runbooks](#).
2. Add the runbook file to the content store Runbooks folder.

NOTE

The default runbook directory is `C:\ContentStore\Runbooks`. However, this location can be changed and may be a different directory.

To get the current location of the runbook directory, open PowerShell and run this `Get-RpsStorageValue` PowerShell snippet:

```
Get-RpsStorageValue -Key "DefaultRunbookFolder"
```

See [How to Get and Set the Default Runbook Folder](#) for more information.

3. On the machine where the runbook should be added, open PowerShell as an Administrator. Run the `New-RpsTaskItem` PowerShell snippet. This will create a new Task Item for RPS.

```
$taskItem = New-RpsTaskItem -WorkflowName 'workflowItem2' -IsActive $true
```

4. Create a new Task Assignment using the `New-RpsTaskAssignment` PowerShell snippet. This will assign the Task Item to the target that the runbook is to be applied.

```
$taskAssign = New-RpsTaskAssignment -TaskItem $taskItem -TargetItem $targetItem1
```


How to Get and Set the Default Runbook Folder

Last updated on January 8, 2021.

Last Reviewed and Approved on PENDING REVIEW

Purpose

The purpose of this document is to provide an overview of how the default runbook folder is installed and manually changed.

Runbook Folder Installation (Default)

The default runbook folder is automatically set during RPS Installation. The default directory is `C:\ContentStore\Runbooks`.

Get Current Location of the Runbook Folder

By using the `Get-RpsStorageValue` command in PowerShell, you are able to get the current location of the runbook directory.

```
Get-RpsStorageValue -Key "DefaultRunbookFolder"
```

Manual Modification of the Runbook Folder

By using the `Set-RpsStorageValue` command in PowerShell, you are able to change the runbook directory.

```
Set-RpsStorageValue -Key "DefaultRunbookFolder" -Value "[Full path of the folder you wish to use]"
```

How to Modify Runbooks in RPS

Last updated on March 30, 2021.

Last Reviewed and Approved on PENDING REVIEW

Purpose

The purpose of this document is to provide an overview of how to modify existing runbooks in RPS.

Modify an Existing Runbook

Use the following steps to modify an existing runbook:

1. Use File Explorer to find the file on disk.

NOTE

The default runbook directory is `C:\ContentStore\Runbooks`. However, this location can be changed and may be a different directory.

To get the current location of the runbook directory, open PowerShell and run this `Get-RpsStorageValue` PowerShell snippet:

```
Get-RpsStorageValue -Key "DefaultRunbookFolder"
```

See [How to Get and Set the Default Runbook Folder](#) for more information.

2. Open the file in your preferred text editor.

NOTE

This file can then be modified and saved. RPS will run the new version when the runbook executes.

3. Open PowerShell as an Administrator to update the runbook assignment's status using the following steps:

1. Run `Get-RpsTaskItem` to get the Task Item.

```
$task = Get-RpsTaskItem -WorkflowName name
```

2. Run `Get-RpsTaskAssignment` to get the Task Assignment for a designated Target Item and current Task Item.

```
$assignment = Get-RpsTaskAssignment -TargetItem $targetItem -TaskItem $task
```

3. Run `Update-RpsTaskAssignment` to update the Task Assignment to a "Ready" Task State.

```
Update-RpsTaskAssignment -TaskAssignment $assignment -TaskState Ready
```

How to Remove Runbooks From RPS

Last updated on March 30, 2021.

Last Reviewed and Approved on PENDING REVIEW

Purpose

The purpose of this document is to provide an overview of how to remove existing runbooks from RPS.

Remove a Runbook

Use the following steps to remove an existing runbook:

1. Open PowerShell as an Administrator to remove Task Assignments and Task Items using the following steps:

1. Run `Get-RpsTaskItem` to get the desired Task Item for a given workflow.

```
$task = Get-RpsTaskItem -WorkflowName name
```

2. Remove all Task Assignments that use the Task Item.

```
Get-RpsTaskAssignment -TaskItem $task | Remove-RpsTaskAssignment
```

3. Run `Remove-RpsTaskItem` to remove the Task Item.

```
Remove-RpsTaskItem -Id $task.Id
```

2. Locate the runbook file in the content store Runbooks folder and delete the file.

NOTE

The default runbook directory is `C:\ContentStore\Runbooks`. However, this location can be changed and may be a different directory.

To get the current location of the runbook directory, open PowerShell and run this `Get-RpsStorageValue` PowerShell snippet:

```
Get-RpsStorageValue -Key "DefaultRunbookFolder"
```

See [How to Get and Set the Default Runbook Folder](#) for more information.

Task Management Service (TMS) Settings

Last updated on March 30, 2021.

Last Reviewed and Approved on PENDING REVIEW

NOTE

Also known as RPS Phyr.

Purpose

The purpose of this document is to provide an overview of the available settings in TMS.

```
<!-- Override values - If these are used, then they will override RPS / other settings. -->
<!-- Fill in value to override RPS Settings to locate Runbook files. Default: C:\ContentStore\Runbooks. -->
<add key="DefaultRunbookFolder" value="" />
<!-- Fill in value to override RPS Settings to set logging level. Values: Verbose, Information, Warning, Error, Fatal. Default: Information. -->
<add key="TMSLoggingLevel" value="Information" />
<!-- Fill in value to override RPS Settings to set the internal global retries. Default: 0. -->
<add key="MaxRetries" value="1" />
<!-- Fill in value to override RPS Settings to set the storage value related to PhyrConnectionString if it is used and valid. Values: Inherit, Memory, SqlServer, PostgreSQL. -->
<add key="StorageType" value="" />
<!-- Fill in value to override RPS Settings to set the queue database. -->
<add key="PhyrConnectionString" value="" />
<!-- Fill in value to override RPS Settings to set the queue database name. Default: TMS. -->
<add key="DefaultDb" value="TMS" />
```

Figure 1: TMS App Config.

Rules

Unless otherwise stated, TMS settings are located in its own configuration file, `Rps.TaskManagement.exe.config`. If a setting does not exist in the TMS configuration file, TMS will use an RPS setting information with the same key, instead.

The TMS configuration file is found in the same directory as `Rps.TaskManagement.exe`. By default, located at `C:\ContentStore\Rps\TaskManagement`.

Configuration Glossary

Settings

SETTING KEY	DEFINITION	DEFAULT
DefaultRunbookFolder	Location TMS can find the runbooks.	Set at RPS installation. By default, located at <code>C:\ContentStore\Runbooks</code> .
TMSLoggingLevel	See Logging Levels .	Information
MaxRetries	Global maximum number of retries for all jobs. Individual jobs do not have a determined retry, but most runbooks will retry themselves, when appropriate.	1
StorageType	Inherit, Memory, SqlServer, PostgreSQL	Inherit
PhyrConnectionString	Database connection string. Used by Hangfire to connect to a database.	N/A (Inherited from RPS)
DefaultDb	Database used to store active TMS job information.	TMS

TMS Logging Levels

LEVEL	DEFINITION
Verbose	Extra processing about TMS, Phyr, and the underlying job processing.
Information	(Recommended / Default) General information about the service and job processing.
Warning	Background job is delayed or throwing an error, but may be retried.
Error	Background job is unable to perform due to an error that cannot be resolved.
Fatal	Fatal error causing the system to be unable to continue processing any jobs. This level is not used in Phyr but may be used in underlying processes. This level is not likely to produce anything.

RPS Tasking Guide

Last updated on August 26, 2021.

Document Status: Document Developer Quality Pending.

Rapid Provisioning System (RPS) Tasking is a feature of RPS that provides a controlled, predictable method to automate complex tasks in correct sequence. A working knowledge of the entities and architecture of RPS is recommended prior to reading this guide.

Introduction

This guide provides an overview of the RPS Tasking architecture, a list of supporting documentation, RPS task assignment creation, and examples in PowerShell code.

Overview

The center of RPS Tasking capabilities is a **Task Assignment**. A task assignment is the assignment of the smallest unit of work, a **Task Item**, to a single **Target Item**.

To perform a complex sequence of steps, RPS uses a **Task Map**, which describes: a set of steps, what order they should be performed in, and what target items those steps apply to. The Task Map is the blueprint of the process or orchestration. Some examples of Task Maps include: provisioning a new server stack, performing complex patching, or maintenance routines such as issuing new certificates.

The result of assigning a task map to a target item is a **Task Map Assignment**, which is a set of task assignments that together, complete the complex process. A major benefit of RPS is the flexibility of Task Maps, which allows Administrators to author simple, reusable PowerShell runbooks, and compose them together into a complex process across multiple devices.

Glossary

The following terms are used in the Tasking Guide:

TERM	DESCRIPTION	ALIASES (DEPRECATED*)
Direct Assignment	The assignment of a task item directly to a target, without the use of a Task Map.	
Map Assignment	The assignment of a task map to a target and its descendants. The result is a set of Task Assignments that together make up a complex orchestration.	TaskMapAssignment, Orchestration
Runbook	PowerShell script which runs a simple task. Usually run in TMS.	
Step	Identify a single step within a Task Map.	Task Map Step, TaskMapStep, TaskMapDefinition*
Step Dependency	A dependency between a step and its dependent (previous) step.	TaskMapStepDependency, TaskMapDefDependency*
Step Filter	A filter to narrow what target items the step applies to.	TaskMapStepFilter, TaskMapDefFilter*
Target Item	A device which is the target of a task item or task map.	TargetItem, Target

TERM	DESCRIPTION	ALIASES (DEPRECATED*)
Task Assignment	Assignment of a task item to a target. Corresponds to a single job in TMS.	TaskAssignment, Assignment
Task Item	Identifies a task (runbook).	TaskItem, Task
Task Management Service	Used by RPS to run RPS Task Assignments.	TMS
Task Map	Identifies a set of steps, what order they should be performed in, and what target items those steps apply to.	TaskMap, Map

Target Item User Actions

The states in which actions on a Target Item may be classified are detailed below.

STATE	DESCRIPTION	CRITERIA
Pending	An action on a Target Item that is Pending.	Task Assignment(s) assigned to the Target Item are Pending user action.
Optional	An action on a Target Item that is Optional.	Task Assignment(s) assigned to the Target Item have a state of Not Ready and are flagged as Optional.
Retry	An action on a Target Item that is Retryable.	Task(s) assigned to the Target Item have a state of Canceled or ErrorStop.

Task Map Authoring

A **Task Map** is a reusable blueprint for a set of steps that need to be performed on one or more devices. The Rps-API module and cmdlets allow a user to quickly create complex task maps.

Example: Create a Task Map

This example demonstrates the method for creating a Task Map using the Rps-API module. The example is a Task Map which first configures a hypervisor, and then creates a new VM.

Step 1: Create Task Map

Provide the Name and Type.

```
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
```

Step 2: Create Task Items

Next, create a Task Item for the runbooks that do the work.

```
$task1 = New-RpsTaskItem -WorkflowName "Configure-HyperV"
$task2 = New-RpsTaskItem -WorkflowName "New-VirtualMachine"
```

Step 3: Create Steps

Next, create a step for each of the Task Items. You can specify the Task Item using the "RunbookName", "TaskItem", or "TaskItemId" parameter.

```
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer"
$step2 = New-RpsTaskMapStep -TaskMap $map -TaskItem $task1 -TargetItemType "Computer"
$step3 = New-RpsTaskMapStep -TaskMapId $map.Id -TaskItemId $task2.Id -TargetItemType "VirtualMachine"
```

NOTE

The **TargetItemType** parameter is required and is used to determine which type of Target Item to assign to. To use more advanced filtering, see [Step 5: Create Filters](#), below.

Step 4: Create Dependencies

Create a dependency between `$step1` and `$step2`. RPS uses dependencies to evaluate when a task assignment is ready to be run. In this example, `$step2` depends on `$step1`, so it will run only after `$step1` has completed.

```
New-RpsTaskMapStepDependency -PreviousStep $step1 -Step $step2
```

Step 5: Create Filters

Use Step Filters to further restrict what Target Items a Step applies to. For our example, we may only want to provision Hyper-V VMs, so we'll make sure the Target VMs have a property called "HostType" with a value of "HyperV".

```
New-RpsTaskMapStepFilter -TaskMapStep $step2 -PropertyName "HostType" -PropertyValue "HyperV"
```

Example: Simplified Task Map

The example above can be simplified to allow easier Task Map creation. Here is an example of the same Task Map, simplified.

```
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer"
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
`
    -Dependencies @($step1) -Filters @{ HostType = "HyperV" }
```

NOTE

The example assumes that the Task Items were created. Typically, Runbooks will be authored and imported into RPS as Task Items before Task Maps are created.

Target Matching

When a Task Map is assigned to a Target Item, each Step is compared to the Target and its descendant items. By default, RPS doesn't enforce that a step matches a target. By default, RPS also allows a Step to match multiple Target Items, though sometimes the desired behavior is to match a single item only.

To designate a Step as required, use the optional parameter, `IsTargetRequired`.

To designate a Step to allow or prevent multiple matching targets, use the optional parameter, `AllowMultipleTargets`.

WARNING

A Task Map will fail if it is assigned to the same Target Item multiple times while `AllowMultipleTargets = false`, because this configuration does not allow multiple targets.

Example: Required Target Item, No Multiples

In our sample Task Map, we probably want the `Configure-HyperV` step to apply to a single Computer hosting Hyper-V. We will change the first Step to require a Target and disallow multiple matches. We won't change the second step, meaning that this Task

Map will work for a Computer with zero or more VMs.

```
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" -
IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
-Dependencies $step1 -Filters @{ HostType = "HyperV" }
```

Task Map Assignment

After creating a Task Map, assign it to a Target Item to create a **Task Map Assignment**. The map assignment will contain one or more Task Assignments that apply to the Target Item and its descendants.

NOTE

In v2.2 and earlier, RPS only allowed Task Maps to be assigned to root Target Items (Containers).

As of v2.3, RPS allows a Task Map to be assigned to any Target Item, allowing more flexibility in how Target Items are structured, and more re-use of Task Maps and Runbooks. This has the side effect that `$taskMap.GetContainers()` may no longer retrieve the actual targets of the Task Map.

Assign Task Map to Target Item

The example below will create a Task Map, a Target Computer with 2 VMs, and assign the Map to the Target.

```
# Create Task Map
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" -
IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
-Dependencies $step1 -Filters @{ HostType = "HyperV" }

# Create Computer & 2 VMs
$server = New-RpsTargetItem -Type Computer -Name "Server-01"
$vm1 = New-RpsTargetItem -ParentItem $server -Type VirtualMachine -Name "VM-01" -Properties @{ HostType =
"HyperV" }
$vm2 = New-RpsTargetItem -ParentItem $server -Type VirtualMachine -Name "VM-02" -Properties @{ HostType =
"HyperV" }

# Assign Task Map to Computer
New-RpsTaskAssignment -TaskMap $map -TargetItem $server
```

The result of this assignment is a Map Assignment consisting of 3 Task Assignments.

STEP	TASK	TARGET	DEPENDENCIES
1-1	Configure-HyperV	Server-01	
2-1	New-VirtualMachine	VM-01	Step 1-1
2-2	New-VirtualMachine	VM-02	Step 1-1

Assign Task Map to Target Group

A Task Map can also be assigned to a Target Group. The Target Group is used simply as a collection of related Target Items. The assignment will result in a new Map Assignment to each Target in the Group, and each Target will be validated separately.

The example below will create a Task Map, a Target Group with 2 Servers, and assign the Map to the Group.

```

# create a server with 2 VMs
$vmProps = @{ Type = "VirtualMachine"; Properties = @{ HostType = "HyperV" } }
$server = New-RpsTargetItem -Type Computer -Name "Server01"
$vm1 = New-RpsTargetItem -ParentItem $server -Name "VM-01" @vmProps
$vm2 = New-RpsTargetItem -ParentItem $server -Name "VM-02" @vmProps

# create a server with 1 VM
$server2 = New-RpsTargetItem -Type Computer -Name "Server02"
$vm3 = New-RpsTargetItem -ParentItem $server2 -Name "VM-03" @vmProps

# create a group with both servers
$group = New-RpsTargetGroup -Name "Servers" -Type "Server"
$group.AddChildren($server, $server2)

Update-RpsTargetGroup -TargetGroup $group

# assign the map to the group
New-RpsTaskAssignment -TaskMap $map -TargetGroup $group

```

Assign a Task Map to run on the Local Node

By default, Task Assignments will be executed on the Target's Node. Optionally, a Task Map can also be assigned to a Target but executed on the Local (current) Node, instead of the Target's Node. This may be useful when building a child Node from a parent Node. The work to provision the child Node's Virtual Machines will happen locally (on the parent node), but the target computers are assigned to the child node.

Assign a Task Map to a Target and run on the Local Node by using the `-RunOnLocalNode` switch with the `New-RpsTaskAssignment` cmdlet.

Example: Assign a Task Map to Child Target, but run on Parent Node

```

# Create Task Map
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" -
IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
-Dependencies $step1 -Filters @{ HostType = "HyperV" }

# Create Computer & 2 VMs
$server = New-RpsTargetItem -Type Computer -Name "Server-01" -Node $childNode
$vm1 = New-RpsTargetItem -ParentItem $server -Type VirtualMachine -Name "VM-01" -Properties @{ HostType =
"HyperV" }
$vm2 = New-RpsTargetItem -ParentItem $server -Type VirtualMachine -Name "VM-02" -Properties @{ HostType =
"HyperV" }

# Assign Task Map to Computer, but run on current Node
New-RpsTaskAssignment -TaskMap $map -TargetItem $server -RunOnLocalNode

```

Scheduled Task Assignments

RPS v2.3 supports the ability to schedule task assignments. These task assignments will not be executed until the scheduled date.

To schedule a Task Map Assignment, use the optional parameter, `StartDate`.

Example: Scheduled Task Map Assignment

```
New-RpsTaskAssignment -TaskMap $map -TargetItem $server -StartDate "8/27/18 16:00"
```

Assignment Validation

When assigning a Task Map to a Target Item, there may be some steps that don't meet the Task Map's requirements. For example, if a Step requires a matching Target but none is found, the assignment will fail, and an error returned to the user.

Invalid Targets

This code:

```
# Assign Task Map to VM1
New-RpsTaskAssignment -TaskMap $map -TargetItem $vm1
```

Will result in this message:

WARNING: Error assigning Task Map: Provision-VMs to Container: VirtualMachine-VM-01

Step requires a matching target item.

Duplicate Assignments

RPS does not allow the same Task Map to be assigned to the same Target Item multiple times.

This code:

```
# Assign Task Map to Server01 Twice
New-RpsTaskAssignment -TaskMap $map -TargetItem $server
New-RpsTaskAssignment -TaskMap $map -TargetItem $server
```

Will result in this message:

WARNING: Error assigning Task Map: Provision-VMs to Container: Computer-Server01

Container is already assigned to the Task Map.

NOTE

The restriction of duplicate assignments is largely in place for backwards-compatibility reasons and will likely be removed from RPS in the future. Conceptually, a Task Map represents a process that may run multiple times on the same target devices. For now, the two alternatives are to clone a task map in order to run it again or reset the task map assignment so it will run again.

Dependency Scope

If we change the "Provision-VMs" Task Map to include a third step which makes sure the VM is started and waits for the OS to boot, it will have a dependency on Step 2.

```
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" `
    -IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
    -Dependencies $step1
$step3 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Start-VirtualMachine" -TargetItemType
    "VirtualMachine" -Dependencies $step2
```

If we assign that map to our server and 2 VMs, the following Task Assignments will be created:

STEP	TASK	TARGET	DEPENDENCIES
1-1	Configure-HyperV	Server-01	
2-1	New-VirtualMachine	VM-01	Step 1-1
2-2	New-VirtualMachine	VM-02	Step 1-1

STEP	TASK	TARGET	DEPENDENCIES
3-1	Start-VirtualMachine	VM-01	Steps 2-1, 2-2
3-2	Start-VirtualMachine	VM-02	Steps 2-1, 2-2

Looking at the dependencies, you can see that the VM-01 won't start (Step 3-1) until VM-02 has been created (Step 2-2). Likewise, VM-02 waits on VM-01. If we add more Steps that apply to VMs and more VMs to our Server, we'd be creating many dependencies that aren't necessary.

Example 1: Use Scope to narrow Dependencies

Instead of creating dependencies to every target that matched a previous step, we can specify a narrower Scope when we define a Dependency. In RPS v2.3, we've included a new `Scope` parameter when creating Dependencies.

```
$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" -
IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
-Dependencies $step1
$step3 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Start-VirtualMachine" -TargetItemType
"VirtualMachine"
New-RpsTaskMapStepDependency -Step $step3 -PreviousStep $step2 -Scope Self
```

Example 2: Use Scope in-line

NOTE

Scope can be added to a `TaskMapStep` as a `TaskMapStep` parameter. The scope will act on the dependency parameters supplied. The scope will default to target if not specified.

```
New-RpsTaskItem -WorkflowName "Configure-HyperV"
New-RpsTaskItem -WorkflowName "New-VirtualMachine"
New-RpsTaskItem -WorkflowName "Start-VirtualMachine"

$map = New-RpsTaskMap -Name "Provision-VMs" -Type "Provision"
$step1 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Configure-HyperV" -TargetItemType "Computer" -
IsTargetRequired $true -AllowMultipleTargets $false
$step2 = New-RpsTaskMapStep -TaskMap $map -RunbookName "New-VirtualMachine" -TargetItemType "VirtualMachine"
-Dependencies $step1 -Scope Self
$step3 = New-RpsTaskMapStep -TaskMap $map -RunbookName "Start-VirtualMachine" -TargetItemType
"VirtualMachine"
```

If we assign that map to our server and 2 VMs, the following Task Assignments will be created:

STEP	TASK	TARGET	DEPENDENCIES
1-1	Configure-HyperV	Server-01	
2-1	New-VirtualMachine	VM-01	Step 1-1
2-2	New-VirtualMachine	VM-02	Step 1-1
3-1	Start-VirtualMachine	VM-01	Step 2-1
3-2	Start-VirtualMachine	VM-02	Step 2-2

Table: Dependency Scopes

The following Scopes are supported:

SCOPE	DESCRIPTION
Target	Indicates that dependencies will be to the Target of the Task Map, which is the assigned Target Item and all its descendants. This is the default value and the behavior of RPS prior to v2.3.
Parent	Indicates that dependencies will be to the Parent Target Item of the previous step and any descendants.
Self	Indicates that dependencies will be to the Target Item of the previous step and any descendants.

Assignment Code Examples

Assign Task Item to Target Item for Execution

Both the Task Item and the Target Item objects must exist prior to assigning a Task Item to a Target Item. The following example shows the method of assignment. You can add a `StartDate` to the task assignment so Master-Controller won't start execution until the DateTime has been met.

```
$taskAssignments = New-RpsTaskAssignment -TaskItem $AlwaysCompletes -TargetItem $targetItem -StartDate (Get-Date).AddMinutes(15)
```

Assign Task Map to Target Item for Execution

Both the Task Map and the Target Item objects must exist prior to assigning a Task Map to a Target Item. The following example shows the method of assignment.

```
$taskAssignments = New-RpsTaskAssignment -TaskMap $TaskAssignmentTaskMap -TargetItem $targetItem
```

Assign Task Item to Target Group for Execution

Both the Task Item and the Target Group objects must exist prior to assigning a Task Item to a Target Group. The following example shows the method of assignment.

```
$taskAssignments = New-RpsTaskAssignment -TaskItem $AlwaysCompletes -TargetGroup $targetGroup
```

Assign Task Map to Target Group for Execution

Both the Task Map and the Target Group objects must exist prior to assigning a Task Map to a Target Group. The following example shows the method of assignment.

```
$taskAssignments = New-RpsTaskAssignment -TaskMap $TaskAssignmentTaskMap -TargetGroup $targetGroup
```

More Resources

- [RPS Software Design](#)
- [RPS Task Assignment Diagram](#)
- [Authoring RPS Runbooks](#)

RPS Task Assignment Diagram

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

Task Assignment Example

Generate Task Assignments

Generate Task Assignments given the Sample Container (Figure 1) and Task Map (Figure 2). Each Target Item will be evaluated against each Task Map Definition. If the item matches the Type and any Filters, an Assignment will be generated. See Figure 3 for the resulting Assignments.

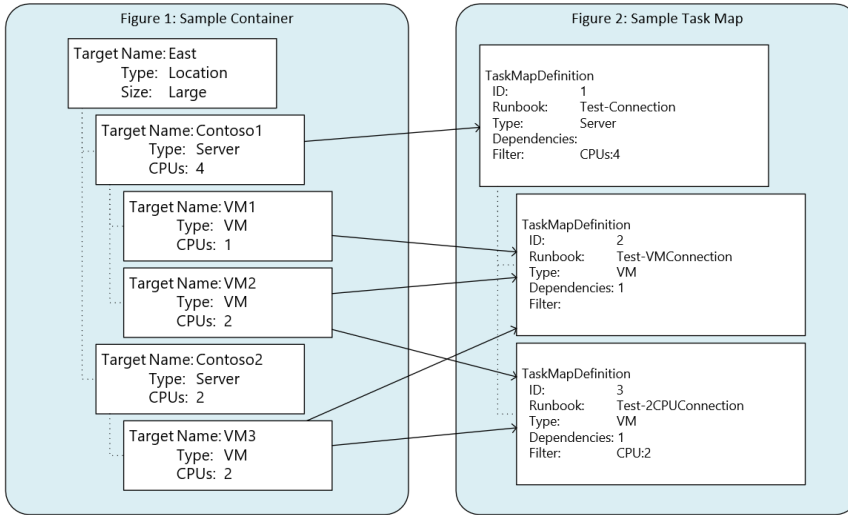


Figure 3: Task Assignments

Target	Task	Status
Contoso1	Test-Connection	NotReady
VM1	Test-VMConnection	NotReady
VM2	Test-VMConnection	NotReady
VM3	Test-VMConnection	NotReady
VM2	Test-2CPUConnection	NotReady
VM3	Test-2CPUConnection	NotReady

More Resources

- [RPS Software Design](#)
- [RPS Tasking Guide](#)
- [Authoring RPS Runbooks](#)

Authoring RPS Runbooks

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

RPS is the Rapid Provisioning System. Runbooks are PowerShell scripts and workflows that RPS will execute with Task Management Service (TMS).

RPS Runbooks are PowerShell runbooks designed to run within the context of RPS. In general, RPS runbooks provide concise, reusable functionality, and use the Rps-Api PowerShell module to access and update the RPS Configuration Management Database (CMDB) data.

The terms "Runbook", "Workflow," and "Task Item" are all used interchangeably within RPS to refer to an RPS runbook. The assignment of a single runbook to a Computer or Target for execution is called a **Task Assignment**, which is executed as one job in TMS. The Task Assignment is the smallest unit of work in RPS, and its status is tracked in the RPS CMDB.

Glossary

The following are common terms used within RPS, Runbooks, and Task Automation.

TERM	DEFINITION
CMDB	RPS [Configuration Management] Database.
Map Assignment	The assignment of a task map to a target and its descendants. The result is a set of Task Assignments that together make up a complex orchestration.
RPS	Rapid Provisioning System.
Rps-Api	PowerShell Module used to access and manipulate RPS configuration and task data.
Runbook	PowerShell script which runs a simple task, usually run in TMS.
Step	Identify a single step within a Task Map.
Target Item	A device which is the target of a Task Item or Task Map.
Task Assignment	Assignment of a Task Item to a target. Corresponds to a single job in TMS.
Task Item	Identifies a task (runbook).
TMS	Task Management Service - hosts & runs runbooks, used by RPS.
Task Map	Identifies a set of steps, what order they should be performed in, and what target items those steps apply to.

Task Assignment States

The following table defines the Task Assignment states used by RPS. Within PowerShell, the available states can be accessed via the `$Rps.TaskStates` variable.

TERM	DEFINITION
Assigned	Task Assignment is created. When TMS creates a job, Sync builds a new Task Assignment.
Canceled	Task Assignment was stopped, either from TMS or manually, and processing of Orchestration may stop.
Completed	Task Assignment is complete, and processing of other Task Assignments in the Orchestration will continue.
ErrorContinue	Task Assignment encountered an error, but processing of Orchestration will continue.
ErrorStop	Task Assignment failed and processing of Orchestration should stop.
None	Internal system default; the task state has not been set.
NotReady	Task Assignment is new or is waiting on other jobs to complete, and not ready to be executed.
PendingUserAction	Task Assignment requires user approval before Orchestration can continue. See User Approvals section.
Ready	Task Assignment is ready to be started, and will be started by TMS.
Removed	Task Assignment contains a target that is being deleted.
Retry	Task Assignment should be retried due to a failure. See Retry section.
Running	Task Assignment was started as a job in TMS (or Direct) and is currently executing.

Hosting Runbooks in TMS vs. Direct Execution

A full instance of RPS (RPS Node) includes the CMDB as a SQL database and an instance of TMS to host and execute RPS runbooks.

With RPS, some of the functionality that TMS uses is exposed directly via the Rps-API module, so that Task Assignments and Task Map Assignments (Orchestrations) can be executed directly in PowerShell. This is useful for an initial RPS Install process, where TMS may not be available. It also helps for developing, testing, and troubleshooting runbooks. See documentation around the `Invoke-RpsTaskAssignment` and `Invoke-RpsTaskMapAssignment` cmdlets for more information.

RPS Runbook Guidance

The following are the three primary guidelines for authoring RPS Runbooks:

1. Runbooks are PowerShell scripts saved as `.ps1` files in the `\Runbooks` folder by default, using the Verb-Noun naming convention.

NOTE

The Runbooks folder location is user changeable. The current Runbook folder location can be determined with the following PowerShell snippet:

```
Get-RpsStorageValue -Key "DefaultRunbookFolder"
```

See: [How to Get and Set the Default Runbook Folder](#)

2. Runbooks contain a single mandatory parameter, `[guid] $TaskAssignmentId`, to identify the Task Assignment.
3. Runbooks use the Rps-API to access and update CMDB data, log important information, and update Task Assignment State.

Example Runbook

The following example is a simplified version of the `Wait-Random.ps1` runbook, which can be used for guidance as well as sample task execution.

```
<# ... #>
[CmdletBinding()]
param
(
    [Parameter(Mandatory = $true)]
    [System.Guid]
    $TaskAssignmentId
)

#Requires -Modules Rps-API

# Load Task Assignment and Target Item
$taskAssignment = Get-RpsTaskAssignment -Id $TaskAssignmentId
$workflowName = $taskAssignment.TaskItem.WorkflowName
if (-not $taskAssignment)
{
    Write-RpsLogItem @LogError -MessageTemplate $LoggingEvents.TaskAssignmentNotFound -Properties
    ($TaskAssignmentId, $workflowName)
    throw "Task Assignment not loaded properly."
}

$targetItem = $taskAssignment.TargetItem
Write-RpsLogItem @LogInformation -MessageTemplate $LoggingEvents.StartingRunbook -Properties $workflowName,
$TaskAssignmentId
Update-RpsTaskAssignment -TaskAssignment $taskAssignment -TaskState $Rps.TaskStates.Running -StatusMessage
"$workflowName is running on $($targetItem.Name)"

# Generate a random sleep time from 1-5 sec
$wait = Get-Random -Min 1 -Max 5
$waitMessage = 'Assignment: {TaskAssignmentId} on {TargetItem} waiting for {Wait} sec'
$logProps = @($TaskAssignmentId, $targetItem.Name, $wait)
Write-RpsLogItem -Level Information -Component Master -MessageTemplate $waitMessage -Properties $logProps

# Sleep
Start-Sleep -s $wait

Update-RpsTaskAssignment -TaskAssignment $taskAssignment -TaskState $Rps.TaskStates.Completed -StatusMessage
"Completed"
```

Updating Task Assignment State from a Runbook

To reduce boilerplate code, RPS will manage the Task Assignment's state as follows:

1. When a Task Assignment is started (via TMS or Direct), its `TaskState` is automatically set to **Running**.
2. When a Task Assignment is completed (via TMS or Direct), and its `TaskState` is **Running**, the `TaskState` is automatically set to **Completed**.
3. When a Task Assignment fails (via TMS or Direct), and its `TaskState` is **Running**, the `TaskState` is automatically set to **ErrorStop**.

In the example runbook, notice that the `TaskState` is never updated to Running or Completed.

To set the state manually, for error handling or a custom workflow, use the `Update-RpsTaskAssignment` cmdlet and the common Task States listed above.

Example: Manually set State

This example manually sets the State to **ErrorContinue** if an error occurs in the `try` block:

```
try
{
    # do work here
}
catch
{
    # catch error but allow orchestration to continue
    # additional logging or error handling here
    Update-RpsTaskAssignment -TaskAssignment $taskAssignment -TaskState $Rps.TaskStates.ErrorContinue -
    Message 'An error occurred in non-critical process.'
}
```

Logging from a Runbook

RPS supports structured logging via the `Write-RpsLogItem` cmdlet. See the Rps-API documentation for full description and examples.

For RPS Runbooks, it's important to understand how to associate information to the Task Assignment for diagnostics or telemetry.

To associate log data with an executing Task Assignment, you must supply a formatted message template with named replacement tokens and the Task Assignment's Id.

Example: Logging Information to the Task Assignment

This example, from the `wait-Random` runbook, shows how to properly log information to a Task Assignment from a runbook. Notice the template is a string with replacement tokens and the properties are simply an object array to associate to the tokens in the template.

```
# Generate a random sleep time from 1-5 sec
$wait = Get-Random -Min 1 -Max 5
$waitMessage = 'Assignment: {TaskAssignmentId} on {TargetItem} waiting for {Wait} sec'
$logProps = @($TaskAssignmentId, $target.Name, $wait)
Write-RpsLogItem -Level Information -MessageTemplate $waitMessage -Properties $logProps
```

Advanced Runbook Guidance

1. Establishing a secure connection to a remote computer
2. Scheduled/Recurring Task Assignments
3. Retrying Task Assignments
4. Require User Approval
5. Additional Parameters

Connect to Target Computer

Many Runbooks will need to connect to the Target (Computer) to perform their duty. To connect, you must get the appropriate credential and then establish a secure connection. In RPS Runbooks, use the `Get-RpsCredential` to load the right credential for the target. Then use `New-SecureSession` from Rps-API to make the connection.

```
#Requires -Modules Rps-API

# Load assignment and target
$assignment = Get-RpsTaskAssignment -Id $TaskAssignmentId
$target = $assignment.TargetItem

try
{
    # Retrieve LocalAdmin credentials for target computer
    $localAdmin = Get-RpsCredential -TargetItem $target -Role 'LocalAdmin'

    # Establish connection to target
    $session = New-SecureSession -IPAddress $target.IPAddress -Type PsSession -Credential $localAdmin -OSType
    $target.OSType

    # Do work on target
}
finally
{
    if ($session)
    {
        # Clean up connection
        Remove-PSSession -Session $session
    }
}
}
```

Scheduled / Recurring Task Assignments

RPS supports scheduling of Task Assignments and Map Assignments. See the RPS Tasking Guide for examples of scheduling new Map Assignments and Task Assignments.

To have a Task Assignment repeat or recur, manually adjust the `StartDate` and set `TaskState` to **Ready** and RPS will run the Task Assignment again.

CAUTION

To prevent TMS from inadvertently completing a recurring task, make sure to clear the `PhyrJobId` when resetting the state.

Example: Set Task Assignment to Recur every 15 minutes

Use the following code in a runbook to have the Task Assignment recur:

TIP

To set a limit on execution counts, add a custom property to the TaskAssignment and logic to update it after each run.

```
$taskAssignment.StartDate = (Get-Date).AddMinutes(15)
$taskAssignment.TaskState = $Rps.TaskStates.Ready
$taskAssignment.PhyrJobGuid = $null
Update-RpsTaskAssignment $taskAssignment
```

Retry Failed Task Assignments

RPS considers the **Retry** state similar to **Ready**. The only difference is that Retry indicates that the Task Assignment failed and is executing again, instead of running for the first time.

Starting with RPS v2.3.2 there is limited support for automatic retries, which was added to support some RPS Install features. To use an automatic retry, set a `RetryCount` property on the Task Assignment. If the Task Assignment fails, RPS will automatically retry it and decrement the `RetryCount` until all retries have been attempted.

This behavior can be simulated in runbooks and may be promoted to a built-in feature in RPS and TMS in a future release. This would also include a `RetryCount` option in the UI or PowerShell when designing a Task Map.

Example: Setting RetryCount property on Task Assignment

```
# Remove -TaskMap parameter to get array of all assignments of $taskItem to $targetItem
$taskAssignment = Get-RpsTaskAssignment -TaskItem $taskItem -TargetItem $targetItem -TaskMap $taskMap
$taskAssignment.RetryCount = 5
Update-RpsTaskAssignment -TaskAssignment $taskAssignment
```

Example: Use RetryCount to limit failures

Place the following sample code into the error handling section of a runbook to implement a limited # of retries. This will only retry a TaskAssignment that already has the `RetryCount` set.

```
$retryCount = [Rps.Api.Utils.Extensions]::GetProperty($TaskAssignment, 'RetryCount', 0)
if ($retryCount -gt 0)
{
    $TaskAssignment.RetryCount = $retryCount - 1
    Update-RpsTaskAssignment -TaskAssignment $TaskAssignment -TaskState $Rps.TaskStates.Retry -StatusMessage
    'Retry from Invoke-RpsTaskAssignment'
}
else
{
    throw "Error on $($computer.Name) in $runbookName. Error: $($TaskAssignment.StatusMessage)"
}
```

Require User Approval

Some Tasks may require manual user approval to continue or stop processing. This approval step requires setting the Task Assignment's state to **PendingUserAction** and setting a prompt for the operator. The operator then approves or denies the action using RPS Web or PowerShell. If approved, the Task's State will be set to **Completed**. If denied, the state is set to **Canceled**.

Example: Update Task Assignment for User Approval

Setting the required approval using the `New-RpsTaskAssignmentUserAction` cmdlet will create a prompt for the user in RPS Web.

```
New-RpsTaskAssignmentUserAction -TaskAssignment $taskAssignment `
    -UserActionPrompt 'Approve Domain Join?' `
    -UserActionApproveLabel 'Approve' `
    -UserActionDenyLabel 'Deny'
```

Additional Parameters

Most runbooks will need more information than just the provided TaskAssignmentID. These runbooks can use ItemMapping attributes such as ResourceItemMapping. When the runbook is executed by TMS these parameters will be resolved and the correct values will be passed in for each parameter. See [How to Configure RPS-Mapped Parameters](#) for more information about ItemMapping attributes.

Example: Multiple Parameters in a Runbook

```
[CmdletBinding()]
param
(
    [Parameter(Mandatory = $true)]
    [System.Guid]
    $TaskAssignmentId,

    [Parameter(Mandatory = $true)]
    [ResourceItemMapping(EntityType = [ResourceTypes]::Credential, Role = 'DomainAdmin')]
    [pscredential]
    $DomainAdminCredential,

    [Parameter(Mandatory = $true)]
    [ResourceItemMapping(EntityType = [ResourceTypes]::SoftwareLicense, Role = 'OfficeActivation')]
    [Hashtable]
    $OfficeLicense
)

#Requires -Modules Rps-API
```

Legacy Runbooks (Workflows) Guidance

Prior to RPS v2.1, all RPS Runbooks were authored as PowerShell workflows instead of pure PowerShell scripts. Some RPS Runbooks are still Workflows, and they require special guidance.

PSDisableSerializationPreference

PowerShell Workflows, and by inheritance TMS Runbooks, serialize objects without methods. This can limit the usability of some RPS objects' method calls. To use these methods, the `PSDisableSerializationPreference` needs to be set to true.

```
$PSDisableSerializationPreference = $true
```

Workflow Naming

As a matter of consistency and support for the DSC resource managing RPS, the Workflow name must match the script name.

Registering Runbooks in RPS

Runbooks in the `/Runbooks` folder are automatically imported into RPS during install, via the `/Setup/Configuration/Import-RpsTasks.ps1` script. Once a full RPS Node is operational and TMS is running, any new or modified scripts in the runbooks folder will be automatically registered and imported into RPS and TMS, using DSC as the folder monitor.

NOTE

Runbooks are automatically available to TMS once they are in the Runbook Folder and a corresponding TaskItem is created.

Example:

```
# Create TaskItem for runbook Wait-TargetReady.ps1
New-RpsTaskItem -WorkflowName "Wait-TargetReady"
```

More Resources

- [Authoring RPS DSC Resources](#)
- [RPS Tasking Guide](#)

Introduction to RBAC in RPS

Last updated on December 30, 2020.

Last Reviewed and Approved on PENDING REVIEW

Introduction

Access management to resources is a critical function for any organization that has multiple types of users accessing a system. Role-Based Access Control (RBAC) helps you manage who has access to resources, what they can do with those resources, and what areas they have access to. RBAC is an authorization system that provides fine-grained access management of the Rapid Provisioning System (RPS) resources.

The way you control access to resources using RPS RBAC is to create role assignments. This is a key concept to understand – it's how permissions are enforced. A role assignment consists of three primary elements: Users, Roles, and Security Rights. The user is a member of a role. A role is what has the security right assigned to it.

RPS uses an RBAC approach to restrict system access only to authorized users. With RPS, roles are predefined such as administrator, or patch creator. These roles are a collection of different security rights. A user is placed in a particular role depending on the function(s) they need to perform. A user assigned to zero roles will not have any access to RPS and a user can also be assigned to multiple roles if they need to perform multiple duties in RPS. User and role assignment is dynamic and data driven. Users can be assigned and unassigned to different roles at any time without reinstallation, allowing users permissions in the system to grow and shrink with their assigned duties.

RPS roles are built around job duties and implement the principle of least privilege, meaning each role is designed to only allow a user to perform the necessary functions related to that role and nothing more.

RBAC Fundamentals

Access and Interfaces

Management of users, roles, and assignments is done either via the RPS Web Graphic User Interface (GUI) or via PowerShell scripts. Regardless of your preferred RBAC management method, either done via GUI or PowerShell, RBAC has been designed to have feature parity across both human interfaces. Via PowerShell, user creation and role assignment can be easily added to scripts to help with automation. However, the RPS Web GUI makes it easy for anyone to manage users and roles.

Users

Users are based on domain and local Windows users. To be assigned a role, Users must still be enrolled in RPS.

Authentication

RPS does not perform user authentication it only handles authorization. All RPS authentication is handled by Windows. All user accounts must be either local or domain Windows accounts to be authenticated and used by RPS. The RPS RBAC access management system does not store any user passwords and it does not manage RPS user passwords. All password management is handled through either the domain or Windows.

Role Assignment

Users are added to roles via the RPS Web GUI or via PowerShell Cmdlet. See [How to Add and Remove User Roles](#). A user must have the proper privileges to add and remove users from roles. Users cannot add and remove themselves from roles, unless they are a super admin.

Installation

On installation of RPS pre-defined roles are imported via a data import process.

How to Add and Remove User Roles

Last updated on July 30, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The Rapid Provisioning System (RPS) uses a role-based authorization approach called Role-Based Access Control (RBAC) to restrict system access only to authorized users. The way you control access to resources using RPS RBAC is to assign users to roles. This is a key concept to understand—it is how permissions are enforced.

A role assignment consists of two primary elements: Users and Roles. The user is a member of a role. A role is what has the security right assigned to it. The first layer of security is to ensure only people who require access to RPS have user accounts. This article will provide the how-to instructions on adding or removing a role from a user in RPS.

Assumptions

1. You have read the [Introduction to RBAC](#) article and have a basic understanding of Role-Based Access Control.
2. You have access to the RPS Graphic User Interface (GUI or UI).
3. You are assigned to the appropriate role to make the changes you intend to make.

RBAC Fundamentals

RBAC Terms and Definitions

RBAC TERM	DEFINITION
Role-Based Access Control (RBAC)	An authorization system that provides fine-grained access management of RPS resources.
Role	A collection of permissions.
User	A logical representation of a person or persona acting as a consumer (of the application). Most users are objects found in the Active Directory; however, some personas—such as service accounts—are treated as users but are not found in the Active Directory.
Local User	The Local User is stored on the computer's local hard disk. Changes made to the Local User profile are specific to the user and to the computer on which the changes are made.

RBAC Concepts

- RPS uses Windows to perform its authentication, but has internal roles for authorization.
- To add users to RPS, the user must be a local or domain account and must be accessible via the system running RPS.
- When a user is added, they will not have any rights or privileges until they are assigned to a role.
- If a local or domain account is suspended or deleted, that account will be unable to access RPS.

RPS Roles

The following pre-defined roles are available upon installation of RPS:

ROLE NAME	DESCRIPTION
AD Admin	Allows full access to Active Directory items.
Audit Entry Viewer	Can view audit entries.
Certificate Admin	Full access to certificates.
Certificate Read	Can read certificates.
Credential Admin	full access to credential.
Credential Read	Allows read access to credential.
DSC Admin	Full DSC access.
DSC Partial Assigner	Can assign DSC partials.
Full Read	Full read can read all data in RPS.
Network Admin	Allows full access to network items.
Patch Admin	Patch admin has full control over patching system.
Patch Stream Approver	Can approve patch streams.
Patch Stream Creator	Can create patch streams.
Patch Stream Scheduler	Can scheduler patch streams.
Patch Viewer	Can view all patch data.
RBAC Admin	Full RBAC control.
RPS Admin	Full control over RPS, except for Security.
Super Admin	Super admin has full control over RPS.
Sync and CDN Admin	Can Administer Sync and CDN.
System Admin	Allows full access to Virtual machines and related items.

How to Add a User to a Role in RPS in the Web User Interface

In RPS you can add single users or multiple users at the same time directly to roles.

To add a user to a role assignment:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Roles**.

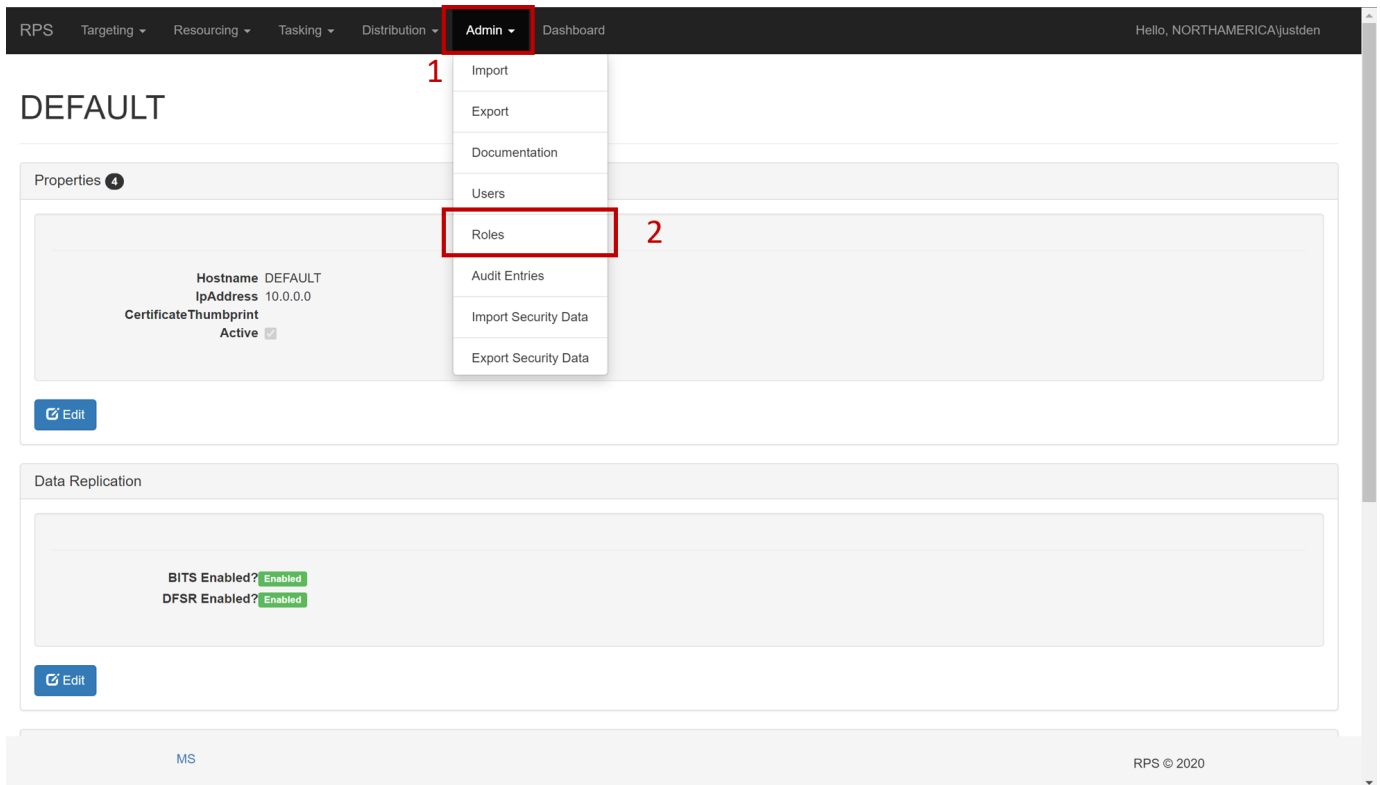


Figure 1: Select **Admin** and then **Roles** in the dropdown.

3. Click on the **[Role]** you would like to add a user to.

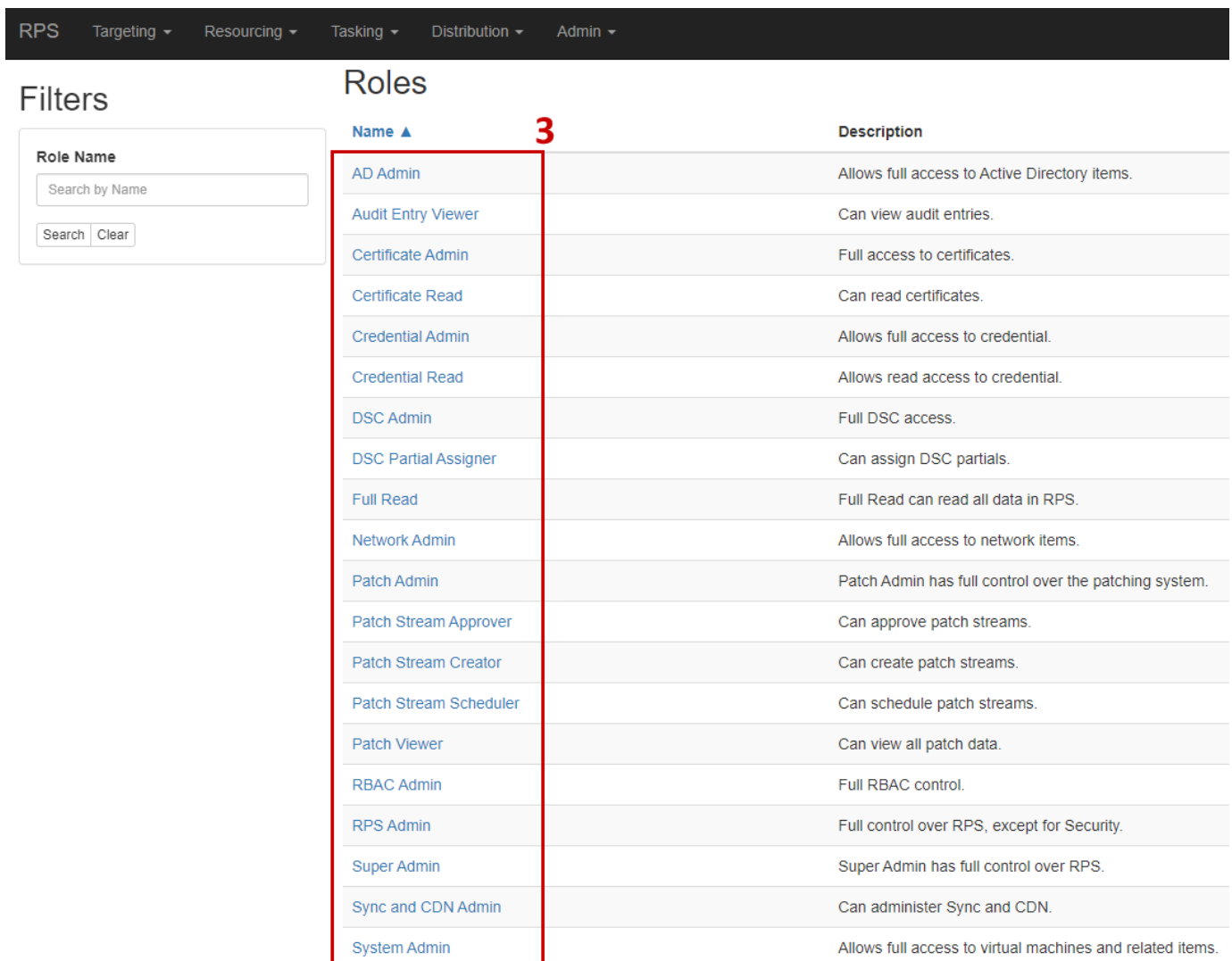


Figure 2: Select a role to add the user to.

4. Scroll to the bottom of the page. In the 'Members' section, click **Edit**.

RPS Targeting ▾ Resourcing ▾ Tasking ▾ Distribution ▾ Admin ▾ Dashboard

System Admin

Description: Allows full access to Virtual machines and related items.

Security Rights

Node 1

Rights	AnyNode	Property Permissions
FullControl	False	View Property Permissions

ResourceItem 2

Type	Rights	Property Permissions
HyperV	FullControl	View Property Permissions
VMTemplate	FullControl	View Property Permissions

TargetGroup 1

Type	Rights	Property Permissions
Patchable_Targets	FullControl	View Property Permissions

TargetItem 4

Type	Rights	Property Permissions
VirtualMachine	FullControl	View Property Permissions
Computer	FullControl	View Property Permissions
NetworkConfiguration	FullControl	View Property Permissions
Processor	FullControl	View Property Permissions

Members

User Name ▲	Domain Name
Edit	

[Back](#)

MS

Figure 3: In the 'Members' section, click **Edit**.

5. Select the **[User]** from the 'Available Users' section you would like to add to the role.
6. Click the **double right arrow >>**.

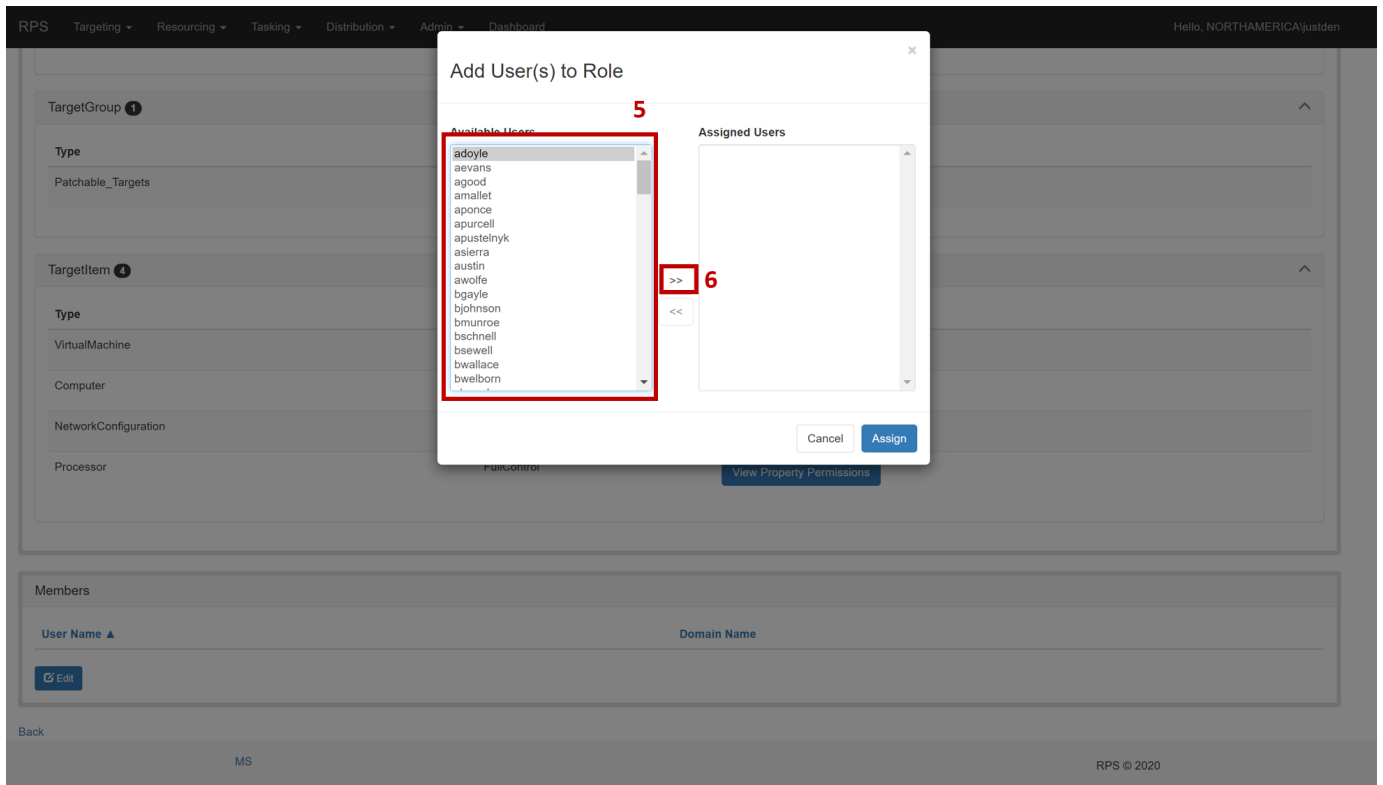


Figure 4: Select the **[User]** from the 'Available Users' section to add to the role and click the **double right arrow >>**.

NOTE

To add multiple users, repeat steps 5 and 6.

7. Click **Assign**.

Add User(s) to Role

Available Users

- aevans
- agood
- amallet
- aponce
- apurcell
- apustelnyk
- asierra
- austin
- awolfe
- bgayle
- bjohnson
- bmunroe
- bschnell
- bsewell
- bwallace
- bwelborn
- cbeard

Assigned Users

- adoyle

>>

<<

7

Cancel

Assign

Figure 5: Click **Assign**.

8. The user you added to the role should now display under that role's members.

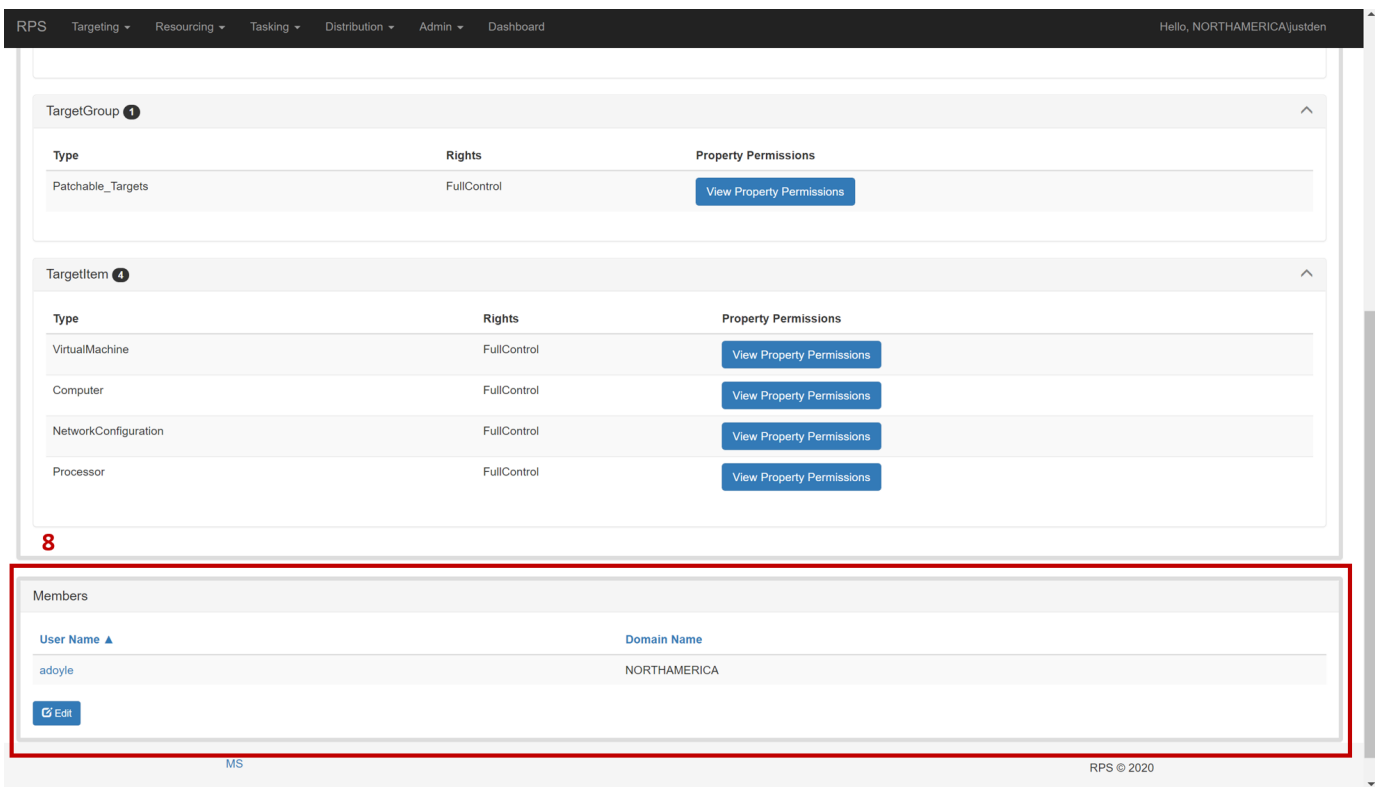


Figure 6: The added user displays under the role's 'Members' section.

How to Remove a User From a Role in RPS Using the Web User Interface

To remove a user from a role assignment:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Roles**.

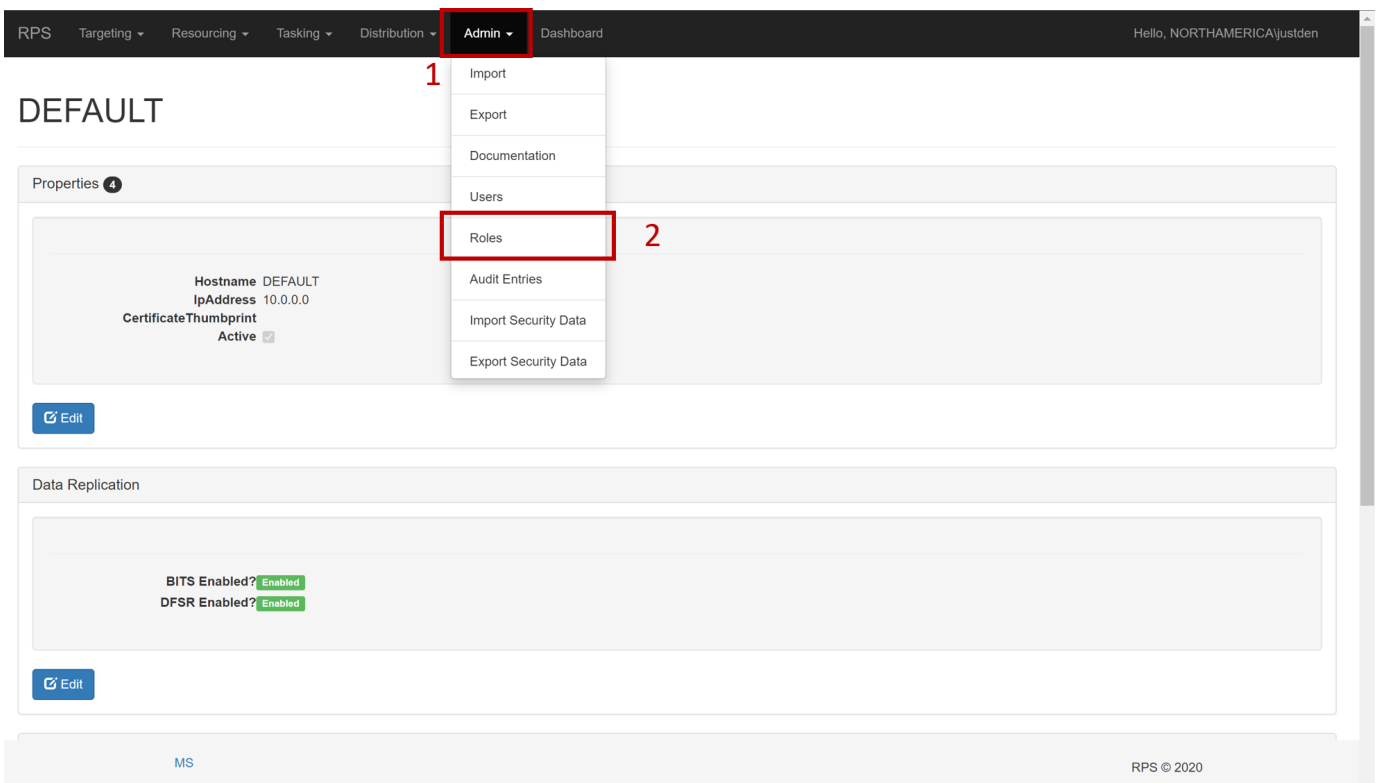


Figure 7: Select **Admin** and then **Roles** in the dropdown.

3. Click on the **[Role]** you would like to remove a user from.

Filters

Role Name

Roles

Name ▲	Description
AD Admin	Allows full access to Active Directory items.
Audit Entry Viewer	Can view audit entries.
Certificate Admin	Full access to certificates.
Certificate Read	Can read certificates.
Credential Admin	Allows full access to credential.
Credential Read	Allows read access to credential.
DSC Admin	Full DSC access.
DSC Partial Assigner	Can assign DSC partials.
Full Read	Full Read can read all data in RPS.
Network Admin	Allows full access to network items.
Patch Admin	Patch Admin has full control over the patching system.
Patch Stream Approver	Can approve patch streams.
Patch Stream Creator	Can create patch streams.
Patch Stream Scheduler	Can schedule patch streams.
Patch Viewer	Can view all patch data.
RBAC Admin	Full RBAC control.
RPS Admin	Full control over RPS, except for Security.
Super Admin	Super Admin has full control over RPS.
Sync and CDN Admin	Can administer Sync and CDN.
System Admin	Allows full access to virtual machines and related items.

Figure 8: Select a role from which to remove the user.

4. Scroll to the bottom of the page. In the 'Members' section, click **Edit**.

RPS Targeting ▾ Resourcing ▾ Tasking ▾ Distribution ▾ Admin ▾

Type	Rights	Property Permissions
Patchable_Targets	FullControl	View Property Permissions

TargetItem 5

Type	Rights	Property Permissions
Processor	FullControl	View Property Permissions
Computer	FullControl	View Property Permissions
VirtualMachine	FullControl	View Property Permissions
Drive	FullControl	View Property Permissions
NetworkConfiguration	FullControl	View Property Permissions

Members

User Name ▲	Domain Name
RpsAdmin	rps

[Edit](#) 4

Figure 9: In the 'Members' section, click **Edit**.

5. Select the **[User]** you would like to remove from the role in the 'Assigned Users' section.
6. Click the **double left arrow <<**.

NOTE

To remove multiple users, repeat steps 5 and 6.

7. Click **Assign**.

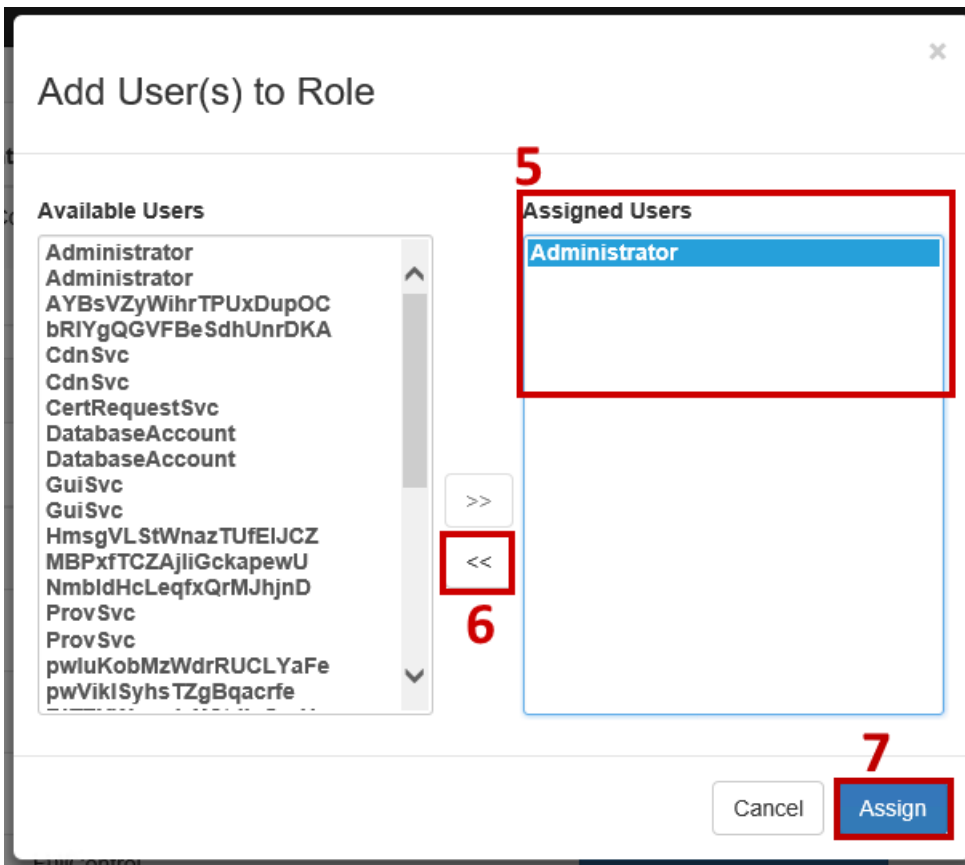


Figure 10: Select the [User] to remove from the role, click the **double left arrow <<**, and click **Assign**.

8. The user(s) you removed from the role should no longer display under that role's 'Members' section.

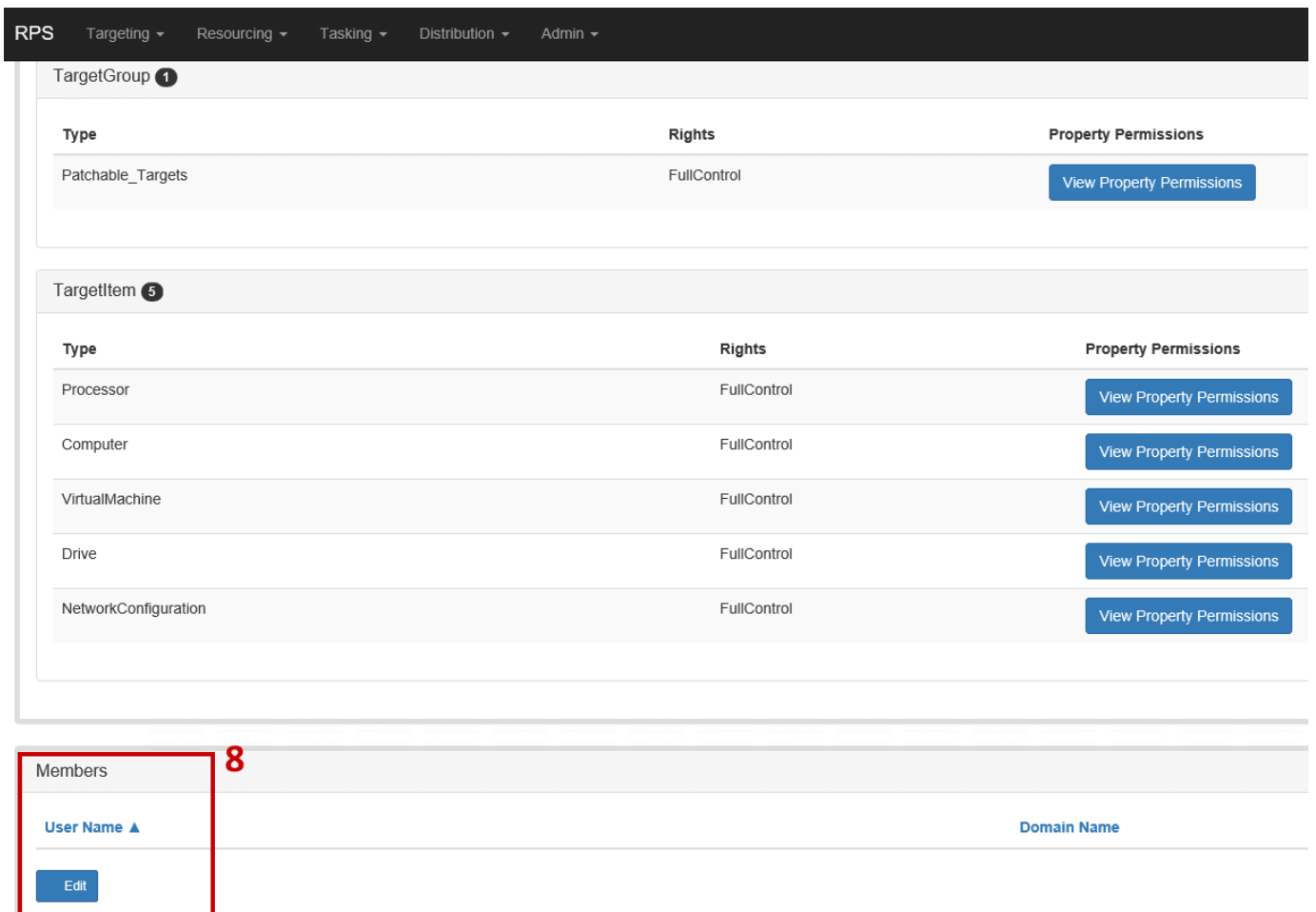


Figure 11: The removed user no longer displays in the selected role's 'Members' section.

How to Add a User to a Role, Remove a User From a Role, or View Role Assignments in RPS Using PowerShell

Using PowerShell for RBAC functions in RPS gives you the ability to do the following actions:

1. [Add a user to a role.](#)
2. [Remove a user from a role.](#)
3. Find a role assignment by:
 - o [User](#)
 - o [Role](#)
 - o [ID](#)

Prerequisite

⚠ IMPORTANT

In order to use the subsequent code snippets, the `$user` and `$role` variables must be set. This **MUST** be done one time per PowerShell session.

In PowerShell, enter the following cmdlets:

```
$user = Get-RpsUser -UserName "user1"  
$role = Get-RpsRole -Name "Patch Admin"
```

⚠ IMPORTANT

The `-UserName` and `-Name` values used in the example above are arbitrary and should be replaced with the username and role name being acted upon.

Adding a User to a Role: The `Add-RpsRoleAssignment` Cmdlet

Parameters

1. **User** – Specified user.
2. **Role** – Role to be added.

How to Use the Cmdlet to Add a User to a Role

In PowerShell, enter the following cmdlet with the information appropriate for your user and role you would like to add, per the parameters above.

```
Add-RpsRoleAssignment -User $user -Role $role
```

```

PS C:\Windows\system32> $user = Get-RpsUser -Username "RpsAdmin"
$role = Get-RpsRole -Name "System Admin"

Add-RpsRoleAssignment -User $user -Role $role

Id                : fba816aa-d4a6-41a8-8429-dc60f92b4e82
Name              : System Admin
Description       : Allows full access to Virtual machines and related items.
GlobalPermission  : None
SecurityRights    : {0a6a0b93-efd2-418e-b1a7-bd0fbd75f6d5, 18a1ef99-4453-4bb5-8802-7bdbdddafc65,
                  1ad8f19d-7c00-411b-b2f9-4734f317d7f1, 1cf87d5f-b8fa-4fb7-b153-da9444a36081...}
CreatedUser       :
CreatedDateUtc   : 1/1/0001 12:00:00 AM
UpdatedUser       :
UpdatedDateUtc   : 1/1/0001 12:00:00 AM
isSystem         : False
isNotified       : False
RoleAssignments  : {188a4665-624c-6477-8299-455415511893}
Users            : {78e7701c-8737-b914-9b63-af0c6ea1f300}
CurrentVersion   : 10290
ModifiedDateUTC  : 1/1/0001 12:00:00 AM
CreatedVersion   :
ChangeType       : I
ToggleUpdate     : False
DeletedByForce   : False

```

Figure 12: PowerShell running as Administrator with the above `Add-RpsRoleAssignment` code snippet entered.

Removing a User From a Role in PowerShell: The `Remove-RpsRoleAssignment` Cmdlet

Parameters

1. **User** – Specified user.
2. **Role** – Role to be removed.

How to Use the Cmdlet to Remove a User From a Role

In PowerShell, enter the following cmdlet with the information appropriate for your user and role you would like to remove, per the parameters identified above.

```
Remove-RpsRoleAssignment -User $user -Role $role
```

```

PS C:\Windows\system32> Remove-RpsRoleAssignment -User $user -Role $role
PS C:\Windows\system32>

```

Figure 13: PowerShell running as Administrator with the above `Remove-RpsRoleAssignment` code snippet entered.

How to Find a Role Assignment in PowerShell: The `Get-RpsRoleAssignment` Cmdlet

This cmdlet can get role assignment by user, role, or role ID.

Parameters

1. **User** – Specified user.
2. **Role** – Specified role.
3. **Id** – ID of the Role assigned.

How to Get a Role Assignment by User

In PowerShell, enter the following cmdlet with the information appropriate for your user you would like to view role assignment for, per the parameters identified above.

```
Get-RpsRoleAssignment -User $user
```

```
PS C:\Windows\system32> Get-RpsRoleAssignment -User $user

Id           : 50f9c4c5-dc0d-aebe-341f-53400fc11678
User_Id      : 78e7701c-8737-b914-9b63-af0c6ea1f300
Role_Id      : 75eaa3c5-8d70-4fb9-8ba5-8cde80e86823
User         : Rps.Api.RBAC.User
Role         : Rps.Api.RBAC.Role
CreatedUser   :
CreatedDateUtc : 1/1/0001 12:00:00 AM
UpdatedUser   :
UpdatedDateUtc : 1/1/0001 12:00:00 AM
CurrentVersion : 10290
ModifiedDateUTC : 1/1/0001 12:00:00 AM
CreatedVersion :
ChangeType    : I
ToggleUpdate  : False
DeletedByForce : False
```

Figure 14: Using the `Get-RpsRoleAssignment` cmdlet to view role assignment by specified User parameter.

How to Get a Role Assignment by Role

In PowerShell, enter the following cmdlet with the information appropriate for your role you would like to view assignment for, per the parameters identified above.

```
Get-RpsRoleAssignment -Role $role
```

```
PS C:\Windows\system32> Get-RpsRoleAssignment -Role $role

Id           : 188a4665-624c-6477-8299-455415511893
User_Id      : 78e7701c-8737-b914-9b63-af0c6ea1f300
Role_Id      : fba816aa-d4a6-41a8-8429-dc60f92b4e82
User         : Rps.Api.RBAC.User
Role         : Rps.Api.RBAC.Role
CreatedUser   : rps\RpsAdmin
CreatedDateUtc : 8/4/2021 3:48:56 PM
UpdatedUser   : rps\RpsAdmin
UpdatedDateUtc : 8/4/2021 3:48:56 PM
CurrentVersion : 2670669
ModifiedDateUTC : 1/1/0001 12:00:00 AM
CreatedVersion :
ChangeType    : I
ToggleUpdate  : False
DeletedByForce : False
```

Figure 15: Using the `Get-RpsRoleAssignment` cmdlet to view role assignment by specified Role parameter.

How to Get a Role Assignment by ID

In PowerShell, enter the following cmdlet with the information appropriate for your role ID you would like to view assignment for, per the parameters identified above.

```
Get-RpsRoleAssignment -Id $lookupId
```

```
PS C:\Windows\system32> Get-RpsRoleAssignment -Id $lookupId

Id           : 188a4665-624c-6477-8299-455415511893
User_Id      : 78e7701c-8737-b914-9b63-af0c6ea1f300
Role_Id      : fba816aa-d4a6-41a8-8429-dc60f92b4e82
User         : Rps.Api.RBAC.User
Role         : Rps.Api.RBAC.Role
CreatedUser   : rps\RpsAdmin
CreatedDateUtc : 8/4/2021 3:48:56 PM
UpdatedUser   : rps\RpsAdmin
UpdatedDateUtc : 8/4/2021 3:48:56 PM
CurrentVersion : 2670669
ModifiedDateUTC : 1/1/0001 12:00:00 AM
CreatedVersion :
ChangeType    : I
ToggleUpdate  : False
DeletedByForce : False
```

Figure 16: Using the `Get-RpsRoleAssignment` cmdlet to view role assignment by specified (Role) Id parameter.

How to Get All Role Assignments

In PowerShell, enter the following cmdlet:

```
Get-RpsRoleAssignment
```

```
PS C:\Windows\system32> Get-RpsRoleAssignment

Id                : 188a4665-624c-6477-8299-455415511893
User_Id           : 78e7701c-8737-b914-9b63-af0c6ea1f300
Role_Id           : fba816aa-d4a6-41a8-8429-dc60f92b4e82
User              : Rps.Api.RBAC.User
Role              : Rps.Api.RBAC.Role
CreatedUser       : rps\RpsAdmin
CreatedDateUtc    : 8/4/2021 3:48:56 PM
UpdatedUser       : rps\RpsAdmin
UpdatedDateUtc    : 8/4/2021 3:48:56 PM
CurrentVersion    : 2670669
ModifiedDateUTC   : 1/1/0001 12:00:00 AM
CreatedVersion    :
ChangeType        : I
ToggleUpdate      : False
DeletedByForce    : False

Id                : 2acde2e7-9531-0ba3-5a15-34878b87c008
User_Id           : b95cba49-6a90-0185-4f64-918532fa95c8
Role_Id           : 75eaa3c5-8d70-4fb9-8ba5-8cde80e86823
User              : Rps.Api.RBAC.User
Role              : Rps.Api.RBAC.Role
CreatedUser       :
CreatedDateUtc    : 1/1/0001 12:00:00 AM
UpdatedUser       :
UpdatedDateUtc    : 1/1/0001 12:00:00 AM
CurrentVersion    : 10290
ModifiedDateUTC   : 1/1/0001 12:00:00 AM
CreatedVersion    :
ChangeType        : I
ToggleUpdate      : False
DeletedByForce    : False

Id                : 2f42a5f1-b30c-1d8b-7eae-5d43b129e54c
User_Id           : 38e2f5b7-bad6-15e2-5c65-73ba01072d02
Role_Id           : 75eaa3c5-8d70-4fb9-8ba5-8cde80e86823
User              : Rps.Api.RBAC.User
Role              : Rps.Api.RBAC.Role
CreatedUser       :
CreatedDateUtc    : 1/1/0001 12:00:00 AM
UpdatedUser       :
UpdatedDateUtc    : 1/1/0001 12:00:00 AM
```

Figure 17: Using the `Get-RpsRoleAssignment` cmdlet to view all role assignments.

How to Manage User Roles with PowerShell

Last updated on March 1, 2021.

Last Reviewed and Approved on PENDING REVIEW

This document describes the step-by-step instructions from end to end managing assigning users enrolled in RPS to specific roles in RPS.

How to Add a local/domain user to RPS Role using PowerShell cmdlets

1. Import the RPS API module

```
Import-Module C:\ContentStore\Modules\Rps-Api
```

2. Get a user from RPS

```
$userAdmin = Get-RpsUser -UserName Admin -DomainName Company
```

3. Get a RPS Role

Parameter options for the **Get-RpsRole** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
Id	Guid- Optional	Id of Role to retrieve.
Name	String- Optional	Name of Role to retrieve.
<i>Empty</i>		Retrieve all roles.

```
$role = Get-RpsRole -Name Patch Admin
```

4. Add a user to a role

Parameter options for the **Add-RpsRoleAssignment** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
User	User- Required	User to be assigned to role.
Role	Role- Required	Role to have user assigned.

```
$role = Add-RpsRoleAssignment -user $userAdmin -role $role
```

NOTE

After the user is added to the role, they will have all the rights and privileges associated with that role

How to Remove a local/domain user from RPS Role using PowerShell cmdlets

1. Import the RPS API module

```
Import-Module C:\ContentStore\Modules\Rps-Api
```

2. Get a user from RPS

```
$userAdmin = Get-RpsUser -UserName Admin -DomainName Company
```

3. Get a RPS Role

Parameter options for the **Get-RpsRole** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
Id	Guid- Optional	Id of Role to retrieve.
Name	String- Optional	Name of Role to retrieve.
<i>Empty</i>		Retrieve all roles.

```
$role = Get-RpsRole -Name Patch Admin
```

4. Remove a user from a role

Parameter options for the **Remove-RpsRoleAssignment** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
User	User- Required	User to be removed from role.
Role	Role- Required	Role to have user removed from.

```
$role = Remove-RpsRoleAssignment -user $userAdmin -role $role
```

WARNING

After the user is no longer in the role, they will immediately lose all rights and privileges associated with that role

RBAC: How to Import and Export Users with the Web User Interface

Last updated on February 4, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The Rapid Provisioning System (RPS) uses a role-based authorization approach called Role-Based Access Control (RBAC) to restrict system access only to authorized users. The way you control access to resources using RPS RBAC is to assign users to roles. This is a key concept to understand – it's how permissions are enforced.

A role assignment consists of two primary elements: Users and Roles. The user is a member of a role. A role is what has the security right assigned to it. The first layer of security is to ensure only people who require access to RPS have user accounts. This article will provide the How-To process for importing and exporting users and their assigned roles to and from RPS.

Assumptions

1. You have read the [Introduction to RBAC](#) article and have a basic understanding of Role-Based Access Control.
2. You have access to the RPS Graphic User Interface (GUI or UI).
3. You are assigned to the appropriate role to make the changes you intend to make.
4. If Importing Users, you have an existing XML file with appropriate data.

RBAC Fundamentals

RBAC Terms and Definitions

- **Role-Based Access Control (RBAC)** – an authorization system that provides fine-grained access management of RPS resources.
- **Role** – a collection of permissions.
- **User** – A logical representation of a person or persona acting as a consumer (of the application). Most users are objects found in Active Directory, however some personas that are treated as users, such as service accounts are users that are not found in Active Directory.

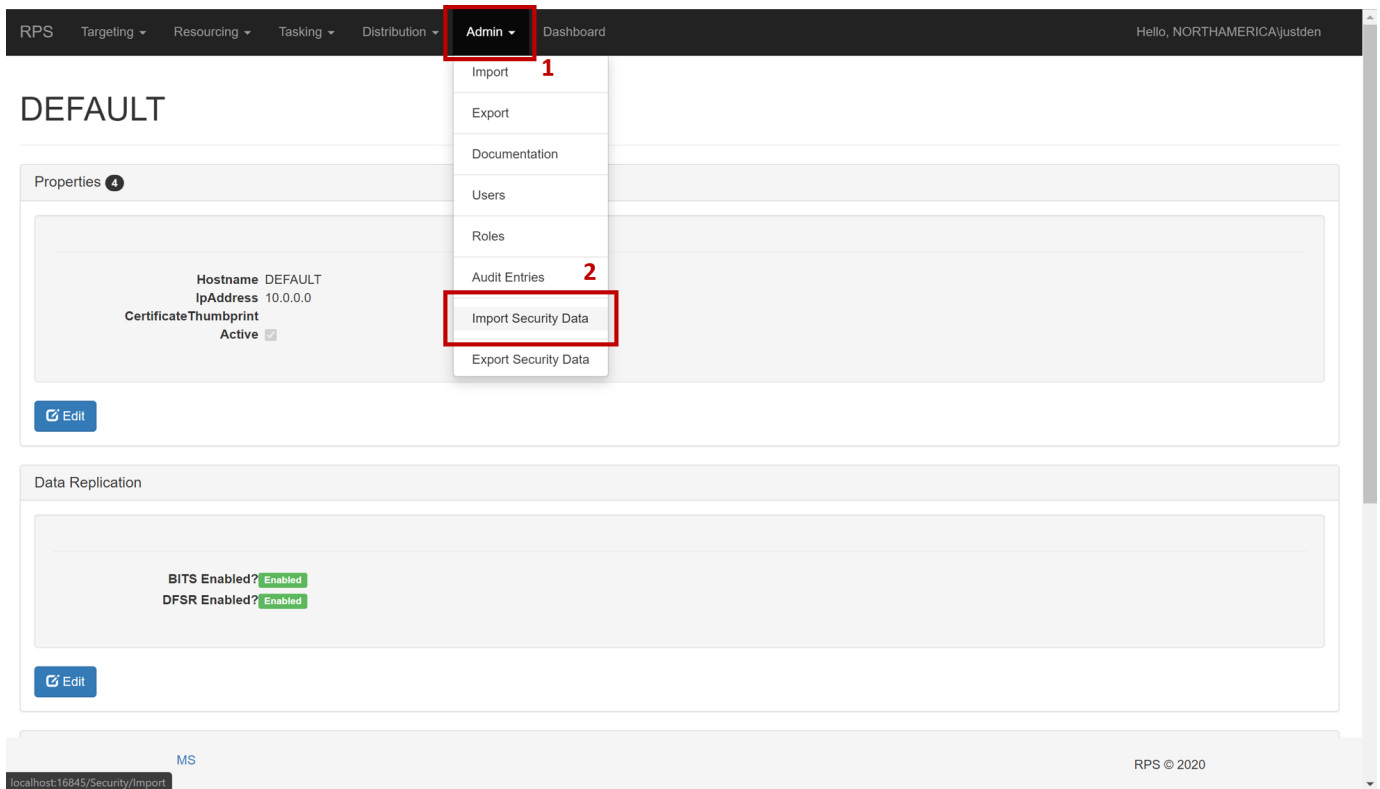
RBAC Concepts

- RPS uses Windows to perform its authentication but has internal roles for authorization.
- To add users to RPS the user must be a local or domain account and must be accessible via the system running RPS.
- Once a user is added they will not have any rights or privileges until they are assigned to a role.
- If a local or domain account is suspended or deleted, that account will be unable to access RPS.

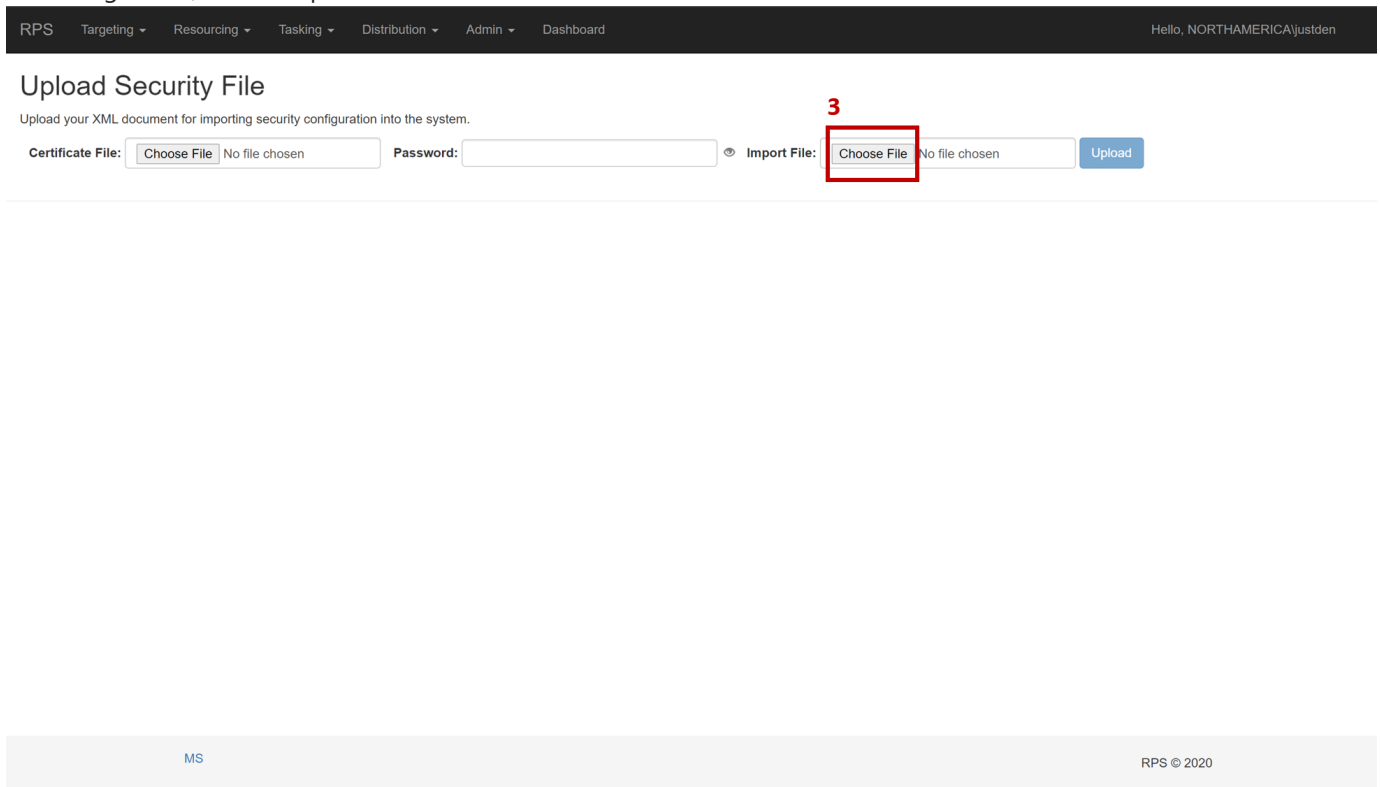
How to Import a List of Local/Domain Users to RPS

To import a user list XML file:

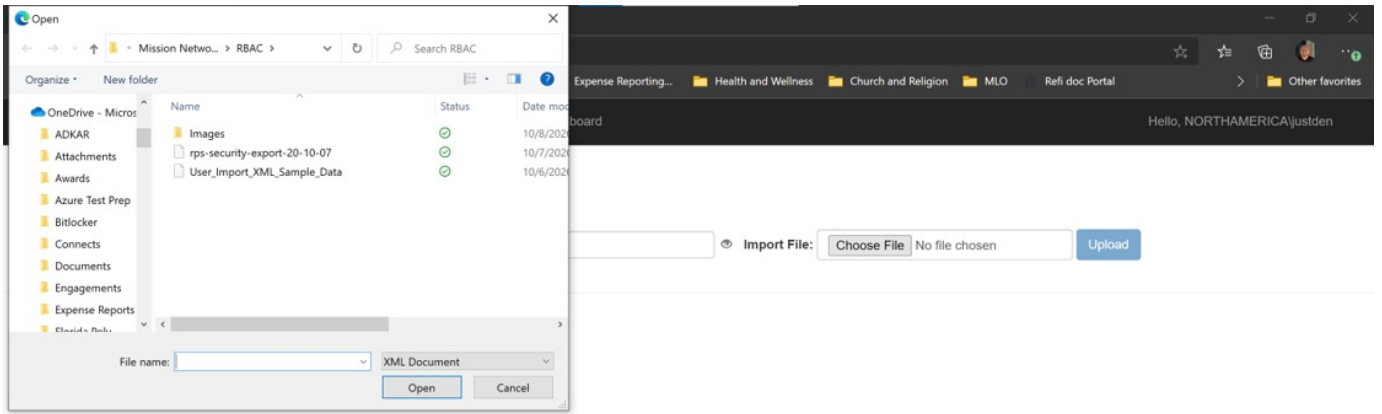
1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Import Security Data**.



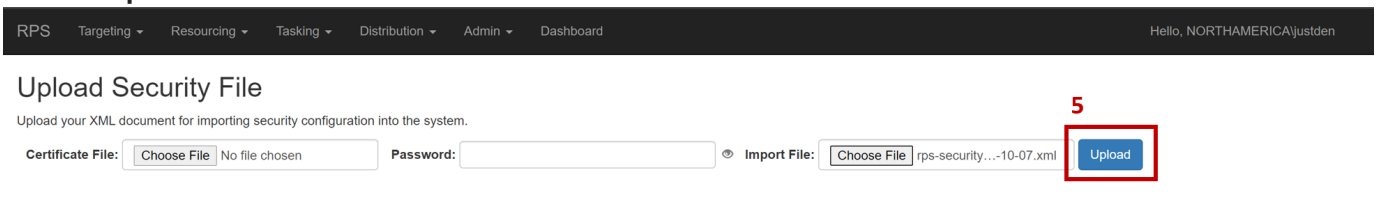
3. On the right side, locate 'Import File'. Click the **Choose File** button.



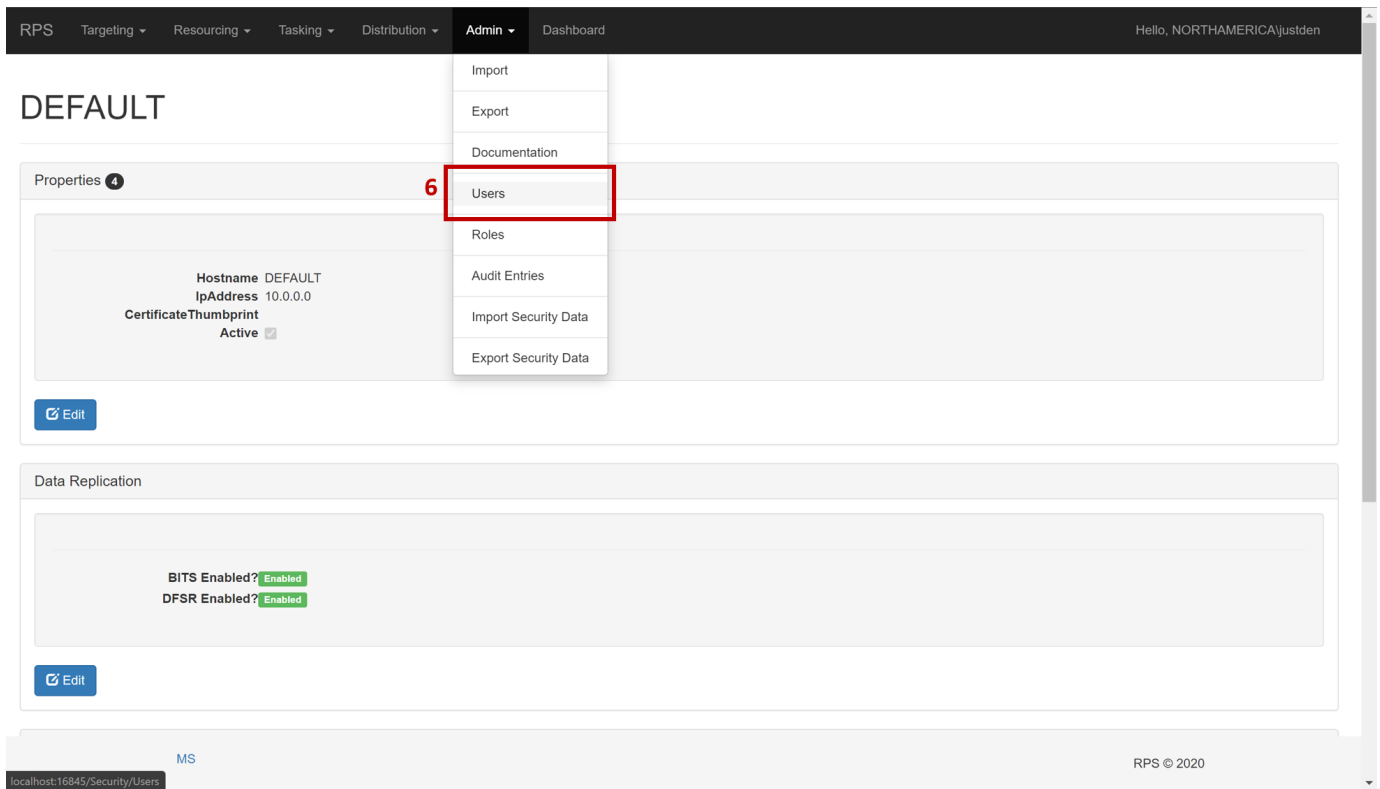
4. After selecting your **file...**



5. ...click on **Upload**.



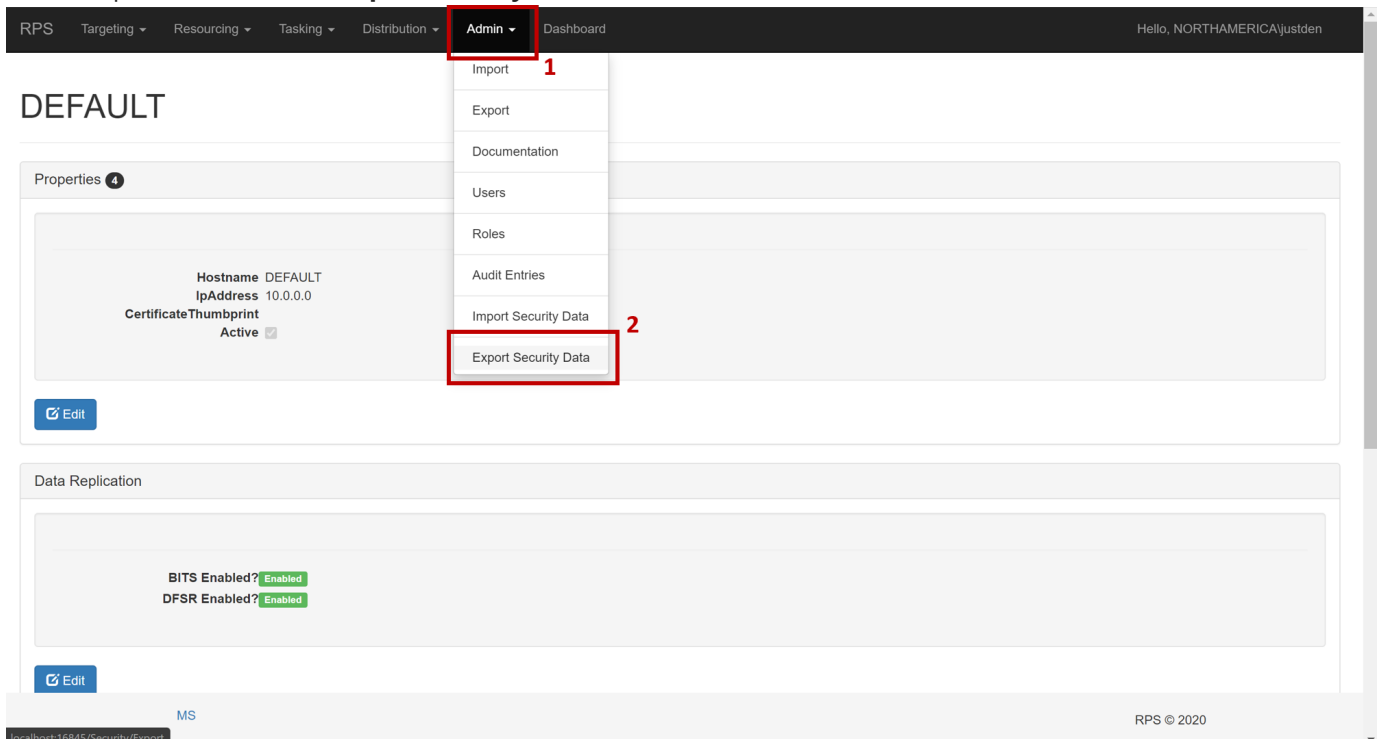
6. To verify your data has imported, you can select **Users** from the Admin dropdown navigation bar and validate the imported users are there.



How to Export a List of Local/Domain Users to RPS

To export a user list XML file:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Export Security Data**.



3. Select the **Users** you would like to export. There are two options to select which users you would like to export.
 - o 3a. Select this box to select all users for export.
 - o 3b. Select individual users' boxes to export specific users.
4. (**Optional**) If you would like to encrypt the user export file, you may use a privately generated certificate and upload it here.
5. (**Optional**) If you would like to export the user's role assignments, check this box.
6. Click **Export Selected**.

Export Data

You can select what data you would like to export below.

Certificate File No file chosen Include Assignments: Cancel

Select All

Users

Optional

	UserName	DomainName
<input type="checkbox"/>	jtorres	NORTHAMERICA
<input type="checkbox"/>	asierra	NORTHAMERICA
<input type="checkbox"/>	bmunroe	NORTHAMERICA
<input type="checkbox"/>	kpreston	EUROPE
<input type="checkbox"/>	dkline	EUROPE
<input type="checkbox"/>	ctussey	NORTHAMERICA
<input type="checkbox"/>	dmarie	NORTHAMERICA
<input type="checkbox"/>	rsander	NORTHAMERICA
<input type="checkbox"/>	kbecker	FVEY
<input type="checkbox"/>	bjohnson	NORTHAMERICA
<input type="checkbox"/>	austin	NORTHAMERICA
<input type="checkbox"/>	bsewell	NORTHAMERICA

MS

RPS © 2020

7. Save the file.

RBAC: How to Add and Remove Users with PowerShell

Last updated on February 4, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The Rapid Provisioning System (RPS) uses a role-based authorization approach called Role-Based Access Control (RBAC) to restrict system access only to authorized users. The way you control access to resources using RPS RBAC is to assign users to roles. This is a key concept to understand – it's how permissions are enforced.

A role assignment consists of two primary elements: Users and Roles. The user is a member of a role. A role is what has the security right assigned to it. The first layer of security is to ensure only people who require access to RPS have user accounts. This article will provide the How-To process for adding and removing Users from RPS using PowerShell.

Assumptions

1. You have read the [Introduction to RBAC](#) article and have a basic understanding of Role-Based Access Control.
2. You have access to the RPS Graphic User Interface (GUI or UI).
3. You are assigned to the appropriate role to make the changes you intend to make.

RBAC Fundamentals

RBAC Terms and Definitions

- **Role-Based Access Control (RBAC)** – an authorization system that provides fine-grained access management of RPS resources.
- **Role** – a collection of permissions.
- **User** – A logical representation of a person or persona acting as a consumer (of the application). Most users are objects found in Active Directory, however some personas that are treated as users, such as service accounts are users that are not found in Active Directory.

RBAC Concepts

- RPS uses Windows to perform its authentication but has internal roles for authorization.
- To add users to RPS the user must be a local or domain account and must be accessible via the system running RPS.
- Once a user is added they will not have any rights or privileges until they are assigned to a role.
- If a local or domain account is suspended or deleted, that account will be unable to access RPS.

How to Add a Local/Domain User to RPS Using PowerShell Cmdlets

1. Import the RPS API module.

```
Import-Module C:\ContentStore\Modules\Rps-API
```

2. Add the user to RPS.

Parameter options for the **New-RpsUser** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
UserName	string-Required	Domain/Local Username of account.
DomainName	string-Optional	Domain Name or Machine Name; if no value is provided Domain will default to Machine Name.

```
New-RpsUser -UserName Admin -DomainName Company
```

NOTE

By Default, the New-RPSUser Command validates the user exists. If you would like to skip validation, add the `-force` flag.

After the user is added to RPS they can then have roles assigned to them using [Add-RpsRoleAssignment](#) Cmdlet. See: [How to Add and Remove User Roles](#)

How to Remove a Local/Domain User From RPS Using PowerShell Cmdlets

1. Import the RPS API module.

```
Import-Module C:\ContentStore\Modules\Rps-Api
```

2. Get all users from RPS.

Parameter options for the **Get-RpsUser** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
Id	Guid- Optional	Get user by ID.
UserName	string- Optional	Get user by UserName.
DomainName	string- Optional	Get user by DomainName.
<i>Empty</i>		Get all users.

```
$userAdmin = Get-RpsUser -UserName Admin -DomainName Company
```

3. Remove the user from RPS.

Parameter options for the **Remove-RpsUser** cmdlet are:

PARAMETER NAME	TYPE	DESCRIPTION
User	User- Required	User object to be removed from RPS.

```
Remove-RpsUser -User $userAdmin
```

WARNING

After the user is removed, they will no longer have any access to RPS.

RBAC: How to Add and Remove Users with the Web User Interface

Last updated on February 4, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The Rapid Provisioning System (RPS) uses a role-based authorization approach called Role-Based Access Control (RBAC) to restrict system access only to authorized users. The way you control access to resources using RPS RBAC is to assign users to roles. This is a key concept to understand – it's how permissions are enforced.

A role assignment consists of two primary elements: Users and Roles. The user is a member of a role. A role is what has the security right assigned to it. The first layer of security is to ensure only people who require access to RPS have user accounts. This article will provide the How-To process for adding and removing Users from RPS using the Web Interface.

Assumptions

1. You have read the [Introduction to RBAC](#) article and have a basic understanding of Role-Based Access Control.
2. You have access to the RPS Graphic User Interface (GUI or UI).
3. You are assigned to the appropriate role to make the changes you intend to make.

RBAC Fundamentals

RBAC Terms and Definitions

- **Role-Based Access Control (RBAC)** – an authorization system that provides fine-grained access management of RPS resources.
- **Role** – a collection of permissions.
- **User** – A logical representation of a person or persona acting as a consumer (of the application). Most users are objects found in Active Directory, however some personas that are treated as users, such as service accounts are users that are not found in Active Directory.

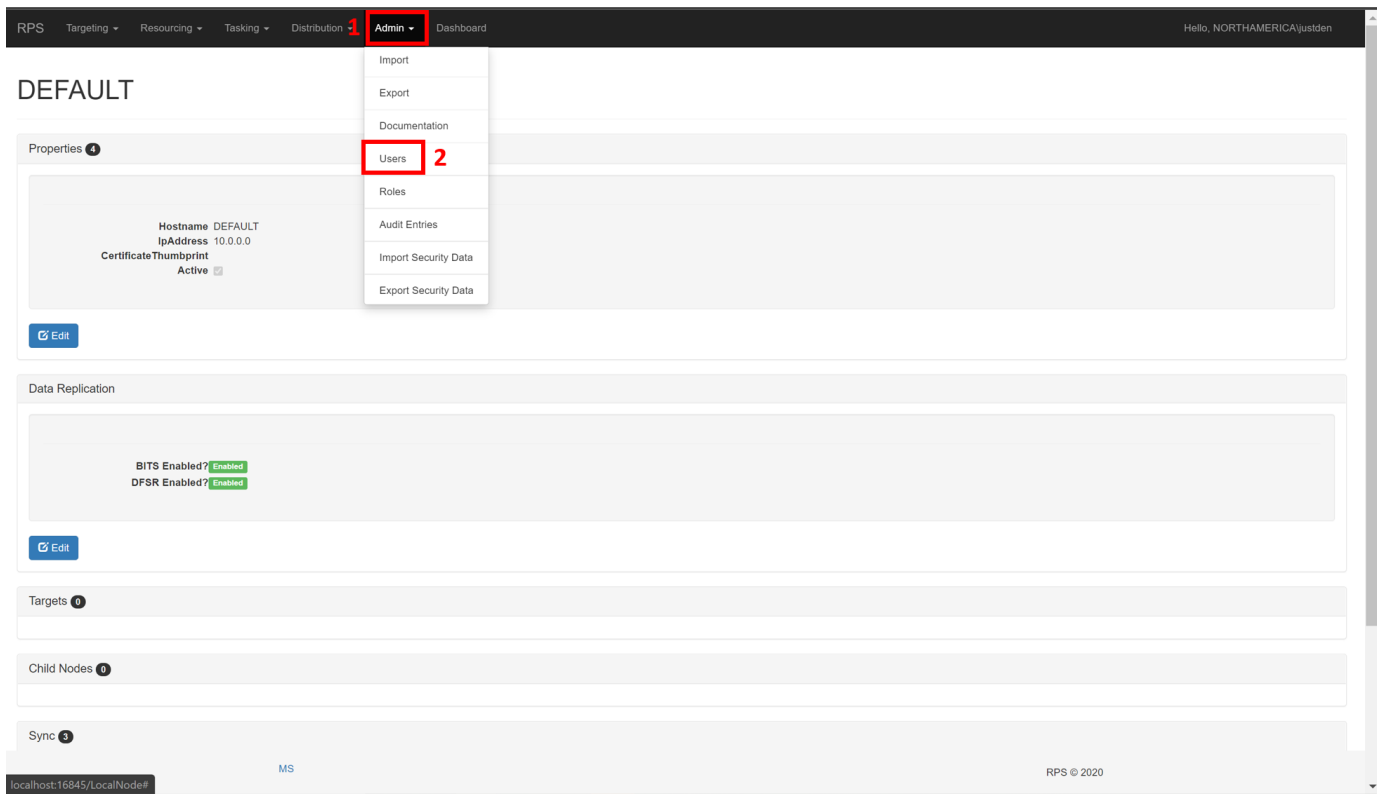
RBAC Concepts

- RPS uses Windows to perform its authentication but has internal roles for authorization.
- To add users to RPS the user must be a local or domain account and must be accessible via the system running RPS.
- Once a user is added they will not have any rights or privileges until they are assigned to a role.
- If a local or domain account is suspended or deleted, that account will be unable to access RPS.

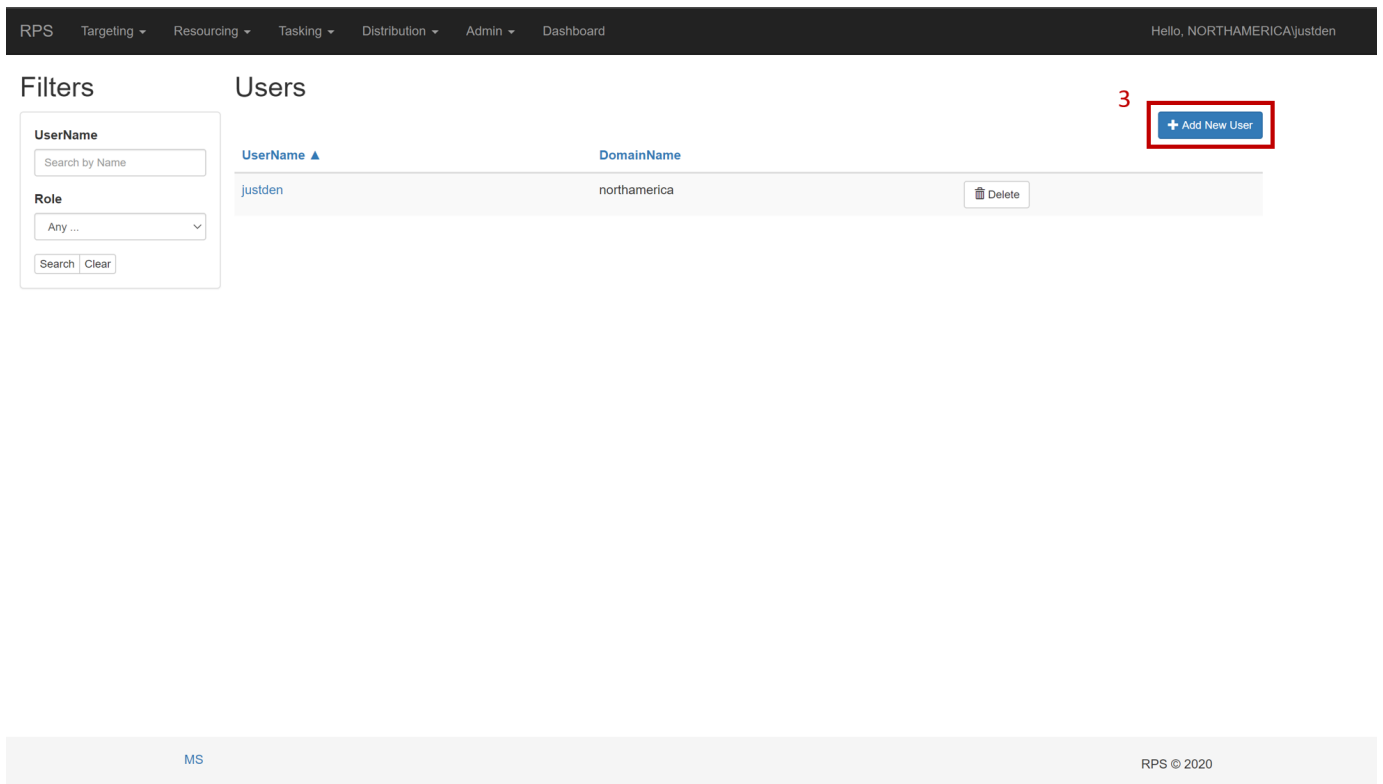
How to Add a Local/Domain User to RPS

To Add a User:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Users**.



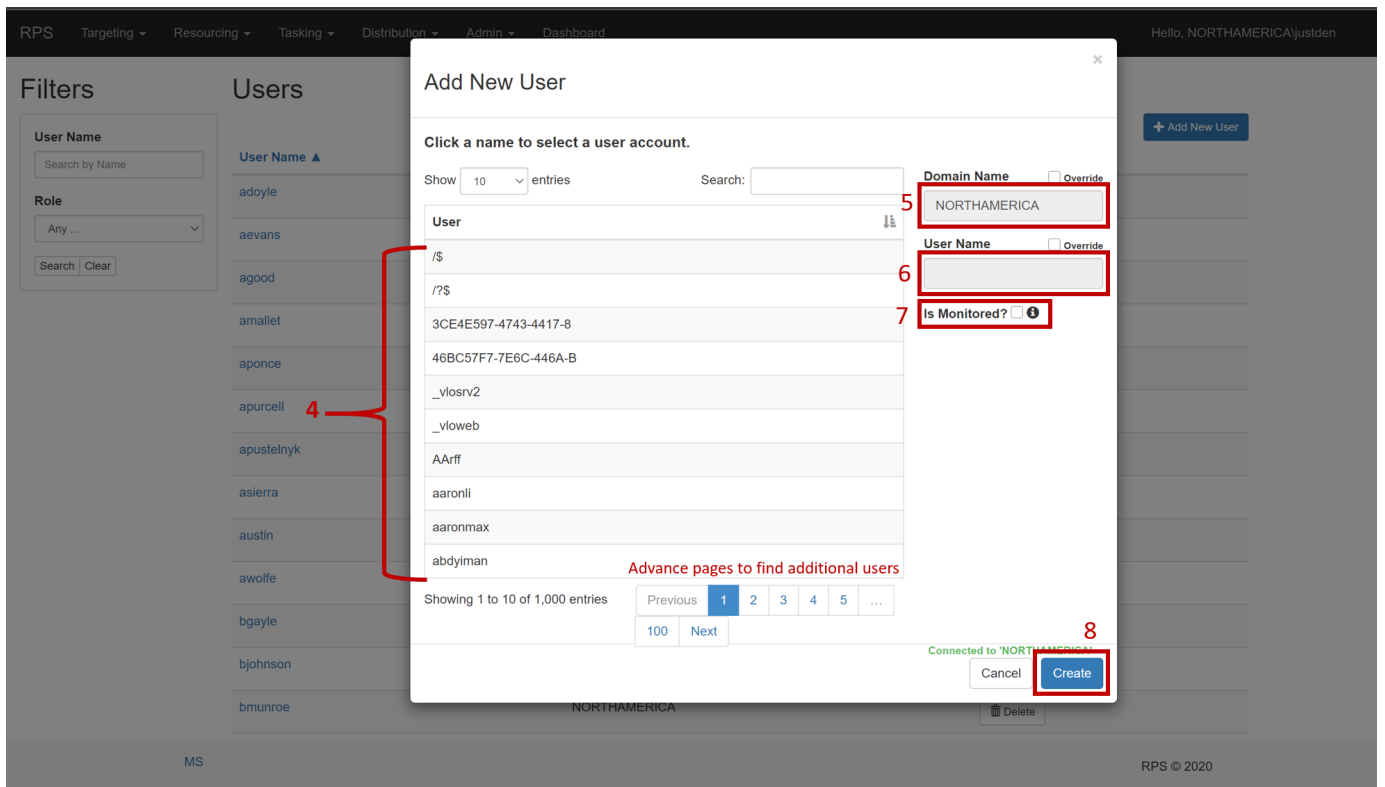
3. Click the **Add User** button.



4. Select a **User** from the Available Users List.
5. **Domain Name** should be pre-populated based on the User Selected.
6. Create a **User Name**.

NOTE
Can be any combination of letters and numbers.

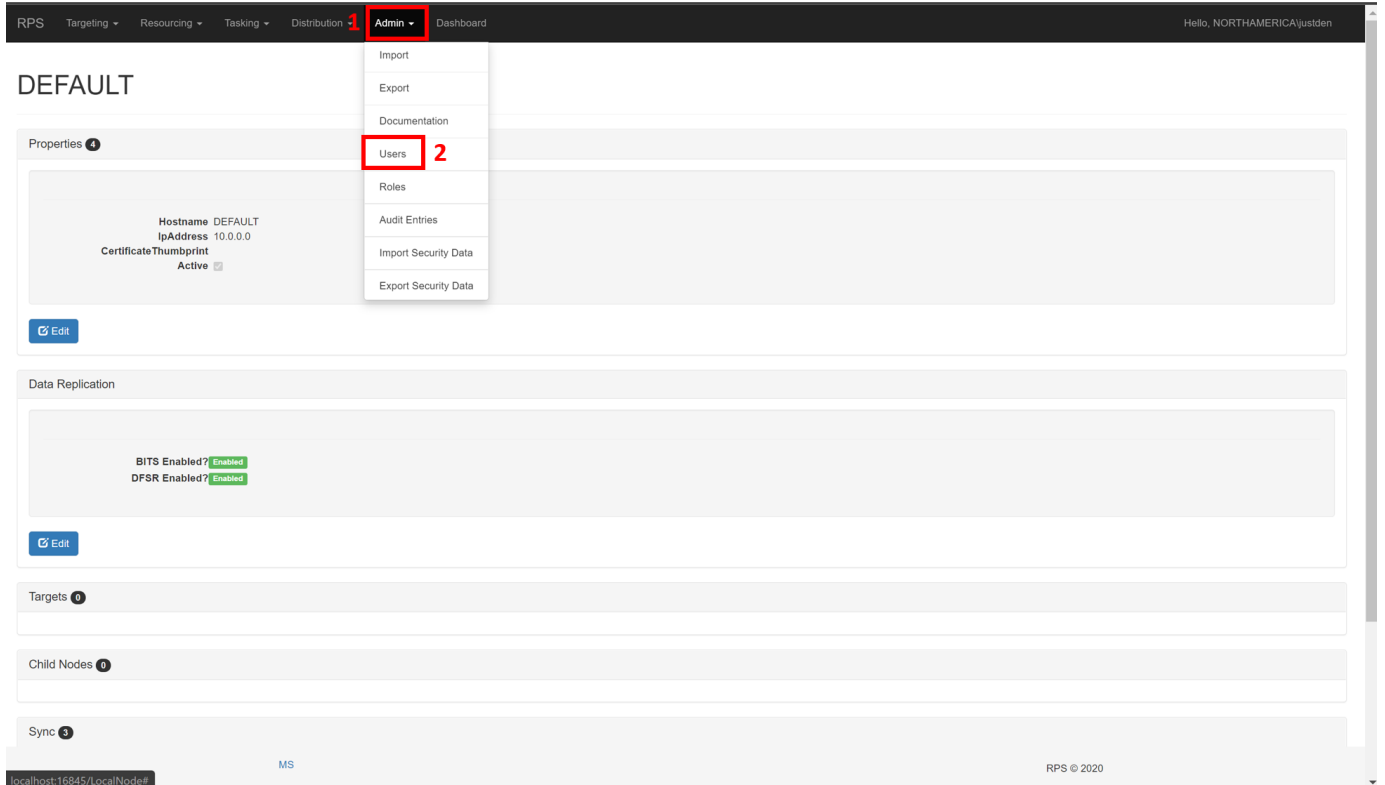
7. (**Optional**) Select "**Is Monitored?**" if the user is a "Break Glass" account.
8. Click **Create**.



How to Remove a Local/Domain User to RPS

To Remove a User:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Users**.



3. Click the **Delete** Trash Can button.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, NORTHAMERICA/justden

Filters

User Name
Search by Name

Role
Any ...

Search Clear

Users

+ Add New User

3

User Name ▲	Domain Name	Delete
adoyle	NORTHAMERICA	Delete
aevans	NORTHAMERICA	Delete
agood	FVEY	Delete
amallet	NORTHAMERICA	Delete
aponce	EUROPE	Delete
apurcell	NORTHAMERICA	Delete
apustelnyk	NORTHAMERICA	Delete
asierra	NORTHAMERICA	Delete
austin	NORTHAMERICA	Delete
awolfe	EUROPE	Delete
bgayle	NORTHAMERICA	Delete
bjohnson	NORTHAMERICA	Delete
bmunroe	NORTHAMERICA	Delete

MS RPS © 2020

4. Click **OK** on the confirmation pop up.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, NORTHAMERICA/justden

Filters

User Name
Search by Name

Role
Any ...

Search Clear

Users

+ Add New User

4

Confirmation

This action will delete the user. Are you sure?

Cancel OK

User Name ▲	Domain Name	Delete
adoyle	NORTHAMERICA	Delete
aevans	NORTHAMERICA	Delete
agood	FVEY	Delete
amallet	NORTHAMERICA	Delete
aponce	EUROPE	Delete
apurcell	NORTHAMERICA	Delete
apustelnyk	NORTHAMERICA	Delete
asierra	NORTHAMERICA	Delete
austin	NORTHAMERICA	Delete
awolfe	EUROPE	Delete
bgayle	NORTHAMERICA	Delete
bjohnson	NORTHAMERICA	Delete
bmunroe	NORTHAMERICA	Delete

MS RPS © 2020

Audit Entries

Last updated on February 4, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The purpose of an Audit Entry is to offer a means to examine a user action and change made to an entity, within RPS.

An Audit Entry represents a change in state of an RPS entity such as a Patch (i.e., a Resource Item). Audit Entries contain information such as both the original and new values of the altered entity. Additionally, a timestamp of the change is provided, as well as the Username of the person responsible for the change. You can also find information such as the effected entity class, type, ID, etc. within an Audit Entry.

This article will provide the How-To process for viewing Audit entries of users from RPS.

Assumptions

1. You have read the [Introduction to RBAC](#) article and have a basic understanding of Role-Based Access Control.
2. You have access to the RPS Graphic User Interface (GUI or UI).
3. You are assigned to the appropriate role to make the changes you intend to make.

RBAC Fundamentals

RBAC Terms and Definitions

- **Role-Based Access Control (RBAC)** – an authorization system that provides fine-grained access management of RPS resources.
- **Role** – a collection of permissions.
- **User** – A logical representation of a person or persona acting as a consumer (of the application). Most users are objects found in Active Directory, however some personas that are treated as users, such as service accounts are users that are not found in Active Directory.
- **Managed User** – An RPS User account whose changes to RPS entities are monitored and reviewable.

Audit Concepts

NOTE

It is important to note that RPS does **not** monitor users out of the box.

Audit Entries work in conjunction with Monitored Users. An Audit Entry is created when a Monitored User performs an action or makes a change within RPS. These records allow administrators to “audit” RPS whenever necessary.

There are two ways to view and interact with Audit Entries: through PowerShell, and through the RPS Web UI. Both features are discussed in detail below.

In the example of an Audit Entry created for a Patch, you might find the following information:

PARAMETER NAME	VALUE
Entity Id	bcc98a09-f4fb-4a65-94a8-6f0699bf0156
Entity Class	ResourceItem
Entity Change Type	Modified
Entity Type	Patch
Modified By	jdoe
Modified Data	9/29/2020 1:22:04 AM
Old Values	x
New Values	y

In this example, you can see that a ResourceItem of type Patch was modified by jdoe – where the old value of “x” was changed to “y” on 9/29/2020 at 1:22:04 AM.

Finding and Reviewing Audit Entries with the RPS Web UI

The Audit Entries web page can be viewed by all users; however, data will only be populated if the currently signed-in user possesses the required privileges/permissions. Users are only able to read/examine Audit Entries through the web user interface (UI). Users cannot create, modify, or delete Audit Entries.

To find audit entries in RPS:

1. From any page in RPS, select **Admin** in the navigation bar.
2. In the dropdown menu, select **Audit Entries**.

The screenshot shows the RPS Web UI interface. At the top, there is a navigation bar with several menu items: RPS, Targeting, Resourcing, Tasking, Distribution, Admin, and Dashboard. The 'Admin' menu item is highlighted with a red box and a red '1'. Below the navigation bar, the main content area is titled 'DEFAULT'. On the left side, there is a 'Properties' section with a list of properties: Hostname (DEFAULT), IpAddress (10.0.0.0), CertificateThumbprint, and Active (checked). Below this is an 'Edit' button. In the center, there is a 'Data Replication' section with two status indicators: 'BITS Enabled?' (Enabled) and 'DFSR Enabled?' (Enabled). Below this is another 'Edit' button. At the bottom, there is a 'Targets' section. The footer of the page shows 'localhost:16845/LocalNode#' on the left and 'RPS © 2020' on the right. The 'Audit Entries' option in the 'Admin' dropdown menu is highlighted with a red box and a red '2'.

Audit Entries are presented in a paginated/sorted table which looks as follows:

The screenshot shows the RPS web interface. At the top, there is a navigation bar with 'RPS' and several menu items: Targeting, Resourcing, Tasking, Distribution, Admin, and Dashboard. On the right, it says 'Hello, TestServer\admin'. Below the navigation bar, there are two main sections: 'Filters' and 'Audit Entries'. The 'Filters' section contains several input fields: 'Entity Id', 'Entity Class' (with a dropdown menu set to 'All'), 'Type', 'Entity Change Type' (with a dropdown menu set to 'All'), 'UserName', 'Start Date' (with a date picker), and 'End Date' (with a date picker). There is an 'Apply' button at the bottom of the filters. The 'Audit Entries' section shows a table with columns: 'Entity Id', 'Entity Class', 'Entity Type', 'Entity Change Type', 'UserName', 'Datetime (UTC)', and 'Details'. Above the table, it says 'Showing 0 to 0 of 0 entries'. There are 'Previous' and 'Next' buttons at the bottom right of the table. The footer of the page contains 'MS' on the left and 'RPS © 2020' on the right.

Filtering and Sorting Parameters

The following parameters can be used to return a filtered set of Audit Entries in both the RPS Web UI and PowerShell:

PARAMETER NAME	DESCRIPTION
EntityId	This parameter specifies the ID (GUID) of the audited entity.
EntityClass	This parameter specifies the class of the audited entity (i.e., "ResourceItem").
EntityType	This parameter specifies the type of the audited entity (i.e., "Patch").
ChangeType	This parameter specifies the state or change type of the audited entity. Valid values are: <ul style="list-style-type: none"> • Unchanged • Added • Modified • Deleted • Detached • Read
UserName	The username of the user responsible for the change associated with the Audit Entry.
StartTime	A start DateTime to filter the Audit Entries by.
EndTime	An end DateTime to filter the Audit Entries by.

RPS Targeting - Resourcing - Tasking - Distribution - Admin - Dashboard Hello, TestServerxadmin

Filters

Entity Id

Entity Class

Type

Entity Change Type

UserName

Start Date

End Date

Audit Entries

Show 10 entries

Entity Id	Entity Class	Entity Type	Entity Change Type	UserName	Datetime (UTC)	Details
No data available in table						

Showing 0 to 0 of 0 entries

Filter Options

MS RPS © 2020

To Sort a Table

Click on the column header you would like to sort by.

To Filter a Table

The table can be filtered with the provided filters on the left-hand side of the page. These filters are the same as described above in this section.

View Audit Entry Details

To view the details of a particular Audit Entry:

1. Click on the audit entries **View Details** button in the row of the entry.

Audit Entries

Show 10 entries

Entity Id	Entity Class	Entity Type	Entity Change Type	UserName	Datetime (UTC)	Details
bc098a09-46b-4a65-94a8-8f0699b0d156	ResourceItem	Patch	Modified	joe	09/29/2020 01:22:04	<input type="button" value="View Details"/>

Showing 1 to 1 of 1 entries

NOTE

By clicking on the button labeled above, you will be taken to a detailed view page for the Audit Entry, with more information such as the original and new values.

AuditEntry

```

Entity Id : bcc96a09-f4fb-4a65-94a8-6f0699bf0156
Entity Class : ResourceItem
Entity Change Type : Modified
Entity Type : Patch
Modified By : jdoe
Modified Data : 9/29/2020 1:22:04 AM
Old Values : Original Values
New Values : Current Values

```

[Back](#)

Finding and Reviewing Audit Entries Using PowerShell

RPS offers a single PowerShell cmdlet, **Get-RpsAuditEntry**, for interacting with Audit Entries. Audit Entries are essentially read-only to users, hence the sole cmdlet being a "GET".

This cmdlet allows authorized users to query the database for either All Audit Entries or a specified Subset. There are no required/mandatory parameters.

If the cmdlet is executed without passing in any parameters, such as by typing:

```
Get-RpsAuditEntry
```

A list of all Audit Entries will be returned to the user.

To Filter or Sort Audit Entries:

The filtering and sorting parameters for PowerShell are the same as the Web User Interface:

PARAMETER NAME	DESCRIPTION
EntityId	This parameter specifies the ID (GUID) of the audited entity.
EntityClass	This parameter specifies the class of the audited entity (i.e., "ResourceItem").
EntityType	This parameter specifies the type of the audited entity (i.e., "Patch").
ChangeType	This parameter specifies the state or change type of the audited entity. Valid values are: <ul style="list-style-type: none"> • Unchanged • Added • Modified • Deleted • Detached • Read
UserName	The username of the user responsible for the change associated with the Audit Entry.
StartTime	A start DateTime to filter the Audit Entries by.
EndTime	An end DateTime to filter the Audit Entries by.

Any of these filtering and sorting parameters can be used in combination with the **Get-RpsAuditEntry** cmdlet in the following format:

```
Get-RpsAuditEntry -Username "jdoe" -ChangeType "Deleted"
```


How to Add a Node to an Existing RPS Environment

Last updated on March 1, 2021.

Last Reviewed and Approved on PENDING REVIEW

This guide will describe how to add a new unparented node to an existing RPS Environment. For the following steps, the example environment is as follows:

- APP.Master
 - APP.Region
 - APP-S.Region
 - APP-S2.Region 1. This node is added after the initial deployment and does not initially have a parent

1. Login to APP.Region and App-S2.Region as RPSAdmin
2. Run this command on App.Region to see the details of the App.Region node. These details will be used in the next step.

```
Get-RpsNode
```

3. Run this command on Site2 to create the Region node:

```
New-RpsNode -NodeId <id_of_app.region> -Name Region -SyncEndpointUrl <sync_endpoint_url_of_app.region> -CertificateThumbprint <certificate_thumbprint_of_app.region> -Hostname <hostname_of_app.region> -IpAddress <ip_address_of_app.region>
```

4. Run these commands on Site2. The last command will show the details of the APP-S2 node. These details will be used in the next step.

```
$node = Get-RpsNode -Name Site2
$node.ParentNodeId = <id_of_app.region>
Set-RpsNode -Node $node
$node
```

5. Run the following command and restart RpsSync services to create the Site2 node:

```
New-RpsNode -NodeId <id_of_site2> -Name Site2 -SyncEndpointUrl <sync_endpoint_url_of_site2> -CertificateThumbprint <certificate_thumbprint_of_site2> -Hostname APP-S2.region.rps -IpAddress <ip_address_of_site2> -ParentNodeId <id_of_app.region>
```

1. Restart the RpsSync service on both machines (APP.Region and APP-S2.region):
 2. Windows Key + R
 3. Services.msc
 4. Find RpsSync, right-click on it, and select Restart
6. On APP.region, after the sync service has restarted
 1. Windows Key + R
 2. iexplore <https://app.region.rps:8080/>
 3. Targeting > Nodes
 4. Find the Region node and click on it
 5. Make sure that the Site2 node is listed as a Child Node
 6. Dashboard > Sync
 7. Check for any sync errors to make sure it's synchronizing properly
7. On APP-S2.region, after the sync service has restarted
 1. Windows Key + R

2. iexplore <https://app-s2.region.rps:8080/>
 3. Targeting > Nodes
 4. Find the Region node and click on it
 5. Make sure that the Site2 node is listed as a Child Node
 6. Dashboard > Sync
 7. Check for any sync errors to make sure it's synchronizing properly
 8. Resourcing > Resources
 9. Add Resource > Resource
8. Fill out the form with any information and click on the Save button
9. Restart the RpsSync service on both machines (APP.Region and APP-S2.region)
10. Check that your new Resource appears on both machines (you may need to give sync a few minutes). Do the following on each machine to check:
1. Windows Key + R
 2. iexplore <https://app.region.rps:8080/>
 3. Resourcing > Resources
 4. Look for your new Resource

How to Self-Register Your Node: Node Self-Parenting

Last updated on January 19, 2021.

Last Reviewed and Approved on PENDING REVIEW

A node can self register with a parent through the **ParentSyncEndpointUrl** property. When this property is set, the Sync service sends a request with its node (and any of its child nodes) to the Sync service endpoint specified by the property. The service at **ParentSyncEndpointUrl** then adds the sending node (and any of that node's child nodes) to its database and set the the sending node as a child. The new parent will then send its node back to the child. The new child then adds the parent node to its database.

NOTE

The parent sends only its node information to the child. It does not tell the child about any other nodes. In this sense, the child will never know about grandparent(s) nor sibling(s).

If the child node already exist on the parent node, then no modifications are made at the parent node. For example, if the parent has information for the child node, but there is no parent-child relationship, a parent-child relationship will not be established. The child node information must first be removed from the parent's database.

If the child node already has a parent node, it will not try to reregister with that parent nor register with a new parent. To reestablish or establish a new parent relationship, the parent node information must first be removed from the child's database.

The Sync service will retry the parenting request at intervals, configurable by setting the **SelfRegisterNodeIntervalInMin** property to the number of minutes between attempts. If not set, the default is 3 minutes.

Prerequisites

- Rps.Sync.exe must be version 3.1.3 or greater.
- The Sync service on the child must be able to communicate and authenticate with the Sync service on the parent.

To Register a Node with a Parent:

1. On the child node, configure the **ParentSyncEndpointUrl** property with the value of the desired parent node's **SyncEndpointUrl**.
2. [Optional] On the child node, configure the **SelfRegisterNodeIntervalInMin** property with the number of minutes to wait between registration attempts. If this property is not set, the default is 3 minutes.
3. On the child node, restart the Sync service. The **ParentSyncEndpointUrl** and **SelfRegisterNodeIntervalInMin** properties are only retrieved from the node when the service starts.

How to Configure RPS Sync Settings

Last updated on August 31, 2021.

Document Status: Document Developer Quality Complete.

Introduction

This article will describe the settings that are used to configure RPS Sync.

SyncOrchestratorTimerSetting

SyncOrchestratorTimerSetting is used to configure the schedule of sync orchestration. When sync orchestration occurs, changes are pushed to and/or pulled from the parent node. Sync orchestration does not include committing changes to the database, only the pulling and pushing of changes.

The value of the setting needs to be a CRON expression. For examples of CRON expressions, see <https://crontab.guru/examples.html>.

NOTE

If SyncOrchestratorTimerSetting is not set, by default sync orchestration will occur every 1 minute.

The following example would configure sync orchestration to occur every 10 minutes.

```
Set-RpsStorageValue -Key 'SyncOrchestratorTimerSetting' -Value '*/10 * * * *'
```

SyncMergeTimerSetting

SyncMergeTimerSetting is used to configure the schedule of merging in received RPS changes. When merge occurs, RPS changes are committed to the database.

The value of the setting needs to be a CRON expression. For examples of CRON expressions, see <https://crontab.guru/examples.html>.

NOTE

If SyncMergeTimerSetting is not set, by default sync merge will occur every 3 minutes.

The following example would configure sync merge timing to occur every 15 minutes.

```
Set-RpsStorageValue -Key 'SyncMergeTimerSetting' -Value '*/15 * * * *'
```

SyncEndpointUrl

Sync is initiated by child nodes and it will use its parent node's `SyncEndpointUrl` value as the endpoint to push to and pull from.

The following example will set the local node's parent Sync Endpoint Url to `https://NOSC.rps.local:777/sync/v1.0/sync`. The value used should be specific to the URL for the local node's parent sync endpoint URL.

```
# Get Local Node's parent
$parentNode = (Get-RpsLocalNode).ParentNode

# Set the Parent Node's Sync Endpoint Url
Set-RpsNode -Node $parentNode -SyncEndpointUrl 'https://NOSC.rps.local:777/sync/v1.0/sync'
```

Introduction to Logging in RPS

Last updated on August 25, 2021.

Document Status: *Document Developer Quality Complete.*

What is Logging in RPS

Logging in RPS is the act of storing relevant information around processes and functions that RPS performs. These logs can be informational, information useful for diagnosing application state, or even errors that occur during processing.

When Does Logging Occur

Logging in RPS occurs when any process, both background and foreground, hits a point where information relevant to the user should be conveyed.

How to Control Logging Behavior in RPS

Last updated on August 25, 2021.

Document Status: Document Developer Quality Pending.

RPS Logging Settings

RPS' logging behavior can be controlled by changing settings in the RPS Settings.

MinimumLogLevel

The `MinimumLogLevel` setting controls what level of logging is written to the event viewer. This setting is global and will apply to **all** services and components of RPS. The setting defaults to *Warning* if a valid value is not set.

```
Set-RpsStorageValue -Key MinimumLogLevel -Value <Value>
```

Acceptable `MinimumLogLevel` setting values:

- Debug
- Error
- Fatal
- Information
- Verbose
- Warning

Use Case

Example: The user would like to see debug level logging to diagnose an issue. The current `MinimumLogLevel` is set to *Warning*. The user should change the log level to *Debug* log level in order to see more specific and detailed logs.

```
Set-RpsStorageValue -Key MinimumLogLevel -Value "Debug"
```

After they have finished troubleshooting, the user can set the log level back to *Warning* to avoid filling the log with informational messages that are not routinely needed.

```
Set-RpsStorageValue -Key MinimumLogLevel -Value "Warning"
```

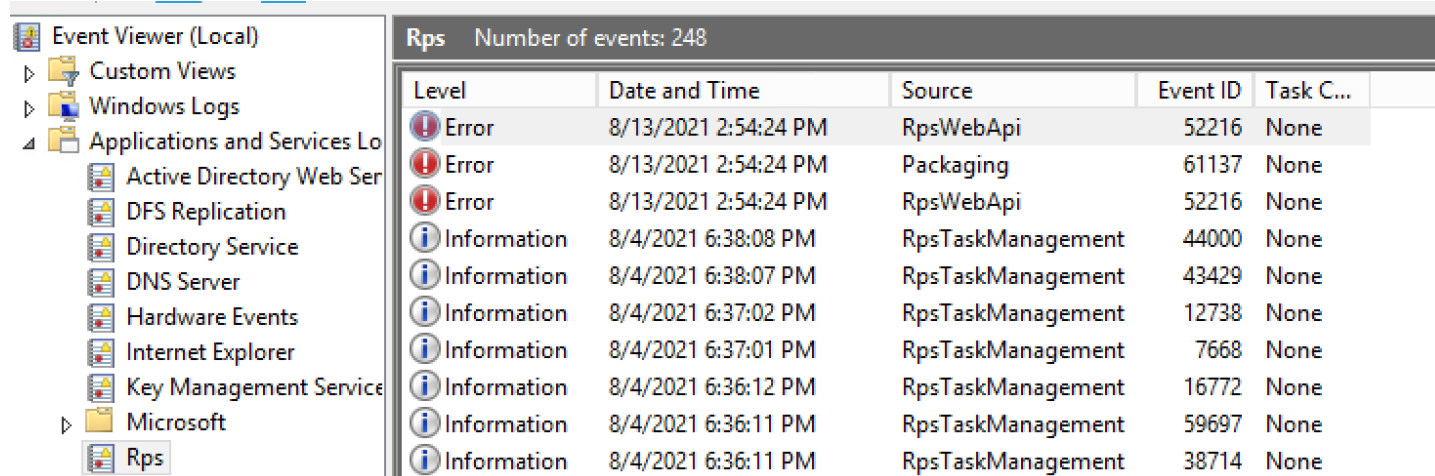
Viewing RPS Logs

Last updated on August 25, 2021.

Document Status: Document Developer Quality Pending.

Where Are RPS Logs Stored

RPS logs are stored in the Windows Event Log under "Application and Services Logs" → RPS. The individual logs are grouped by the component that logged them, such as RpsApi or RpsWebApi.

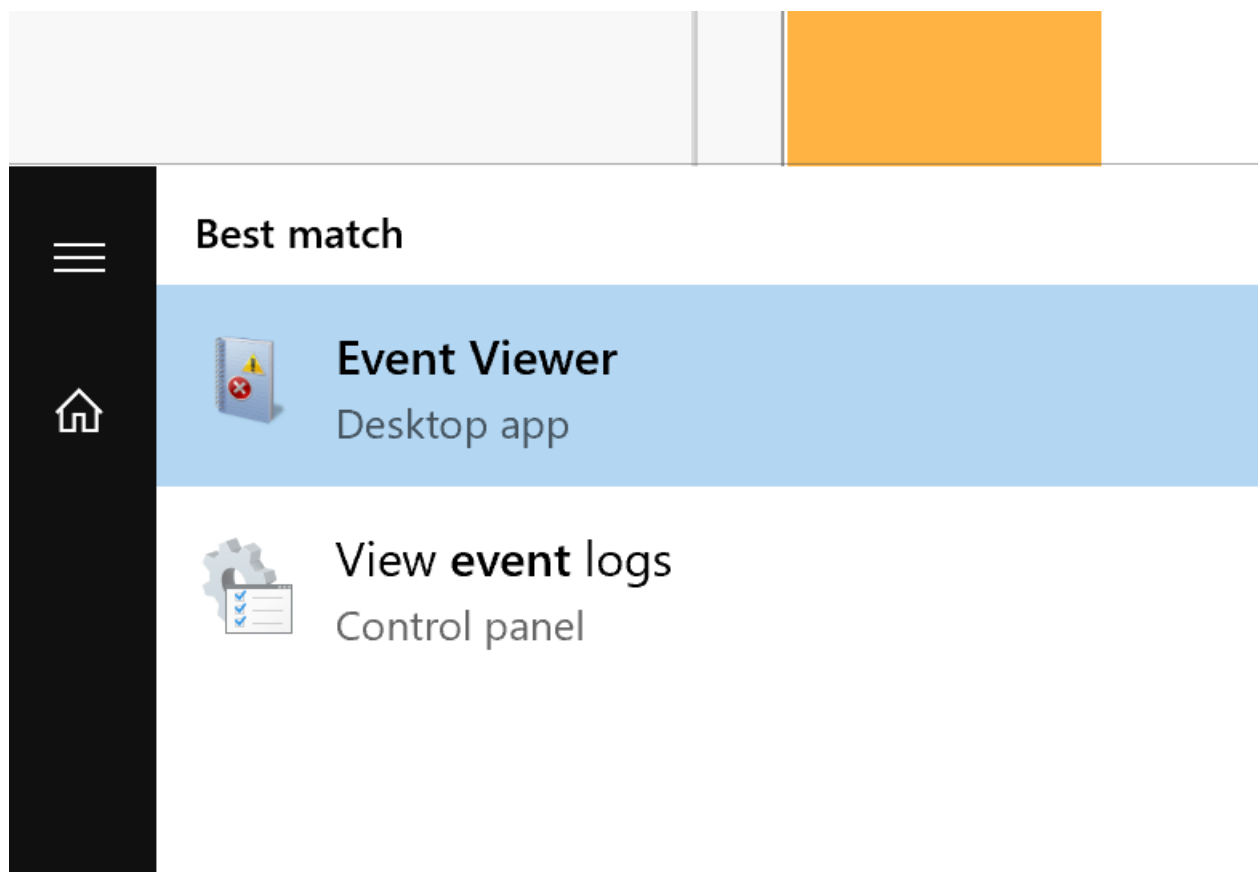


Level	Date and Time	Source	Event ID	Task C...
Error	8/13/2021 2:54:24 PM	RpsWebApi	52216	None
Error	8/13/2021 2:54:24 PM	Packaging	61137	None
Error	8/13/2021 2:54:24 PM	RpsWebApi	52216	None
Information	8/4/2021 6:38:08 PM	RpsTaskManagement	44000	None
Information	8/4/2021 6:38:07 PM	RpsTaskManagement	43429	None
Information	8/4/2021 6:37:02 PM	RpsTaskManagement	12738	None
Information	8/4/2021 6:37:01 PM	RpsTaskManagement	7668	None
Information	8/4/2021 6:36:12 PM	RpsTaskManagement	16772	None
Information	8/4/2021 6:36:11 PM	RpsTaskManagement	59697	None
Information	8/4/2021 6:36:11 PM	RpsTaskManagement	38714	None

Figure 1: Window Event Viewer showing RPS Logs.

How to Open Windows Event Log Viewer

1. Open the start menu and type "Event Viewer".
2. Select Event Viewer at the top of the menu list.



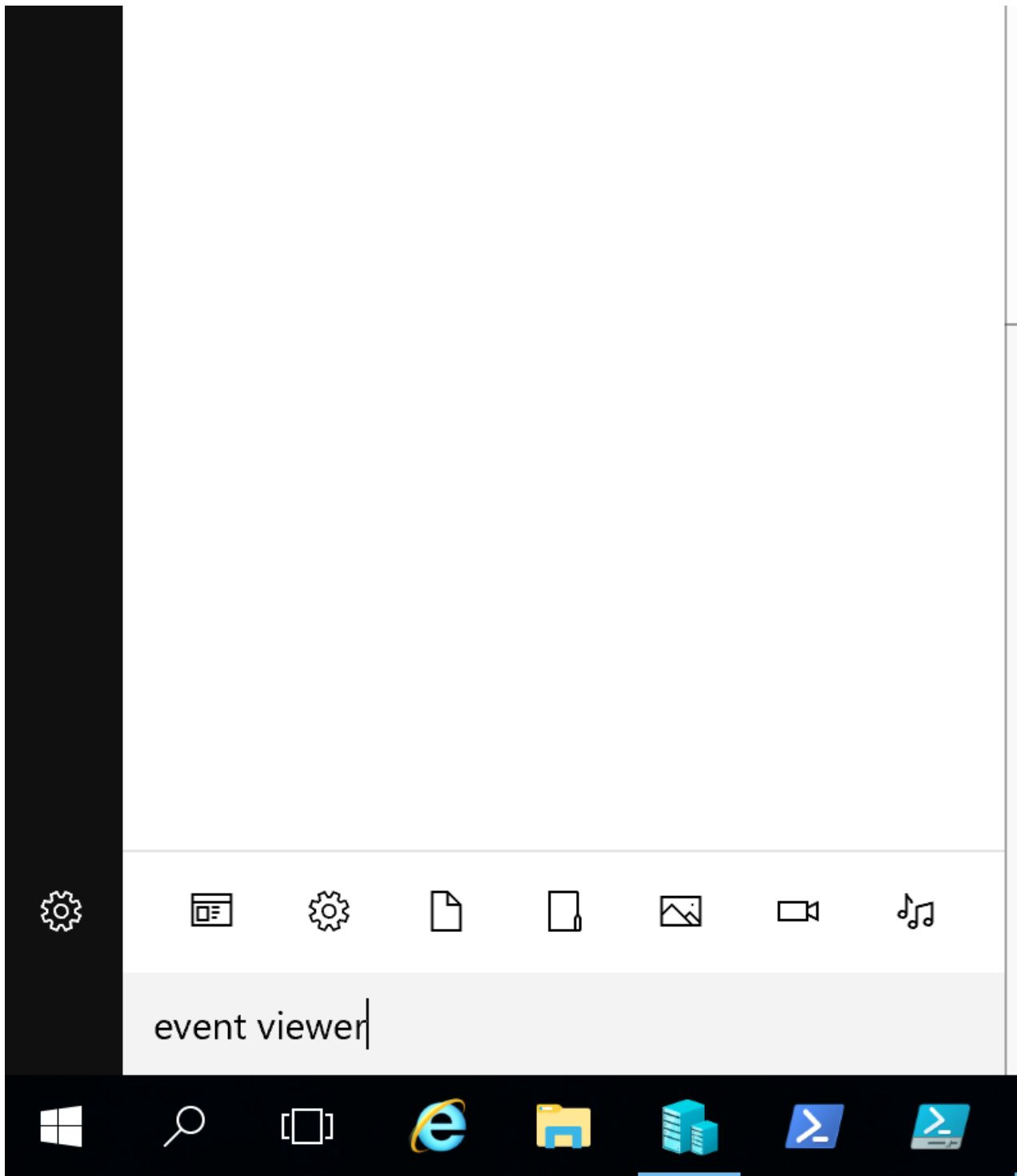


Figure 2: Start menu showing Event Viewer search.

3. Expand "Application and Services Logs" and select RPS.

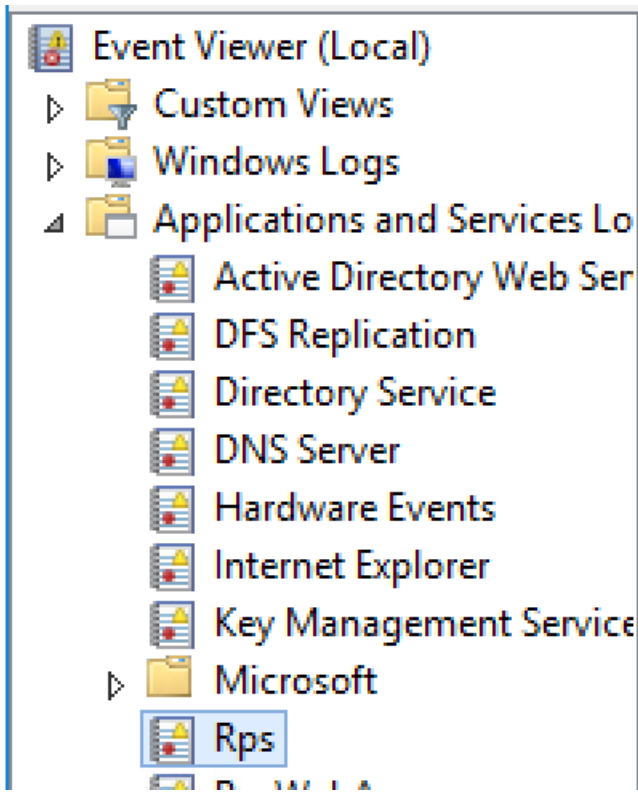


Figure 3: Selecting RPS logs in Event Viewer.

Writing Log Messages

Last updated on August 25, 2021.

Document Status: Document Developer Quality Pending.

Write-RpsLogItem PowerShell Cmdlet

Logs can be written to the RPS log via the `Write-RpsLogItem` Cmdlet in PowerShell.

⚠ IMPORTANT

The first time writing a log with a component that has not been written to the log before, the session needs to be run as a user with Administrator credentials in order to create the component as a new Log Source.

Parameters

PARAMETER NAME	TYPE	REQUIRED	DESCRIPTION
Level	String	True	The logging level of the log. Acceptable values are: <i>Debug, Error, Fatal, Information, Verbose, Warning.</i>
Component	String	False	The name of the component that is responsible for the statement. Defaults to A.
MessageTemplate	String	True	The message template to log. Parameterized values should be closed in "{}". I.e., "This is a log message for {parameter}".
Properties	String	False	The properties to attach to the log message.

Examples

Without Properties

This will write a message without parameterized values to the backing RPS log.

```
Write-RpsLogItem -Level "Warning" -Component "Master-Controller" -MessageTemplate "Task map is already assigned." -Force
```

The output would be similar to this:

```
PS C:\Windows\system32> Write-RpsLogItem -Level "Warning" -Component "Master-Controller" -MessageTemplate "Task map is already assigned."  
WARNING: Task map is already assigned.
```

Figure 1: Writing RPS log message without parameters.

With Properties

Properties will be used to fill in parameterized parts of the MessageTemplate. For example, "User {0} has performed an {1} action" is the template. The parameters could be `@('SomeRpsUser', 'Create')`. The resulting message would be "User SomeRpsUser has performed an Create action".

```
Write-RpsLogItem -Level "Debug" -Component "RpsWebApi" -MessageTemplate "User {0} has performed an {1} action" -Properties @('SomeRpsUser', 'Create')
```

The output would be similar to this:

```
PS C:\Windows\system32> Write-RpsLogItem -Level "Warning" -Component "RpsWebApi" -MessageTemplate "User {0} has performed an {1} action" -Properties @('SomeRpsUser', 'Create')  
WARNING: User SomeRpsUser has performed an Create action
```

Figure 2: Writing RPS log message with parameters.

RPS Testing Strategy

Last updated on May 12, 2020.

Last Reviewed and Approved on PENDING REVIEW

Overview

Automation is the primary form of testing for the RPS development team. An ever growing suite of unit, integration, systems and UI tests are run on a regular basis to ensure functionality aligns with assumptions and expectations. Tests are driven by the requirements of user stories and verifying that remediated defects do not return. There is no manual test plan maintained by the RPS team. Instead, we rely on our collection of automated tests to serve as a living test plan. Results of automated tests, and the builds they are associated with, can be found in Azure DevOps.

Automated Tests

All automated tests are stored as code within the Core and Common RPS repositories. In order to review all automated tests, an evaluator would need access to the Core and Common RPS repositories and be able to read C# and PowerShell code. If you do not have access to the RPS repositories, and you would like access for reviewing the suite of tests that RPS runs, please contact an RPS Project Manager. The results of running automated tests are available in Azure DevOps Dashboards. If you require access to the dashboards to review the results of automated tests, please contact an RPS Project Manager.

Current Test Locations

At the time of writing, system tests can be found in the SystemTest folder within the RPS Core repository. Integration tests can be found within the Core repository in Source/{FolderName}.IntegrationTests folders. Unit tests can be found in the Core repository in Source/{FolderName}.Tests folders. As repository rationalization continues, these test locations may change over time. Please contact an RPS Project Manager if test files and folders do not align with this documentation.

RPS Automation Package Guidelines

Last updated on January 13, 2021.

Last Reviewed and Approved on PENDING REVIEW

The Rapid Provisioning System (RPS) is designed to be a re-usable solution that offers similar, flexible functionality for programs, efforts, and automation requirements. The system was initially designed to survive the inherent constraints and challenges of a mobile tactical network.

The goal of this design is to create a stable, secure, and highly automated management infrastructure solution to provide the following (but not limited to) capabilities:

How to Load an Automation Package

An Automation Package is typically loaded using a Load Script. The load script is flexible but is coded to simply place the components for a given automation package in the proper locations. This script is typically only executed one time to initially load a set of functionalities, scripts, metadata, or other configuration information into the RPS system. This script is run once during the initial load of the Automation Package. This can occur at initial system build, or after the system is already established, and should contain all logic necessary to add itself to the RPS system.

Some common actions a Load Script may take will include:

- Inserting Metadata into the RPS Configuration Management Database (CMDB)
- Placing PowerShell Scripts and Modules into the appropriate locations defined for RPS, this will allow the system to automatically register and load them for use (See Section 2.5 for info on placement)
- Placing external software components used by those Scripts and Modules into the necessary locations as coded by the authors (e.g., VMWare tools being installed)
- Inserting TaskItem and TaskMap metadata into the database. These are the definitions of what should be run, targeting what devices, and in what order.

Once the load script is executed, it is typically removed and no longer required. The initial insert of data serves as a way to inject new business logic into the system. From there, the system will function based on the inserted POR authored Automation Package.

Post-Load Actions and Activity

Once a package is loaded, business requirements begin to dictate what will happen next. For example, if the Load Script is coded to activate automation work on load, then RPS actions will begin immediately processing.

If the Load Script is not coded to activate automation work on load, the following RPS components can be used to control and trigger automation execution:

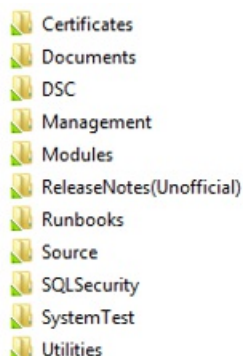
- IsActive settings
- PendingUserActions
- TaskAssignment creation
- TaskMapAssignment creation

Using these core functionalities, automation can be triggered, controlled, and centrally managed on an as-needed basis. Post-Load activities are controlled by the needs of the business and should be coded to support the intended execution model.

Format of an Automation Package

The format of an automation package should follow the folder structure of the build output that RPS produces, and either be included with the initial set of installations of an RPS software stack, or placed into the same folders once RPS is already installed.

For example, using the figure below showing a sample build output from RPS, there are clearly defined locations for various components of an Automation Package. There are folders and descriptions for the intended location of specific components already defined. The format of the Automation package will vary depending on business use and intended uptake of the content.



Build Output and Automated Support

Some folders supplied as part of the RPS Build output will support automated control and placement of inserted code product. This section will explain which folders are supported and by what action or activity:

TERM	DEFINITION
Certificates	This is a general store for storage of certificates required by the system. Some are controlled automatically, but this is done per certificate currently.
DSC	This folder is used to hold both Desired State Configuration (DSC) configurations and Resources. RPS support of DSC processes are coded to look to these locations for supporting DSC resources and configuration files.
Modules	Any PowerShell modules located in this directory will be automatically placed into the appropriate PowerShell Modules directory on the server RPS is installed on. Once this directory exists on a server with RPS installed, any newly added modules will be copied on next pass of the DSC configuration (typically 30 minutes).
Runbooks	Default folder for Task Management Service to locate and execute Runbooks. Only runbooks located here will be executed by TMS. Subfolders are not supported.

More Resources

- [RPS Customization Guide - High Level Overview](#)
- [RPS IPSheet Parser](#)

Token Based Software Activation with PowerShell

Last updated on April 14, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the Lead Service Integrator (LSI) or developer.

Introduction

During a new vehicle field activation, you will likely need to activate a Windows Operating System and/or Microsoft Office products. There are three primary methods that can be used: PowerShell, RPS runbooks, and through the RPS Web User Interface.

Assumptions

The following is assumed prior to performing the actions described in this article:

- You have access to the certificate (token), the public certificate, and issuance license file mentioned in the requirements above.
- You have access to PowerShell and the `Rps-SoftwareActivation` module is installed.

Token Based Activation Requirements and Pre-Requisites

Token Based Activation Pre-Requisites

To activate Microsoft products via token based activation, three things are needed:

1. The **certificate (token)** that will be used to activate the software.
2. The **public certificate** from the Certificate Authority that signed the activation token.
3. The **issuance license file** that defines what software products will be activated.

Installing Pre-Requisites

Before activating any software, you must establish your working session. To do so, execute the following PowerShell Command in PowerShell ISE Administrator Mode.

IMPORTANT

Start by establishing your working session in PowerShell ISE Administrator Mode.

1. Click on the **Search Icon** from the Start Menu.

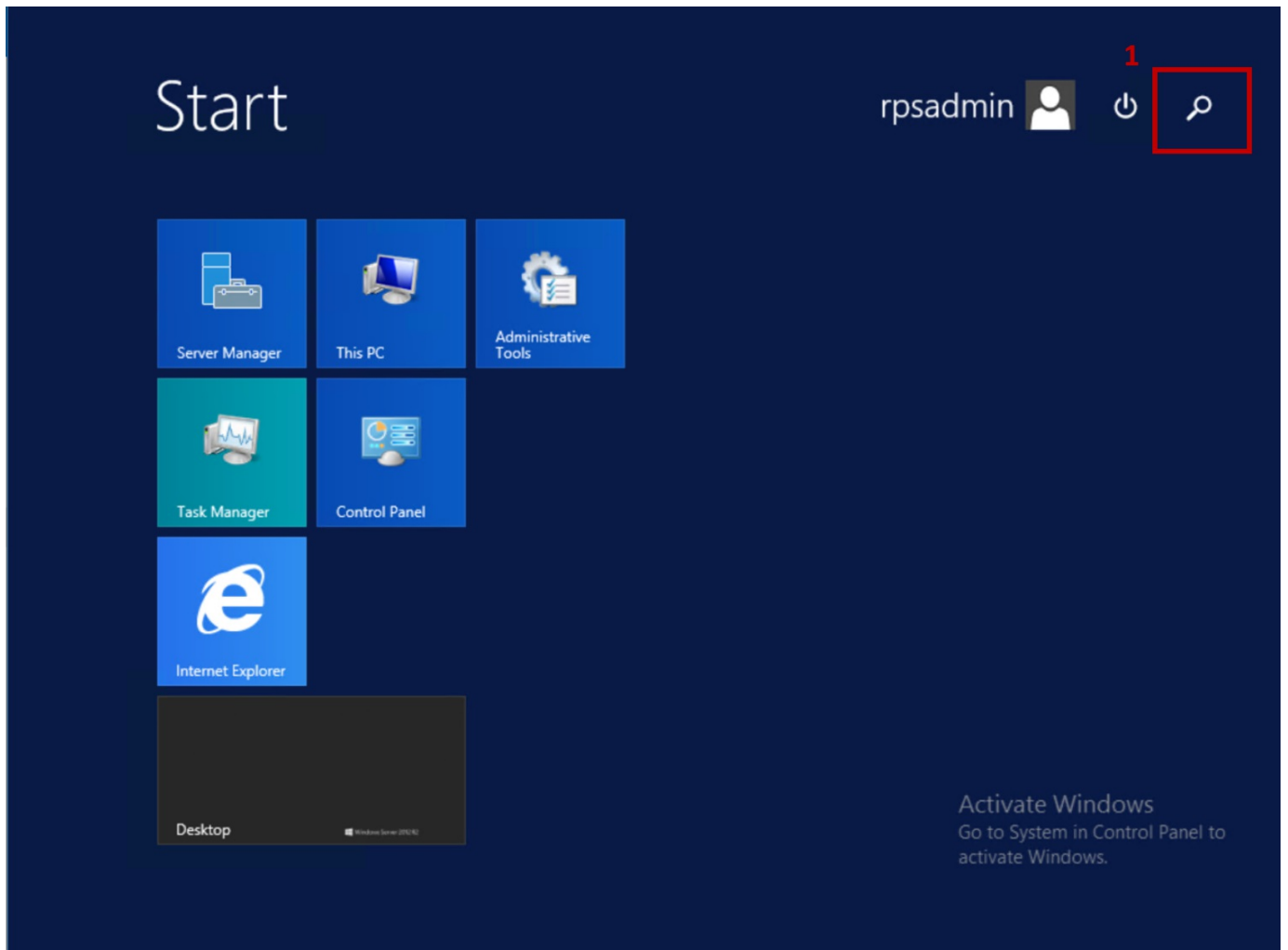


Figure 1: Click on Search Icon.

2. Search for PowerShell ISE by typing **PowerShell ISE** in the Search bar.

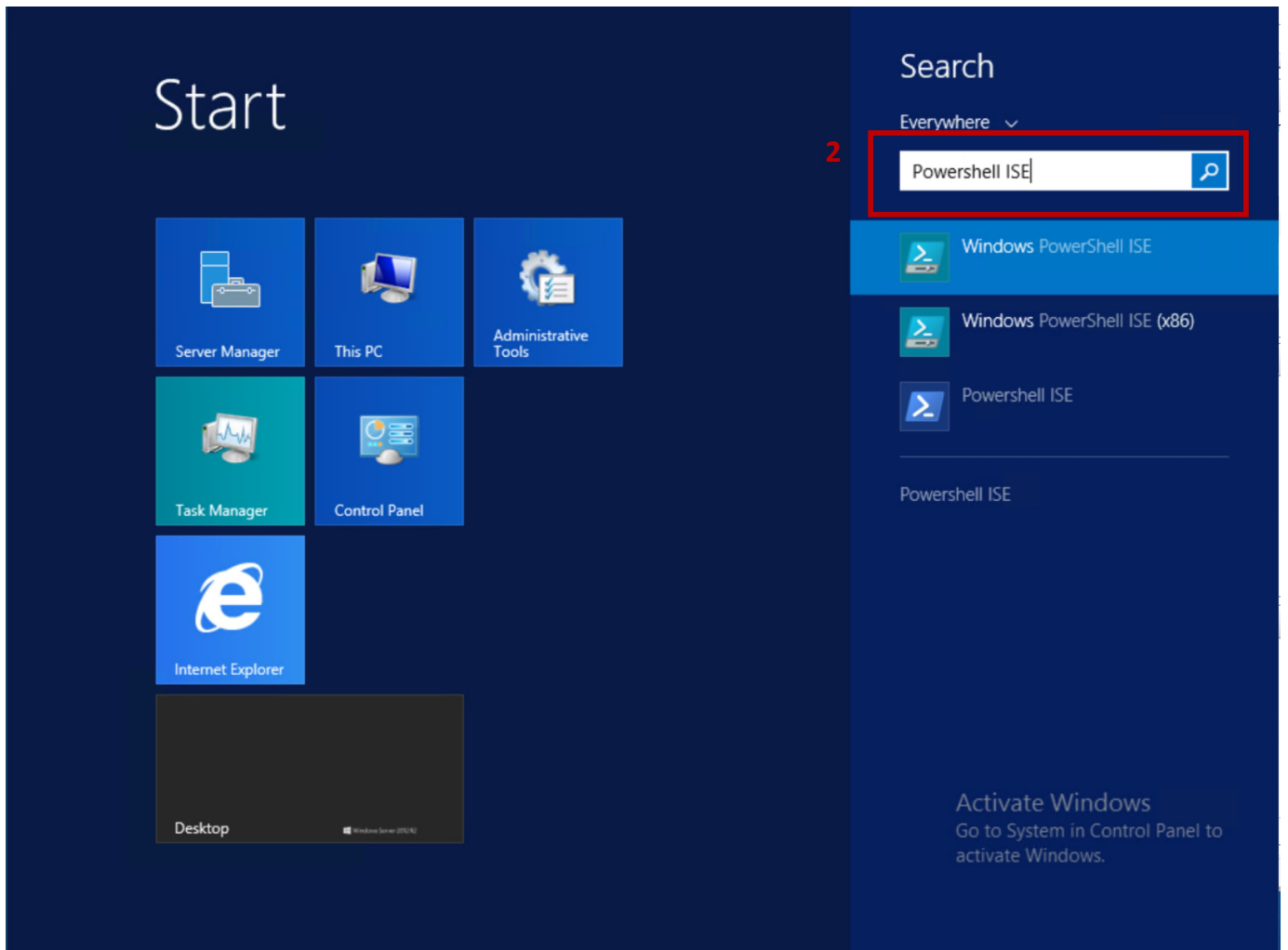


Figure 2: Search for PowerShell ISE.

3. Click on **Run As Administrator**.

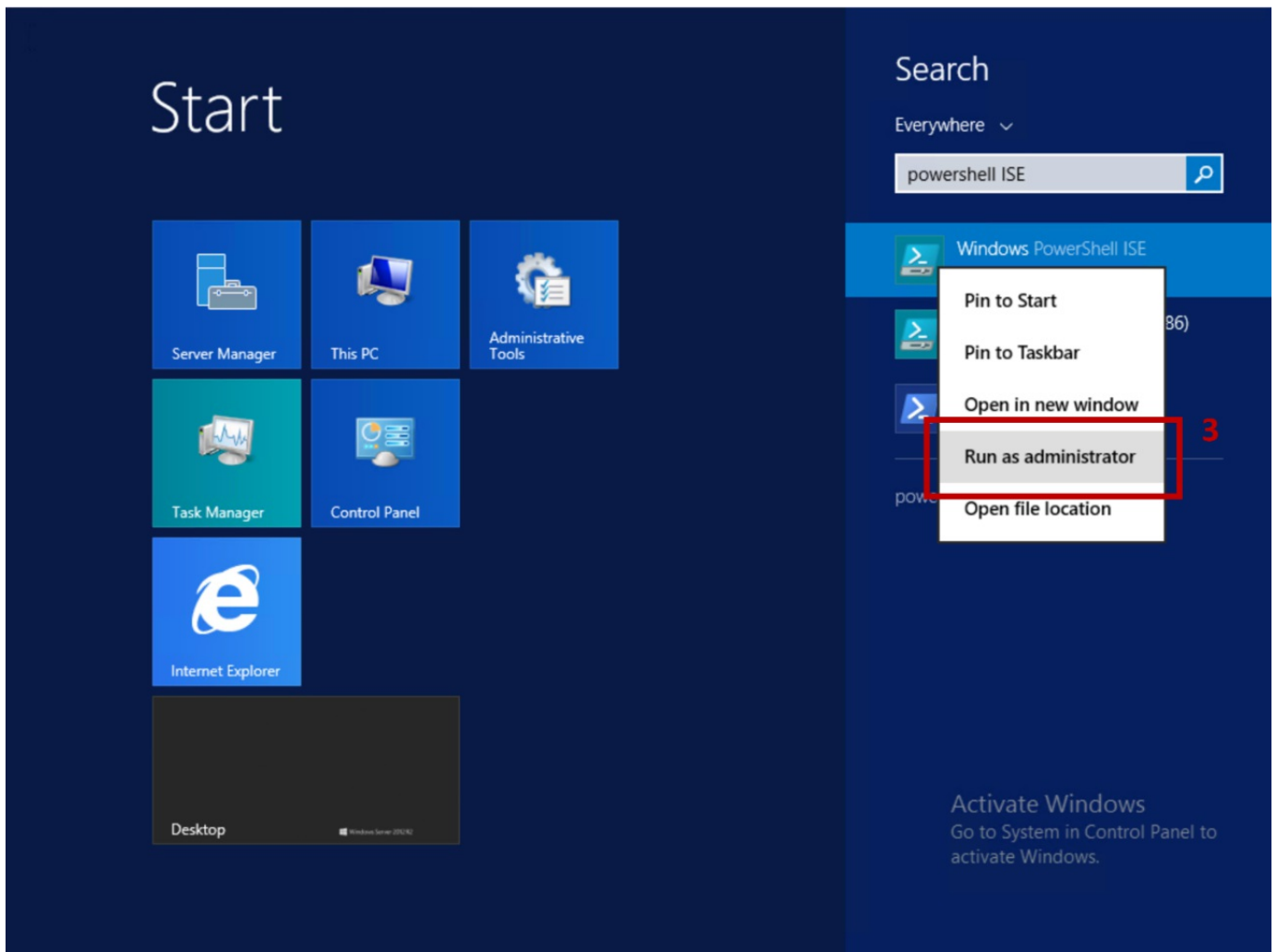


Figure 3: Open PowerShell ISE as Administrator.

NOTE

Any component of the PowerShell scripts below that are bookended by single quotes, must be replaced with use specific data (i.e., setting your own password).

4. Install the Required Certificates. The following cmdlet will install both the certificate (token) and public certificate:

```
$certificatePassword = ConvertTo-SecureString -String 'password!' -AsPlainText -Force

Install-TokenBasedActivationCertificate -ActivationPrivateCertificate 'tokenBasedActivation.pfx' -
Password $certificatePassword -ActivationPublicCertificate 'MicrosoftProductActivationPCA 2017.cer'
```

5. Install the Issuance License File. The following cmdlet will install the issuance license:

```
Install-SoftwareActivationIssuanceLicense -IssuanceLicensePath 'C:\Windows
Client_Server_Office_Visio_SHA2.Request52416.xrm-ms' -Verbose
```

After completing the pre-requisites, you can now activate the Windows OS and/or Office Product.

Activating a Windows OS

To activate a Windows OS, execute the PowerShell commands below. This PowerShell cmdlet searches through the local machine certificate store to find the certificate that has "Microsoft Product Activation" in the Subject. This will be the certificate used as the token for activation.

```
$cert = dir Cert:\LocalMachine\My | Where {$_.Subject -match 'Microsoft Product Activation'}
Register-TokenActivationIssuanceLicense -CertificateThumbprint $cert.Thumbprint -Verbose
```

Activating a Microsoft Office Product

To activate an Office product, execute the PowerShell commands below. This PowerShell cmdlet is used to activate Office and requires both the thumbprint of the certificate used for activation and the path to the issuance license file.

```
$cert = dir Cert:\LocalMachine\My | Where {$_.Subject -match 'Microsoft Product Activation'}
Register-OfficeProductActivation -IssuanceLicensePath 'license.xml' -CertificateThumbprint $cert.Thumbprint
```

More Resources

- [Token Based Software Activation with RPS Runbooks](#)

Token Based Software Activation with RPS Runbooks

Last updated on April 14, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the Lead Service Integrator (LSI) or developer.

Introduction

During a new vehicle field activation, you will likely need to activate a Windows Operating System and/or Microsoft Office Products. There are three primary methods that can be used: PowerShell, runbooks, and through the Web User Interface. This article will demonstrate how to activate software using runbooks.

Assumptions

The following is assumed prior to performing the actions described in this article:

- You have access to the certificate (token), the public certificate, and issuance license file mentioned in the requirements below.
- You have access to PowerShell and/or RPS Graphic User Interface (GUI or UI).
- You are assigned to the appropriate role to make the changes you intend to make.

Token Based Activation Requirements and Pre-Requisites

Runbook Pre-Requisites

To activate Microsoft products with runbooks via token based activation, three things are needed:

1. The **certificate (token)** that will be used to activate the software.
2. The **public certificate** from the Certificate Authority that signed the activation token.
3. The **issuance license file** that defines what software products will be activated.

To activate Windows software with runbooks in RPS, the following runbooks may be used:

- **Set-WindowsActivation**: used to activate Windows OS.
- **Set-OfficeActivation**: used to activate Office products.

Verifying Runbook Requirements

In order to activate Windows or Office with their respective runbook, you must first verify that the following exist:

1. **Certificates** (See [Token Based Software Activation with PowerShell](#)).
2. **Software License Resource Item** (See [Token Based Software Activation with PowerShell](#)).

Runbook TargetItem Pre-Requisites: Certificates & Roles

The runbook will be run against a TargetItem. This TargetItem must have two certificates assigned with the following roles:

- **WindowsActivation**
- **WindowsActivationCA**

RPS ResourceItem Pre-Requisites

RPS must have a ResourceItem with the following:

1. `Type` of "SoftwareLicense".
2. `Role` of "Windows".

Activating Software with RPS Runbooks

Establish a Working Session

Before activating any software, you must establish your working session. To do so, execute the following PowerShell Command in PowerShell ISE Administrator Mode.

IMPORTANT

Start by establishing your working session in PowerShell ISE Administrator Mode.

1. Search for PowerShell ISE by typing **PowerShell ISE** in the Cortana Search bar.
2. Click on **Run As Administrator**.

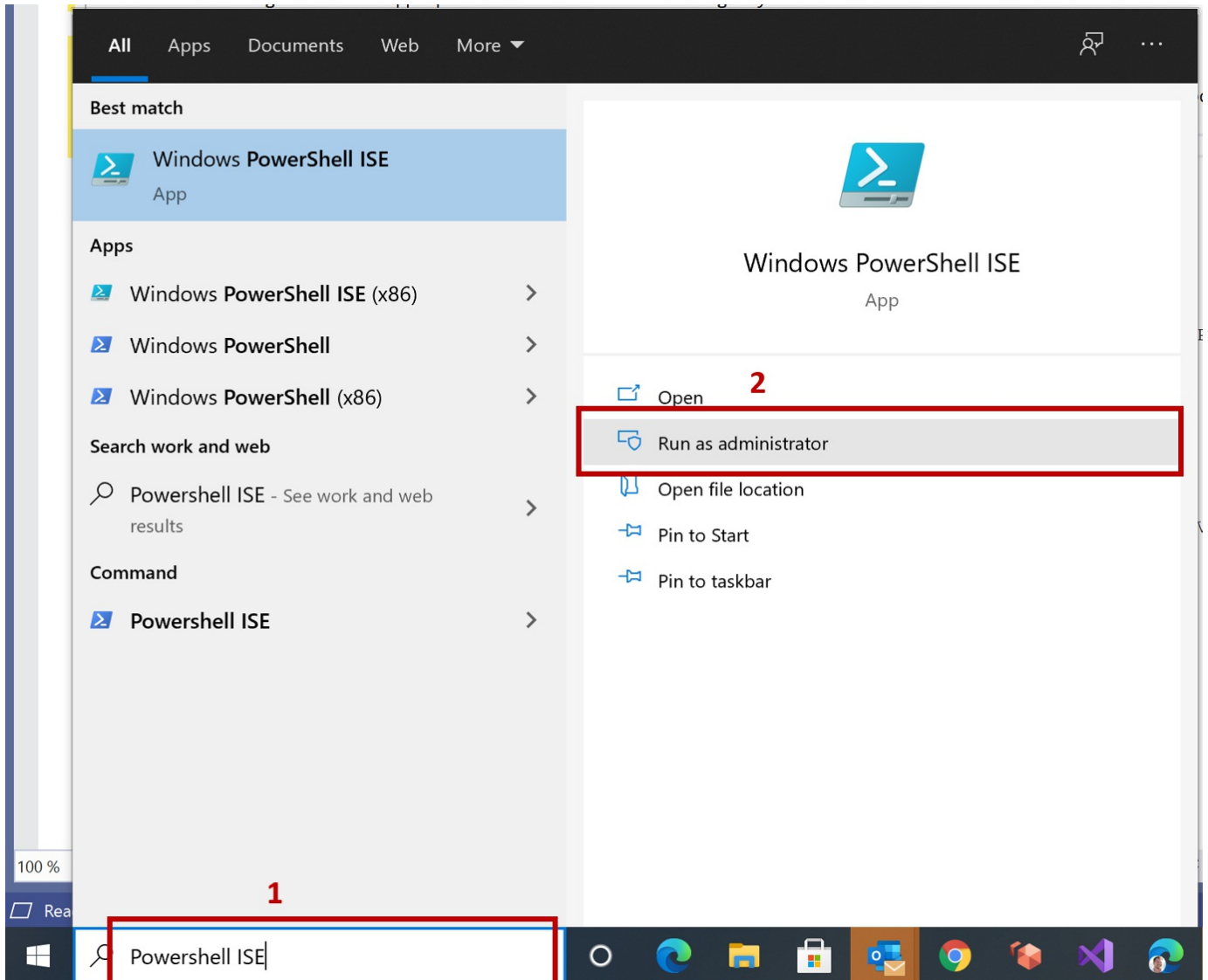


Figure 1: Open PowerShell ISE as Administrator.

Activation

NOTE

Any component of the PowerShell scripts below that are bookended by double quotes, must be replaced with use specific data (i.e., setting your own password).

Activating the Windows OS on Server1

After establishing your session in PowerShell ISE, execute the following cmdlets to activate Windows OS:

```
$task = Get-RpsTaskItem -WorkflowName "Set-WindowsActivation"  
$target = Get-RpsTargetItem -Type "VirtualMachine" -Name "Server1"  
$assignment = New-RpsTaskAssignment -TaskItem $task -TargetItem $target
```

Activating Office Products on Server1

After establishing your session in PowerShell ISE, execute the following cmdlets to activate Microsoft Office Products:

```
$task = Get-RpsTaskItem -WorkflowName "Set-OfficeActivation"  
$target = Get-RpsTargetItem -Type "VirtualMachine" -Name "Server1"  
$assignment = New-RpsTaskAssignment -TaskItem $task -TargetItem $target
```

More Resources

- [Token Based Software Activation with PowerShell](#)

Token Based Software Activation Using the User Interface

Last updated on April 9, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the Lead Service Integrator (LSI) or developer.

Introduction

During a new vehicle field activation, you will likely need to activate a Windows Operating System and/or Microsoft Office Products. There are three primary methods that can be used: PowerShell, runbooks, and through the Web User Interface. This article will demonstrate how to activate software using the Web User Interface.

Assumptions

The following is assumed prior to performing the actions described in this article:

- You have access to the RPS Graphic User Interface (GUI or UI).
- You are assigned to the appropriate role to make the changes you intend to make.
- The license certificates are available in RPS (see [Verifying Pre-Requisites](#) to confirm).

Verifying Pre-Requisites

The activation process for a Windows Operating System or Microsoft Office Product is the same. Before activating the license, you must verify the required license certificates are available in RPS.

1. From any screen in RPS, select **Targeting** to expand the dropdown in the navigation menu.
2. Click on **Items**.

The screenshot shows the RPS interface. At the top, a navigation bar includes 'RPS', 'Targeting', 'Resourcing', 'Tasking', 'Distribution', 'Admin', and 'Dashboard'. The 'Targeting' menu is open, showing options: 'Nodes', 'Containers', 'Item Groups', and 'Items'. A red box labeled '1' highlights the 'Targeting' menu, and another red box labeled '2' highlights the 'Items' option. Below the menu is a 'Filter' section with 'All', 'Name', and 'Status' filters. The 'Assignments' table is visible, with columns: 'Target Item', 'Target Group', 'Workflow', 'Task Map', 'Status', 'Assignment Date', 'Start Date', and 'Message'. The table contains one entry: 'APP.rps.local', 'Set-WindowsActivation', 'Ready', and '2021-02-17T15:06:59.798537'. The footer shows 'MS' and 'RPS © 2021'.

Target Item	Target Group	Workflow	Task Map	Status	Assignment Date	Start Date	Message
APP.rps.local		Set-WindowsActivation		Ready	2021-02-17T15:06:59.798537		

Figure 1: Select Targeting(1) and click Items(2).

3. Click on the [**Virtual Machine Name**] you would like to activate Windows OS or Office Products on.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, rps\RpsAdmin

Auto Refresh List

Filters

All Active InActive

Name

Item Type

Status

Properties

+ Add Filter

Items

Name ↑	Type	# Of Children	Errors	Active
AD.unit.domain	VirtualMachine	2		✓
AD.unit.domain	Drive	0		✓
AD.unit.domain-NIC1	NetworkConfiguration	0		✓
APP-S.rps.local	VirtualMachine	2		✓
APP-S.rps.local	Drive	0		✓
APP-S.rps.local-NIC1	NetworkConfiguration	0		✓
APP-S2.rps.local	Drive	0		✓
APP-S2.rps.local	VirtualMachine	2		✓
APP-S2.rps.local-NIC1	NetworkConfiguration	0		✓
3 APP.rps.local	VirtualMachine	2		✓
APP.rps.local	Drive	0		✓
APP.rps.local-NIC1	NetworkConfiguration	0		✓
Member.unit.domain	Drive	0		✓
Member.unit.domain	VirtualMachine	2		✓
Member.unit.domain-NIC1	NetworkConfiguration	0		✓
NOSC	Vehicle	2		✓
SNE	Vehicle	1		✓
TCN	Vehicle	2		✓

MS RPS © 2021

Figure 2: Example selecting an Item of VirtualMachine Type to activate Windows OS or Office Products on.

4. Scroll down the page to the Resource Assignments section and verify that both license certificates with the following names are available.

1. *Certificate-TokenBasedActivation-WindowsActivation*
2. *Certificate-MicrosoftProductActivationPCA2017_WindowsActivationCA*

NOTE

"PCA2017" may have a different name specific to the version of Office Product you are installing.

Certificate Name	Enabled	Status	Action
Certificate - APP-S.rps.local_DscEncryption-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_RpsClientCdn	True	Approved	Remove
Certificate - APP-S.rps.local_RpsClientCdn-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_RpsGuiSSL	True	Approved	Remove
Certificate - APP-S.rps.local_RpsGuiSSL-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_RpsSync	True	Approved	Remove
Certificate - APP-S.rps.local_RpsSync-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_RpsSyncSSL	True	Approved	Remove
Certificate - APP-S.rps.local_RpsSyncSSL-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_RpsWebAPISSL	True	Approved	Remove
Certificate - APP-S.rps.local_RpsWebAPISSL-CA_Enabled-1	False	Approved	Remove
Certificate - APP-S.rps.local_WinRm	True	Approved	Remove
Certificate - APP-S.rps.local_WinRm-CA_Enabled-1	False	Approved	Remove
Certificate - MicrosoftProductActivationPCA2017_WindowsActivationCA	True	Approved	Remove
Certificate - RpsRoot	True	Approved	Remove
Certificate - Site_MasterKeyEncryption	True	Approved	Remove
Certificate - Site_MasterKeyEncryption-CA_Enabled-1	False	Approved	Remove
Certificate - Site_NodeEncryption	True	Approved	Remove
Certificate - TokenBasedActivation_WindowsActivation	True	Approved	Remove
Credential - APP-S.Site_Administrator	False	Approved	Remove
Credential - rps_CdnSvc	False	Approved	Remove
Credential - rps_DatabaseAccount	False	Approved	Remove

Figure 3: Certificate Verification.

- At the top of the screen, select **Resourcing** to expand the dropdown in the navigation menu.
- Click on **Resources**.

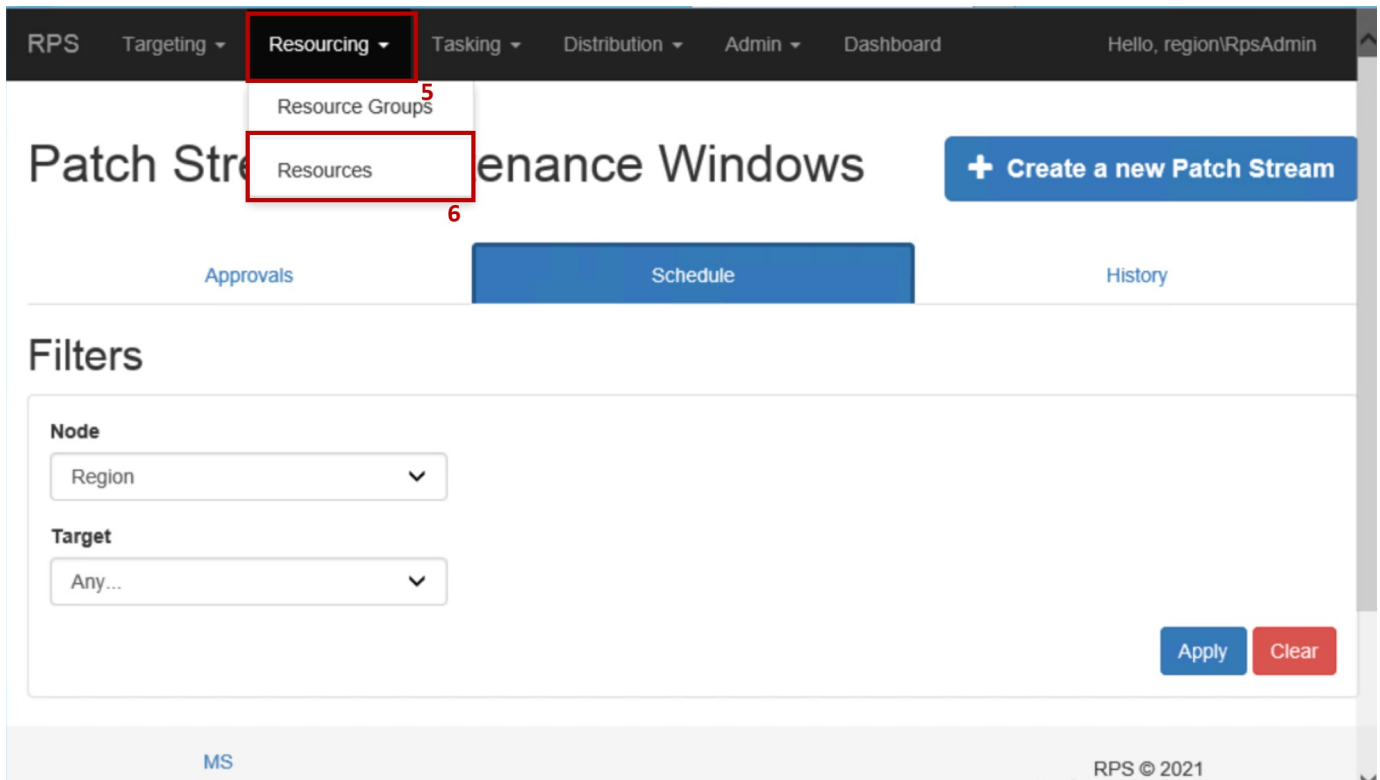


Figure 4: Select Resourcing(5) and click on Resources(6).

- On the left side, select the **Item Type** dropdown.

8. Select **SoftwareLicense**.

The screenshot shows the RPS Resources page. The top navigation bar includes 'RPS', 'Targeting', 'Resourcing', 'Tasking', 'Distribution', 'Admin', and 'Dashboard', along with the user 'Hello, region\RpsAdmin'. The 'Filters' section on the left has tabs for 'All', 'Active', and 'Inactive'. The 'Item Type' dropdown is open, showing a list of options with 'SoftwareLicense' highlighted and marked with a red box and the number '8'. The 'Resources' table on the right lists several Content Type Mapping items, with the first one highlighted and marked with a red box and the number '7'.

Name ↑	Type	Target Item Type	Global	Act
.config -> text/xml	ContentTypeMapping		✓	▼
.dll -> application/x-msdownload	ContentTypeMapping		✓	▼
.exe -> application/octet-stream	ContentTypeMapping		✓	▼
.json -> application/json	ContentTypeMapping		✓	▼

Figure 5: Select the Item Type dropdown(7) and select SoftwareLicense(8).

9. Click on the appropriate License.

The screenshot shows the RPS Resources page with the 'Filters' section updated. The 'Item Type' dropdown is now set to 'SoftwareLicense'. The 'Resources' table shows a single entry, 'Test Windows License', which is highlighted with a red box and the number '9'. The table has columns for 'Name ↑', 'Type', 'Target Item Type', 'Global', and 'Active'. Action buttons for '+ Add Resource', 'Edit', and 'Remove' are visible for the selected resource.

Name ↑	Type	Target Item Type	Global	Active	
Test Windows License	SoftwareLicense		✓	✓	+ Add Resource Edit Remove

Figure 6: Click on the appropriate License.

10. Verify that Role is set to **Windows**.

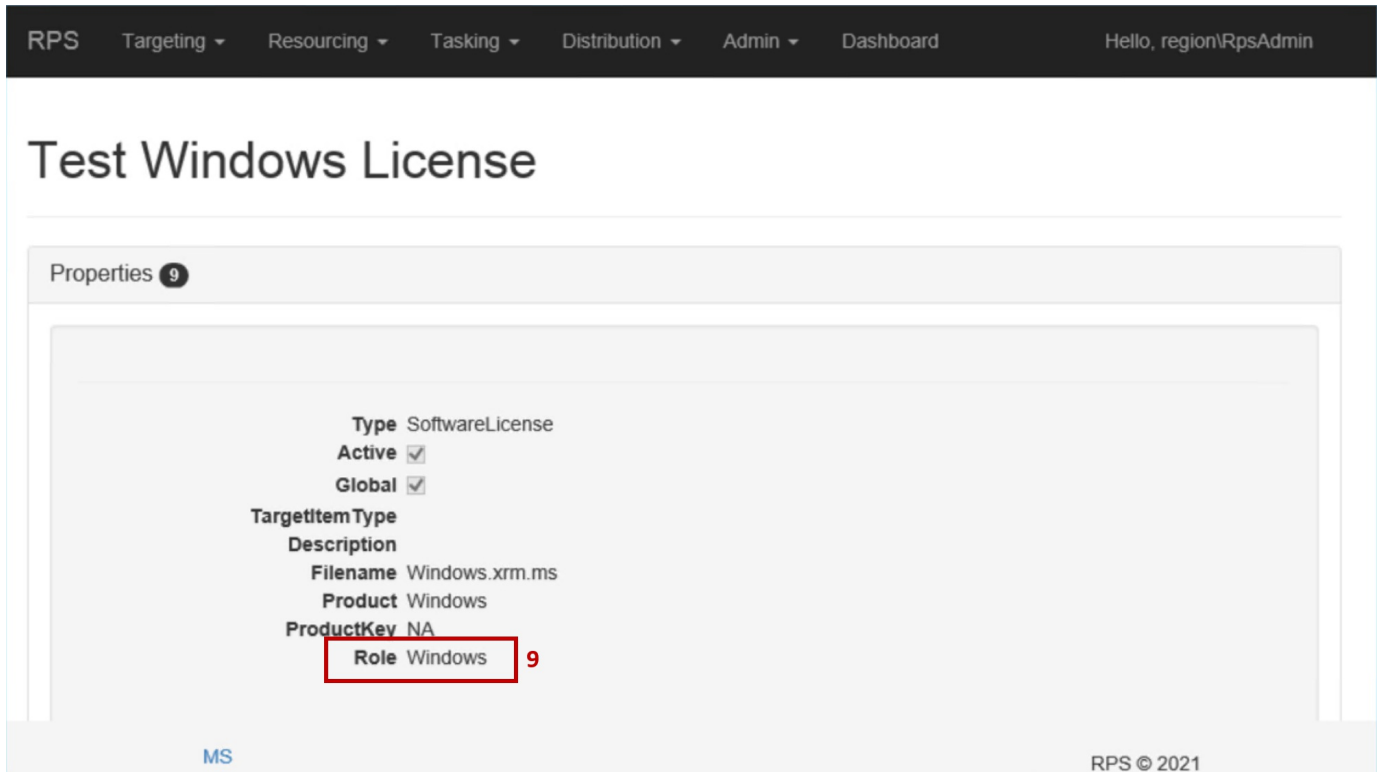


Figure 7: Verify Role is set to Windows.

Installation

After successfully verifying the pre-requisites, the subsequent instructions can be followed to activate either the Windows OS or Office Products.

1. From any screen in RPS, select **Targeting** to expand the drop down in the navigation menu.
2. Click on **Items**.

The screenshot shows the RPS interface. At the top, a navigation bar includes 'RPS', 'Targeting', 'Resourcing', 'Tasking', 'Distribution', 'Admin', and 'Dashboard'. The 'Targeting' menu is open, showing options: 'Nodes', 'Containers', 'Item Groups', and 'Items'. A red box labeled '1' highlights the 'Targeting' menu, and another red box labeled '2' highlights the 'Items' option. Below the menu is a 'Filter' section with 'All' and 'Name' buttons, and a 'Status' dropdown set to 'Any ...'. The main area is titled 'Assignments' and contains a table with the following data:

Target Item ↑	Target Group	Workflow	Task Map	Status	Assignment Date	Start Date	Message
APP.rps.local		Set-WindowsActivation		Ready	2021-02-17T15:06:59.798537		

At the bottom of the page, there is a footer with 'MS' and 'RPS © 2021'. The browser address bar shows 'https://app.rps.local:8080/ItemGroups'.

Figure 8: Select Targeting(1) and click Items(2).

3. Click on the [**Virtual Machine Name**] you would like to activate Windows OS or Office products on.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, rps\RpsAdmin

Auto Refresh List

Filters

All Active InActive

Name

Item Type

Status

Properties

+ Add Filter

Items

Name ↑	Type	# Of Children	Errors	Active
AD.unit.domain	VirtualMachine	2		✓
AD.unit.domain	Drive	0		✓
AD.unit.domain-NIC1	NetworkConfiguration	0		✓
APP-S.rps.local	VirtualMachine	2		✓
APP-S.rps.local	Drive	0		✓
APP-S.rps.local-NIC1	NetworkConfiguration	0		✓
APP-S2.rps.local	Drive	0		✓
APP-S2.rps.local	VirtualMachine	2		✓
APP-S2.rps.local-NIC1	NetworkConfiguration	0		✓
3 APP.rps.local	VirtualMachine	2		✓
APP.rps.local	Drive	0		✓
APP.rps.local-NIC1	NetworkConfiguration	0		✓
Member.unit.domain	Drive	0		✓
Member.unit.domain	VirtualMachine	2		✓
Member.unit.domain-NIC1	NetworkConfiguration	0		✓
NOSC	Vehicle	2		✓
SNE	Vehicle	1		✓
TCN	Vehicle	2		✓

MS RPS © 2021

Figure 9: Example selecting an Item of VirtualMachine Type to activate Windows OS or Office Products on.

4. Scroll down to the Task Assignments section of the page.

NOTE

If you cannot see the blue + **Add Task Assignment** button, you can expand the section by clicking on the **Task Assignment** section header.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, rps\RpsAdmin

DtsrRpcPort 5735
 DnsZone rps.local
 CanPatch True
 IsGui True
 Architecture x64
 CanManageCertificate True
 IsAppliance False
 IsCDN True
 IsDB True
 IPAddress 192.0.0.202
 Generation 2
 ImagesParentPath C:\ContentStore
 ImportFilename TCN.xml

[Edit](#)

Child Items 2

Name	Type	Active
APP-S.rps.local	Drive	✓
APP-S.rps.local-NIC1	NetworkConfiguration	✓

[+ Add Child Target Item](#)

Task Assignments 0 4

Task	Date	Status	Message
------	------	--------	---------

Resource Assignments 49

Resource	Resource Group	Target Group	Global	State	Status Changes
ADDomain - rps.local			True	Approved	Remove
Certificate - APP-S.rps.local_DscEncryption			True	Approved	Remove
Certificate - APP-S.rps.local_DscEncryption-CA_Enabled-1			False	Approved	Remove
Certificate - APP-S.rps.local_RpsClientCdn			True	Approved	Remove

MS RPS © 2021

Windows taskbar: 3:08 PM 2/17/2021

Figure 10: Expand Task Assignment.

5. Click on **+ Add Task Assignment**.

RPS Targeting Resourcing Tasking Distribution Admin Dashboard Hello, region\RpsAdmin

[Edit](#)

Child Items 2

Name	Type	Active
APP-S.region.rps-NIC1	NetworkConfiguration	✓
APP-S.region.rps	Drive	✓

[+ Add Child Target Item](#)

Task Assignments 0 5

Task	Date	Status	Message
------	------	--------	---------

[+ Add Task Assignment](#)

Resource Assignments 50

MS <https://app-s.region.rps:8080/>

Activate Windows
 Go to System in Control Panel to activate Windows.
 RPS © 2021

Figure 11: Click + Add Task Assignment.

6. In the New Task Assignment pop-up box, select the appropriate Task.
 - o To activate a Windows OS, select **Set-WindowsActivation**.
 - o To activate Office products, select **Set-OfficeActivation**.
7. Click **Assign**.

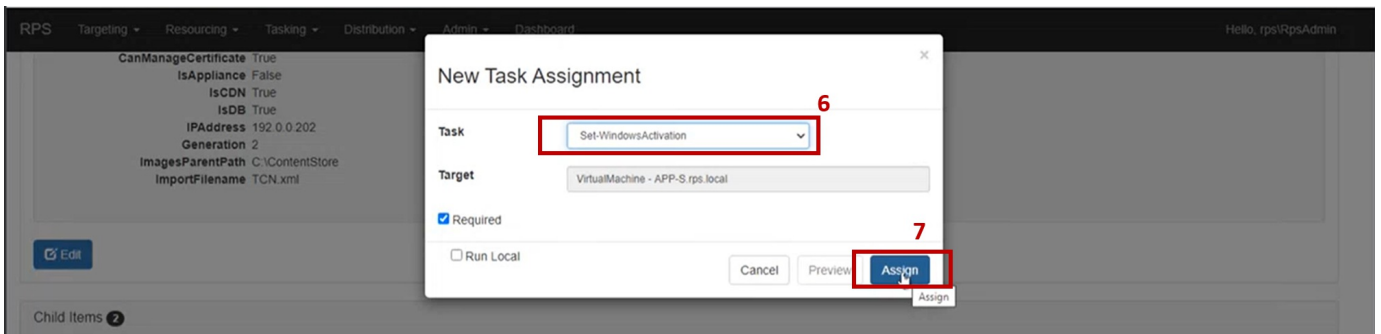


Figure 12: Select the appropriate Task(6) and click Assign(7).

Verifying Activation Status

You can verify the activation status by checking the Task Assignments status screen.

1. From any screen in RPS, select **Tasking** to expand the dropdown in the navigation menu.
2. Click on **Assignments**.

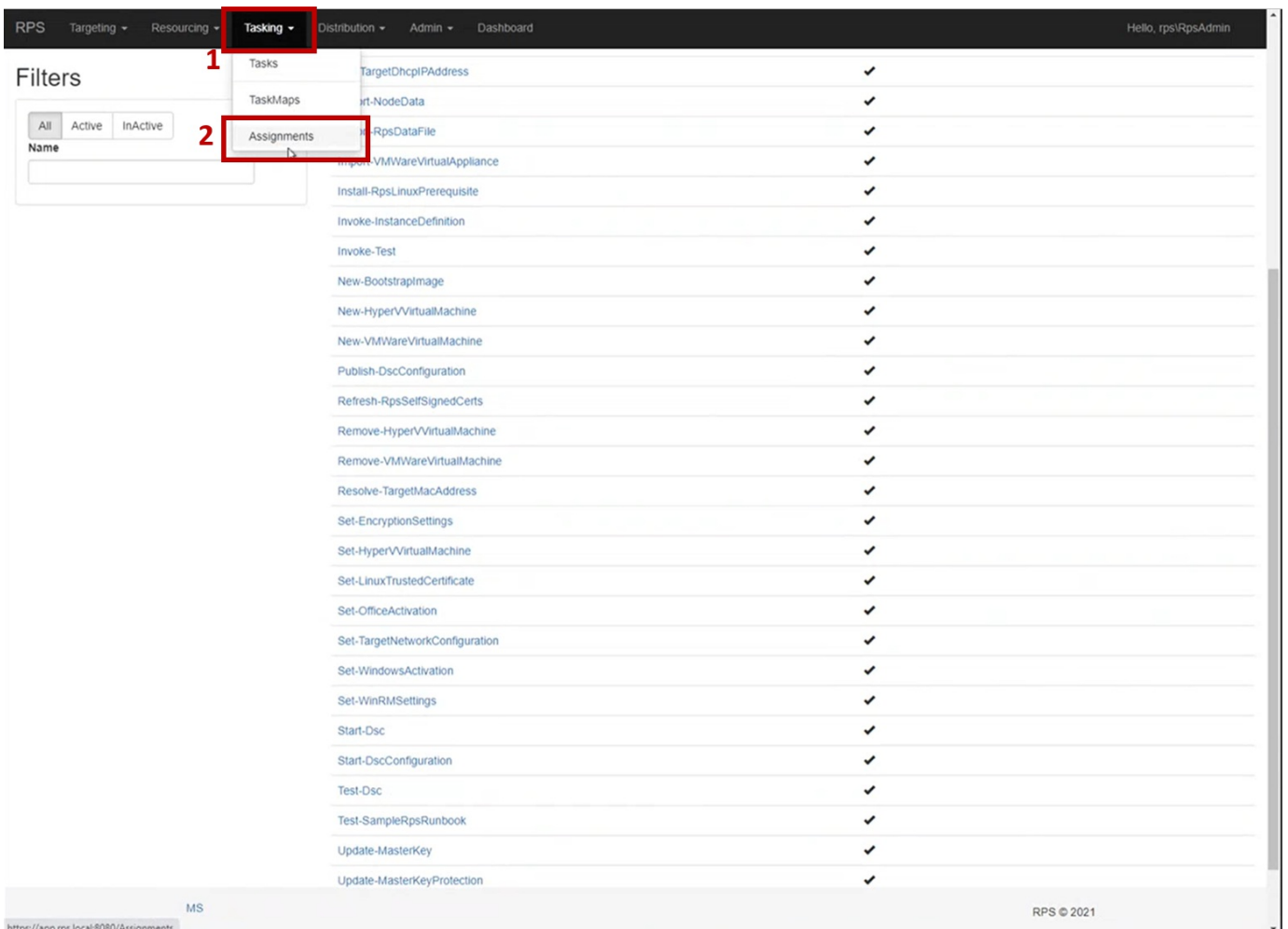


Figure 13: Select Tasking(1) and click on Assignments(2).

3. The dashboard will show you the status of the Task Assignment Activation.

- *Ready* shows the activation is queued up and will run.
- *Completed* shows that the Task Assignment has been activated.

The screenshot shows the RPS dashboard interface. At the top, there is a navigation bar with 'RPS' and several menu items: Targeting, Resourcing, Tasking, Distribution, Admin, and Dashboard. On the right side of the navigation bar, it says 'Hello, rps\RpsAdmin' and 'Auto Refresh List'. Below the navigation bar, there is a 'Filters' section on the left with buttons for 'All', 'Active', and 'Inactive', a 'Name' search field, and a 'Status' dropdown menu. The main area is titled 'Assignments' and contains a table with the following data:

Target Item ↑	Target Group	Workflow	Task Map	Status	Assignment Date	Start Date	Message
APP-S.rps.local		Set-WindowsActivation		Ready	2021-02-17T15:08:33.014991		
APP.rps.local		Set-WindowsActivation		Completed	2021-02-17T15:06:59.796537		Completed Set-WindowsActivation on APP.rps.local

At the bottom of the dashboard, there is a footer with 'MS' on the left and 'RPS © 2021' on the right. A URL is visible at the bottom left: <https://app.rps.local:8080/Items/45af0ebe-76ba-4f27-a1be-440b102ac421>.

Figure 14: Activation Status.

Token Based Software Activation with MNActivation Tool

Last updated on May 13, 2022.

Last Reviewed and Approved on PENDING REVIEW

Use Cases

The MNActivation Tool is intended to run on standalone workstations and servers that are not provisioned using RPS but are part of Missions Network's (MN) SIPR and XLESS environments.

Pre-Requisites to Run the MNActivation Tool

NOTE

The MNActivation Tool is completely self-contained and does not need any additional programs installed in order to run.

Pre-Requisites: User

The tool must be run using the built-in Administrator account.

```
Get-LocalUser | Where-Object {$_.SID -like "*-500"}
```

Pre-Requisites: Target Machine

- The target machine must be running on MN's SIPR or XLESS environment.
- The target machine was provisioned using a TBA Ready REACTR Image (February 2021 or later).
- The Windows Operating System must be one of the following:
 - Windows 10 Enterprise LTSC 2019 (Based on 1809)
 - Windows 10 Enterprise LTSC 2021 (Based on 21H2)
 - Windows Server 2012 R2 Standard
 - Windows Server 2019 Datacenter
- The Office version must be a Volume License install.
- The Office version must be one of the following:
 - Office 2013 Professional Plus
 - Office 2013 Visio Professional
 - Office 2016 Professional Plus
 - Office 2016 Visio Professional
 - Office 2019 Professional Plus
 - Office 2019 Visio Professional

How to Run the Tool

NOTE

It is not required to run the MNActivation Tool locally. The tool can be run from the local drive, a USB drive, or a network share.

1. Double click the MNActivation Tool EXE.
2. The tool will prompt the user for consent. Enter **Y** or **y** and strike the [Enter] key.
3. After completion, the tool will provide an ending prompt. Pressing any key will close the prompt.

⚠ IMPORTANT

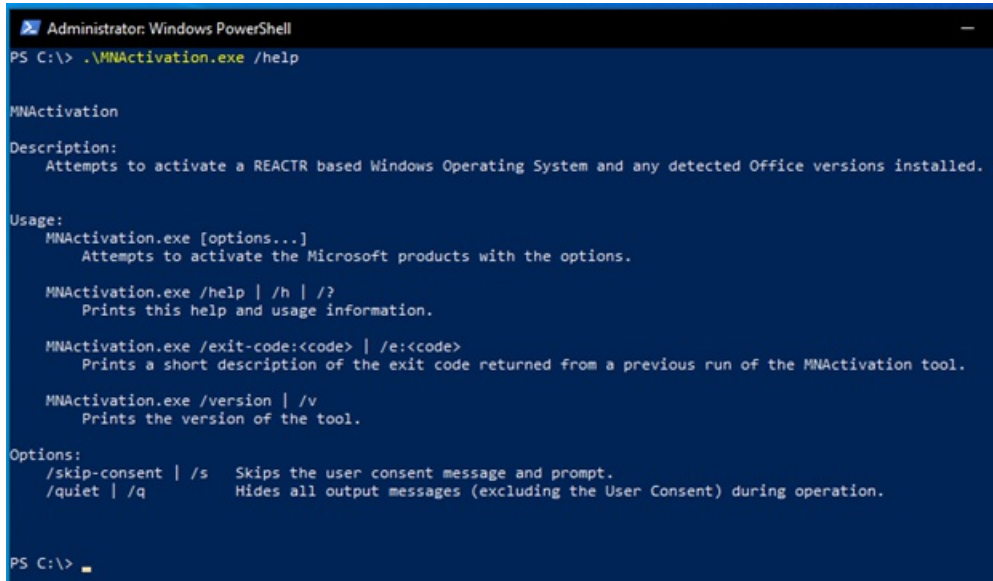
The target machine should **NEVER** be connected to the internet once the MNActivation Tool has been used!

Command Line Functionality

The MNActivation Tool has limited command line options available. It can be leveraged to run the tool using a PowerShell script, for example.

To display the command line options:

```
MNActivation.exe /help
```



```
Administrator: Windows PowerShell
PS C:\> .\MNActivation.exe /help

MNActivation
Description:
  Attempts to activate a REACTR based Windows Operating System and any detected Office versions installed.

Usage:
  MNActivation.exe [options...]
  Attempts to activate the Microsoft products with the options.

  MNActivation.exe /help | /h | /?
  Prints this help and usage information.

  MNActivation.exe /exit-code:<code> | /e:<code>
  Prints a short description of the exit code returned from a previous run of the MNActivation tool.

  MNActivation.exe /version | /v
  Prints the version of the tool.

Options:
  /skip-consent | /s  Skips the user consent message and prompt.
  /quiet | /q        Hides all output messages (excluding the User Consent) during operation.

PS C:\>
```

Figure 1: MNActivation Tool Command Line Help Output.

Scripting and Automating

For fully silent execution (hides output, does not prompt for user consent, and does not do a completion prompt):

```
MNActivation.exe /skip-consent /quiet
```

Same functionality as above, but in shorthand:

```
# shorthand
MNActivation.exe /s /q
```

Examples

Without Office Products Installed

```
C:\Users\LocalAdmin\Desktop\MNAActivation_1.2.0.exe
You are about to run Mission Network's Activation Tool.
This tool should ONLY be used on XLESS and SIPR machines.
This tool is only intended to run on REACTR made images, it will not work otherwise.
DO NOT RUN THIS OUTSIDE OF XLESS AND SIPR
Do you wish to continue execution of the MN Activation Tool?
Enter Y/y to continue:
y
Validating required certificates...
  Reactr Encryption certificate...
  Reactr Encryption Private Key access...
  Microsoft Root certificate...
  Microsoft Intermediate Root...
  Activation Certificate (SoftCert)...
  Activation Certificate (SoftCert) Private Key access...
Validating correct Issuance License is installed...
Configuration machine for Token-Based Activation...
  Updating installed License Key to be the KMS Client Key...
  Refreshing Licensing status...
Activating Operating System...
  Creating Challenge...
  Requesting Authorization...
  Authorizing Request...
  Refreshing Licensing status...
  **SUCCESS**
Activating Office...
  Searching for Office installs...
  No Office installs found.

=====
Activation Program is complete. Press any key to exit...
```

Figure 2: Without Office Products Installed.

With Office Products Installed

```
C:\Users\LocalAdmin\Desktop\MNAActivation_1.2.0.exe
You are about to run Mission Network's Activation Tool.
This tool should ONLY be used on XLESS and SIPR machines.
This tool is only intended to run on REACTR made images, it will not work otherwise.
DO NOT RUN THIS OUTSIDE OF XLESS AND SIPR
Do you wish to continue execution of the MN Activation Tool?
Enter Y/y to continue:
y
Validating required certificates...
  Reactr Encryption certificate...
  Reactr Encryption Private Key access...
  Microsoft Root certificate...
  Microsoft Intermediate Root...
  Activation Certificate (SoftCert)...
  Activation Certificate (SoftCert) Private Key access...
Validating correct Issuance License is installed...
Configuration machine for Token-Based Activation...
  Updating installed License Key to be the KMS Client Key...
  Refreshing Licensing status...
Activating Operating System...
  Creating Challenge...
  Requesting Authorization...
  Authorizing Request...
  Refreshing Licensing status...
  **SUCCESS**
Activating Office...
  Searching for Office installs...
  Validating the Office installs can be activated via Token-Based Activation...
  Activating Office 15, OfficeProPlusVL_KMS_Client edition...
  Configuring install for Token-Based Activation...
  Creating Challenge...
  Requesting Authorization...
  Authorizing Request...
  **SUCCESS**

=====
Activation Program is complete. Press any key to exit...
```

Figure 3: With Office Products Installed.

Release Notes

File Version: Found by right-clicking the MNActivation Tool EXE file and navigating to the Properties tab.

Assembly Version: Found by executing `MNActivation.exe /version` in the command line.

RELEASE DATE	FILE VERSION	ASSEMBLY VERSION	NOTES
2021-03-10	1.0.0	1.0.7738.25976	Initial Release
2021-08-11	1.1.0	1.1.7893.29909	Correctly sets the Operating System's Activation Type to be Token Only .
2022-05-13	1.2.0	1.2.8168.13739	Add support for <i>Windows 10 LTSC 2021 (based on 21H2)</i> . Upgrade to .NET 6 LTS. Force system to use KMS Client Key to ensure activation success.

RPS Customization Guide - High Level Overview

Last updated on March 15, 2019.

Last Reviewed and Approved on PENDING REVIEW

RPS provides the ability for the admin to customize some aspects of the web based GUI. This functionality allows the admin to have the application better fit within existing branding and theming guidelines that maybe already be established.

Footer

The footer for the web user interface is off by default. Within the Web.Config file you can enable it so that it will be shown on all pages within the web application.

For configuring the footer, you will need to add a section declaration and then create the config section called `footerConfig`.

Section declaration

Within the Web.Config you will want to add the following section declaration to the configuration > configSections element.

```
<section name="footerConfig" type="Rps.Web.ConfigSections.FooterSection"/>
```

Once it is added, your configSections element should look like this:

```
<configSections>
  <section name="footerConfig" type="Rps.Web.ConfigSections.FooterSection"/>
</configSections>
```

footerConfig Element

With the `footerConfig` section has been declared in the configSections element, you can add the following `footerConfig` element after the `appSettings` element:

```
<footerConfig show="false">
  <links></links>
</footerConfig>
```

Figure 2 FooterConfig within the Web.Config

To enable the footer to be displayed, set the show attribute to true.

Adding links within the footerConfig

Within the `footerConfig` element there is a child element called `links`. This will take a variable number of `link` elements to be displayed on the footer.

A `link` element has two attributes, the text attribute and URL attribute. The text attribute controls the text that will be displayed to the user. The URL attribute is the actual location that user will be sent to when they click the displayed link.

For example, adding a link to the Microsoft & MSDN web sites, the configuration look like so:

```
<add text="Microsoft" url="https://microsoft.com" />
<add text="MSDN" url="https://msdn.microsoft.com" />
```

Resulting in a view to the user as such:

NOTE

Note that the ASP.Net runtime monitors the Web.Config file to detect any changes that are made. This will cause the web application to restart and then load the changes. This should only take a matter of seconds for the application to restart.

Branding Customizations

RPS allows the header of the web application to be customized with specific branding so the application can look like existing properties within the organization.



Figure 3 The RPS Header that can be configured.

Icon

The RPS Header can be configured to show an icon to show on the left side of the header. When using the setting, the value should represent a path that reflects the location of the image from within the application.

For example, if the image is stored at C:\WebApps\RPS\Images\brand.png and the root of the web application points to C:\WebApps\RPS then the path would be /Images/brand.png just like it would be defined within an HTML document.

We recommend an image of 50px by 50px for best results.

Setting Name

`ui:BrandImage`

Example

```
<add key="ui:BrandImage" value="/Images/brand.png" />
```

Will update the header to look like this:



Default:

No Image will be displayed

Brand Text

The brand text configuration option allows the name of the web application to be changed within the header.

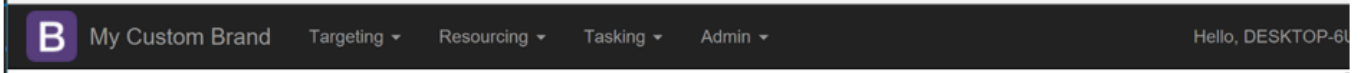
Setting Name

`ui:Brand`

Example

```
<add key="ui:Brand" value="My Custom Brand" />
```

Will update the header to look like this:



Default:

The value "RPS" will be displayed.

Greeting Text

The greeting text configuration allows the administrator to update the text displaying before the logged in user's name. This can be set to anything or nothing if so configured.

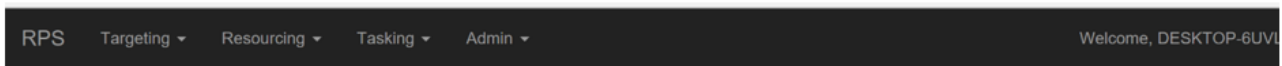
Setting Name

`ui:Greeting`

Example

```
<add key="ui:Greeting" value="Welcome, " />
```

Will update the header to look like this:



Default:

The value "Hello, " will be displayed before the user's name.

Theming

Theming within RPS allows the administrator to update some color and font aspects of the web application to best match the existing look and feel that is already established. A theme consists of background color, font color, font type and font size. There are two main areas for theming, the header and the content area. Each of the areas is configurable within the applications Web.Config.

Header

Header theming allows customization of background color, font and font-size for top most portion of the web application.

Background Color

Setting

```
ui:HeaderBackgroundColor
```

Example

```
<add key="ui:HeaderBackgroundColor" value="#FF00FF" />
```

This will turn the header background color to Fuchsia.

Default

The default color will be the HTML Hex color code #222222

Font

Setting

```
ui:HeaderFont
```

Example

```
<add key="ui:HeaderFont" value="Segoe UI" />
```

This will turn all the header fonts to Segoe UI.

Default

The default font is defined as `"Helvetica Neue", Helvetica, Arial, sans-serif`.

Font Color

Setting

```
ui:HeaderFontColor
```

Example

```
<add key="ui:HeaderFontColor" value="#F0F8FF" />
```

This will turn the header font color to Alice Blue.

Note: This will only change the text color for the header and the top navigation item. The dropdown list and text will not be affected.

Default

The default color will be the HTML Hex color code #9D9D9D

Font Size

This setting will take any valid CSS defined size. This includes absolute size values, relative size values, pixel and percentage sizes. The values can be found here. This setting is applied to all the fonts within the header.

Setting

```
ui:HeaderFontSize
```

Example

```
<add key="ui:HeaderFontSize" value="x-large" />
```

This will turn the header font size to extra-large.

Default

The default font size for the header text varies but ranges from 10px to 14px.

Footer

Footer theming allows customization of background color, font and font-size for top most portion of the web application.

Background Color

Setting

```
ui:FooterBackgroundColor
```

Example

```
<add key="ui:FooterBackgroundColor" value="#FF00FF" />
```

This will turn the footer background color to Fuchsia.

Default

The default color will be the HTML Hex color code #222222

Font

Setting

```
ui:FooterFont
```

Example

```
<add key="ui:FooterFont" value="Segoe UI" />
```

This will turn all the footer fonts to Segoe UI.

Default

The default font is defined as `"Helvetica Neue", Helvetica, Arial, sans-serif`.

Font Color

Setting

```
ui:FooterFontColor
```

Example

```
<add key="ui:FooterFontColor" value="#F0F8FF" />
```

This will turn the footer font color to Alice Blue.

Note: This will only change the text color for the footer and the top navigation item. The dropdown list and text will not be affected.

Default

The default color will be the HTML Hex color code #9D9D9D

Font Size

This setting will take any valid CSS defined size. This includes absolute size values, relative size values, pixel and percentage sizes. The values can be found here. This setting is applied to all the fonts within the footer.

Setting

```
ui:FooterFontSize
```

Example

```
<add key="ui:FooterFontSize" value="x-large" />
```

This will turn the footer font size to extra-large.

Default

The default font size for the footer text varies but ranges from 10px to 14px.

More Resources

- [RPS Automation Package Guidelines](#)
- [RPS IPSheet Parser](#)

RPS IPSheet Parser

Last updated on March 15, 2019.

Last Reviewed and Approved on PENDING REVIEW

The RPS IPSheet Excel file is a Microsoft Excel spreadsheet that contains CMDB endpoint data for a customer. The IPSheet will contain a set of worksheets for NIPR, SIPR, and Colorless. Each worksheet will contain Endpoint data for Subnets and IPAddresses needed for the CMDB. These Subnets and IPAddresses will be stored as Resource Groups and Items in the RPS database.

A sample IPSheet Excel file can be found at:

```
RPS > Source > Rps.Extension.IpSheet > Samples > TestIpSheet.xlsx
```

Assumptions

- The 1st column in the worksheet is ignored.
- Subnet/IPAddress are used as the name of the imported Subnet or IpAddress and are expected to be unique.
- Column headers are exactly, "Telephony Rng", "Subnet", "Mask", "Gateway", "Reserved", "Assignment"
- Subnet name is expected to be named "SIPR", "NIPR" or "Colorless", exactly.
- Reserved IPAddresses are expected to be listed in the same order as the Subnets are.
- Each set of reserved IpAddresses for a Subnet will list the subnet name in the first row ONLY.
- **Note:** This is important because subnet names aren't unique ("Tactical Spare") so we use a combination of sequence and presence of the name to identify a new group, rather than IP Logic.
- Row 1 contains column headers.
- Row 2 contains header information.
- Rows 3 -> x contain Subnets.
- Row x+1 contains value "Unit Base Info" in the 2nd column.
- Row y contains value "Detailed Info" in the 2nd column.
- Row y+1 -> z-1 contains reserved IPAddresses.
- Row z contains value "End" in the 2nd column

RPS IPSheet Parser .NET Library (DLL)

The RPS IPSheet Parser is a .NET library (DLL) written in C#. This library contains two key Powershell Cmdlets for importing or removing IPSheet data to/from the RPS CMDB database.

The PS Module defined by the RPS IPSheet Parser library assembly is imported in Powershell through the module's manifest (psd1). Subsequently, the Cmdlets can be called to import and parse IPSheet data into RPS.

Example:

```
Import-Module Rps-IPSheet
```

Import-RpsIpSheet Cmdlet

The Import-RpsIpSheet Powershell Cmdlet is used to import IPSheet data into the RPS database. This cmdlet accepts one parameter "Path" that accepts an absolute or relative path to the IPSheet Excel file location. Results will show the total number of Subnets and IpAddresses imported. Warnings will show any import errors, including invalid file paths.

NOTE

The import will not overwrite or merge with existing data, so if there is a collision on subnet or ipaddress, then a warning is displayed.

Example:

```
Import-RpsIpSheet -Path c:\ipsheet\abc-tactical-2018.xls
```

```
Import-RpsIpSheet -Path .\abc-tactical-2018.xls
```

Remove-RpsIpSheetCmdlet

The Remove-RpsIpSheet Powershell Cmdlet is used to remove previously imported IPSheet data from the RPS database. This cmdlet has no parameters. It will remove all previous imports of IPSheet data. Results will display all resource groups (subnets) and resource items (ip addresses) removed from the RPS database.

Example:

```
Remove-RpsIpSheet
```

Powershell Script

The IPSheet data can be imported into an RPS database using Powershell. The Powershell script will import PS Module from the RPS.Extension.IpSheet .NET assembly. The PS Module is distributed to the Content Store located at ContentStore\Modules\Rps-IpSheet. Then the Import-RpsIpSheet cmdlet is called with an absolute or relative path to the location of the IpSheet Excel file that contains the Subnet and IpAddress endpoint data for the CMDB.

Example:

```
Import-Module Rps-IpSheet
$start = Get-Date
Import-RpsIpSheet -Path C:\ipsheet\86ESB-A-Tactical-Template-11-17-2015.xls -Verbose
(Get-Date).Subtract($start).TotalSeconds
```

More Resources

- [RPS Automation Package Guidelines](#)
- [RPS Customization Guide - High Level Overview](#)

Introduction to RPS Instance Definitions

Last updated on May 12, 2021.

Last Reviewed and Approved on PENDING REVIEW

Introduction

RPS Instance Definitions are an abstraction layer on top of existing RPS Types. Types currently allow us to define a specific Type of Resource or Target Item, its expected or possible Properties and their associated value types (string/int/etc...). Instance Definitions will allow us to take these generic Types and combine them together to create complex, nested, entities composed of many different Types through Parent/Child and assignment relationships and the known default values for each individual Types' properties.

Examples: SNEs, TCNs, TSI-Large, laptop, could be a VM (because it could be a collection of components)

Instance Definition Fundamentals

Instance Definition Reference

RPS Instance Definition Reference is the assignment of the instance definition to a root Resource Item, which results in a set of one or more Resource Items to be run. This is also referred to as an "Instance Definition". The Instance Definition Reference classes are able to create assignments between all of the items in the hierarchy and are able to invoke everything that is not at the top Instance Definition level.

Why RPS Instance Definition?

We have defined many existing Types such as Vehicle, Computer, VirtualMachine, NetworkConfiguration, and so on. The entities we are now interacting with though are complex and composed of many of these Types in Parent/Child hierarchies. Instance Definitions are collections of Type Definitions that have the ability to create an object. Type Definitions are individual parts that define the properties of an object; however, they do not create an object on their own.

Example: Virtual Machine, NIC, drive, etc.

NOTE

The word Instance Definition is formerly known as 'Templates'.

Instance Definition Terms and Definitions

- **Instance Definition** The abstraction layer on top of existing RPS Types.
- **Instance Definition Reference** The assignment of the instance definition to a root Resource Item, which results in a set of one or more Resource Items to be run.
- **Instance Definition Item** The item on the abstraction layer on top of existing RPS Types.
- **Instance Definition Node** A node item that will result in association of a node with target items that are created when an Instance Definition is invoked.

Instance Definition PowerShell Cmdlets

Getting an Instance Definition

Gets an Instance Definition by Id.

```
Get-RpsInstanceDefinition -Id $lookupId
```

Setting an Instance Definition

Creates or Updates an Instance Definition.

```
$instanceDef = Set-RpsInstanceDefinition -Name "MyInstanceDef" -Properties @{Prop1 = "Value1"} -ParentNodeId $nodeId
```

Invoking an Instance Definition

Creates instances based on the Instance Definition using a Resource Item to hold its settings.

```
$settings = Get-RpsResourceItem -Id $guid  
Invoke-RpsInstanceDefinition -settings $settings
```

Creating an Instance Definition

Creates a new Instance Definition.

```
New-RpsInstanceDefinition -Name testName
```

Creating a New Instance Definition With Properties

```
$hs = @{  
    Prop1 = "value1"  
    Prop2 = "value2"  
}  
New-RpsInstanceDefinition -Name testName -Properties $hs
```

Creating a New Instance Definition With Properties and a Parent Node

```
$nodeId = "81B8272D-B49C-4350-A8F4-ABBB9CE29C68"  
$hs = @{  
    Prop1 = "value1"  
    Prop2 = "value2"  
}  
New-RpsInstanceDefinition -Name testName -Properties $hs -ParentNodeId $nodeId
```

Creating a New Instance Definition With Virtual Machine and Network Configuration


```

$vmTypeDef = Get-RpsResourceItem -Name VirtualMachine -Type RpsTargetType
$nicTypeDef = Get-RpsResourceItem -Name NetworkConfiguration -Type RpsTargetType
$credentialTypeDef = Get-RpsResourceItem -Name Credential -Type RpsResourceType

$vmAdServer = New-RpsInstanceDefinitionItem -EntityName "Ad.[^DomainName]" -Name AdServer -TypeDefinition
$vmTypeDef -Properties @{
    JoinDomain = $false
    ComputerName = 'AD'
    DnsZone = '[^DomainName]'
    OSType = 'Windows'
    OSVersion = '8.1'
    MemoryMB = 1024
    IsDC = $true
}
$vmAppServer = New-RpsInstanceDefinitionItem -EntityName "App.[^DomainName]" -Name AppServer -TypeDefinition
$vmTypeDef -Properties @{
    JoinDomain = $true
    ComputerName = 'APP'
    DnsZone = '[^DomainName]'
    OSType = 'Windows'
    OSVersion = '8.1'
    MemoryMB = 2048
    IsCDN = $true
    IsDB = $true
    IsTms = $true
}
$nicVlan996 = New-RpsInstanceDefinitionItem -EntityName "[^ParentName]-Vlan996" -Name Nic-Vlan996 -
TypeDefinition $nicTypeDef -Properties @{
    Primary = $true
    VlanId = 996
    NetworkCategory = 'DomainAuthenticated'
}
$credentialRpsAdmin = New-RpsInstanceDefinitionItem -EntityName "[^DomainPrefix]_RpsAdmin" -Name
Credential_RpsAdmin -TypeDefinition $credentialTypeDef -Properties @{
    UserName = '[^DomainPrefix]\RpsAdmin'
    Role = 'ServerAdmin'
    IsLocal = $false
    CreateAccount = $true
    PasswordNeverExpires = $false
}
$instanceDefinition = New-RpsInstanceDefinition -Name Vehicle -Properties @{DomainName = 'RequiredValue';
DomainPrefix = 'RequiredValue'}
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $vmAppServer
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $vmAdServer
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $nicVlan996 -ParentItem $vmAppServer
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $nicVlan996 -ParentItem $vmAdServer
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $credentialRpsAdmin
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $credentialRpsAdmin -ParentItem $vmAppServer
New-RpsInstanceDefinitionReference -Name Vehicle_AppServer -InstanceDefinition $instanceDefinition -
InstanceDefinitionItem $credentialRpsAdmin -ParentItem $vmAdServer
$settingsResourceItem = New-RpsResourceItem -Type Settings -Name UnitASettings -Properties @{
    DomainName = 'Master.Rps'
    DomainPrefix = 'Master'
}
Invoke-RpsInstanceDefinition -InstanceDefinition $instanceDefinition -settings $settingsResourceItem

```

More Resources

- [RPS Instance Definition Item](#)
- [RPS Instance Definition Node](#)

RPS Instance Definition Item

Last updated on June 25, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by the developer.

Introduction

RPS Instance Definition Items are items on an abstraction layer on top of existing RPS Types. An Instance Definition Item is a wrapper for an RPS Type and associated Properties.

Terms and Definitions

- **Instance Definition** The abstraction layer on top of existing RPS Types.
- **Instance Definition Reference** The assignment of the Instance Definition to a root Resource Item, which results in a set of one or more Resource Items to be run.
- **Instance Definition Item** The item on the abstraction layer on top of existing RPS Types.
- **Instance Definition Node** A node item that will result in an association of a node with target items that are created when an Instance Definition is invoked.

Creating, Getting, Setting, and Removing an Instance Definition Item

Before activating any software, you must establish your working session. To do so, execute the following PowerShell Command in PowerShell ISE Administrator Mode.

IMPORTANT

Start by establishing your working session in PowerShell ISE Administrator Mode.

1. Click on the **Search Icon** from the Start Menu.

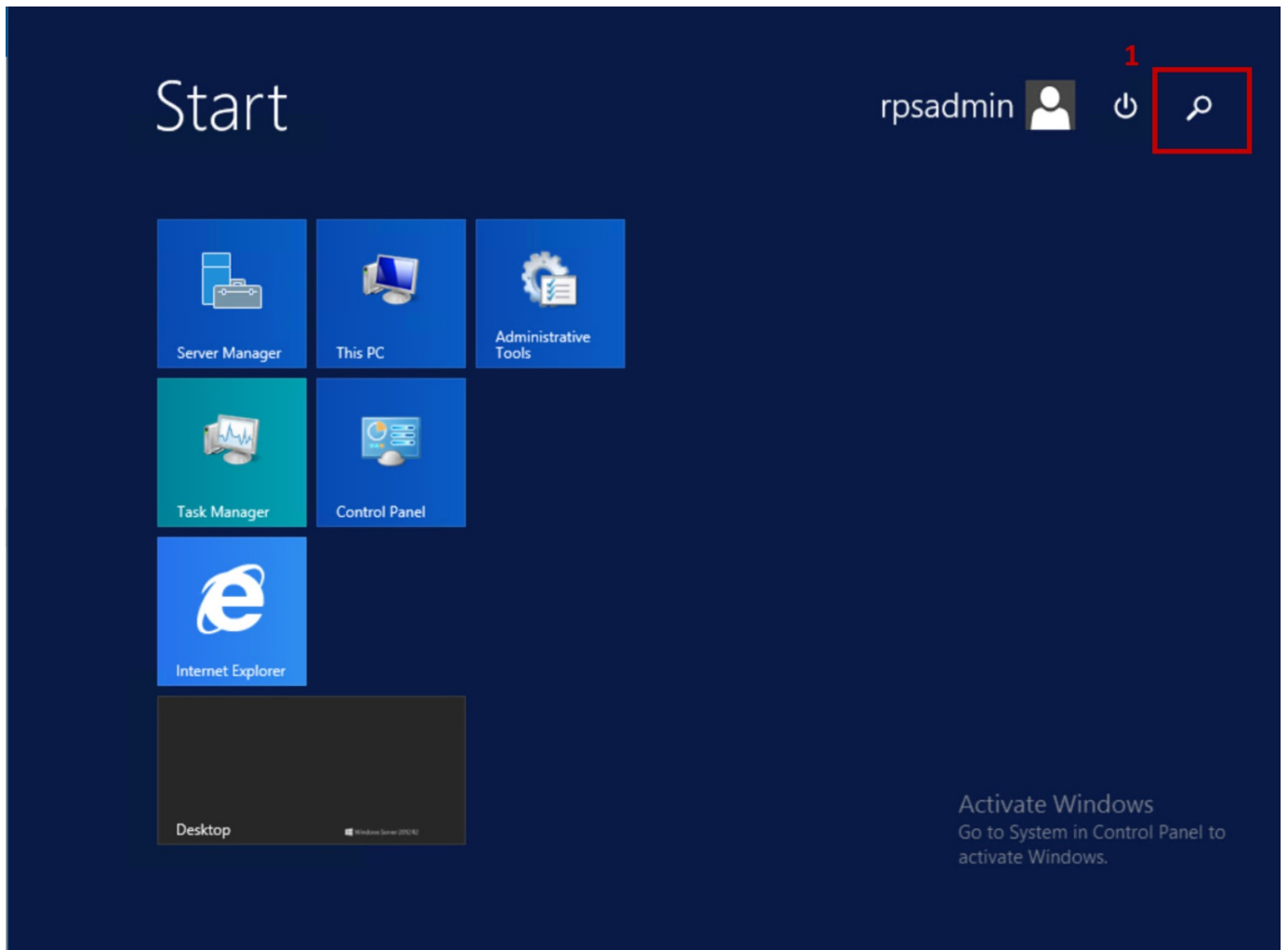


Figure 1: Click on Search Icon.

2. Search for PowerShell ISE by typing **PowerShell ISE** in the Search bar.

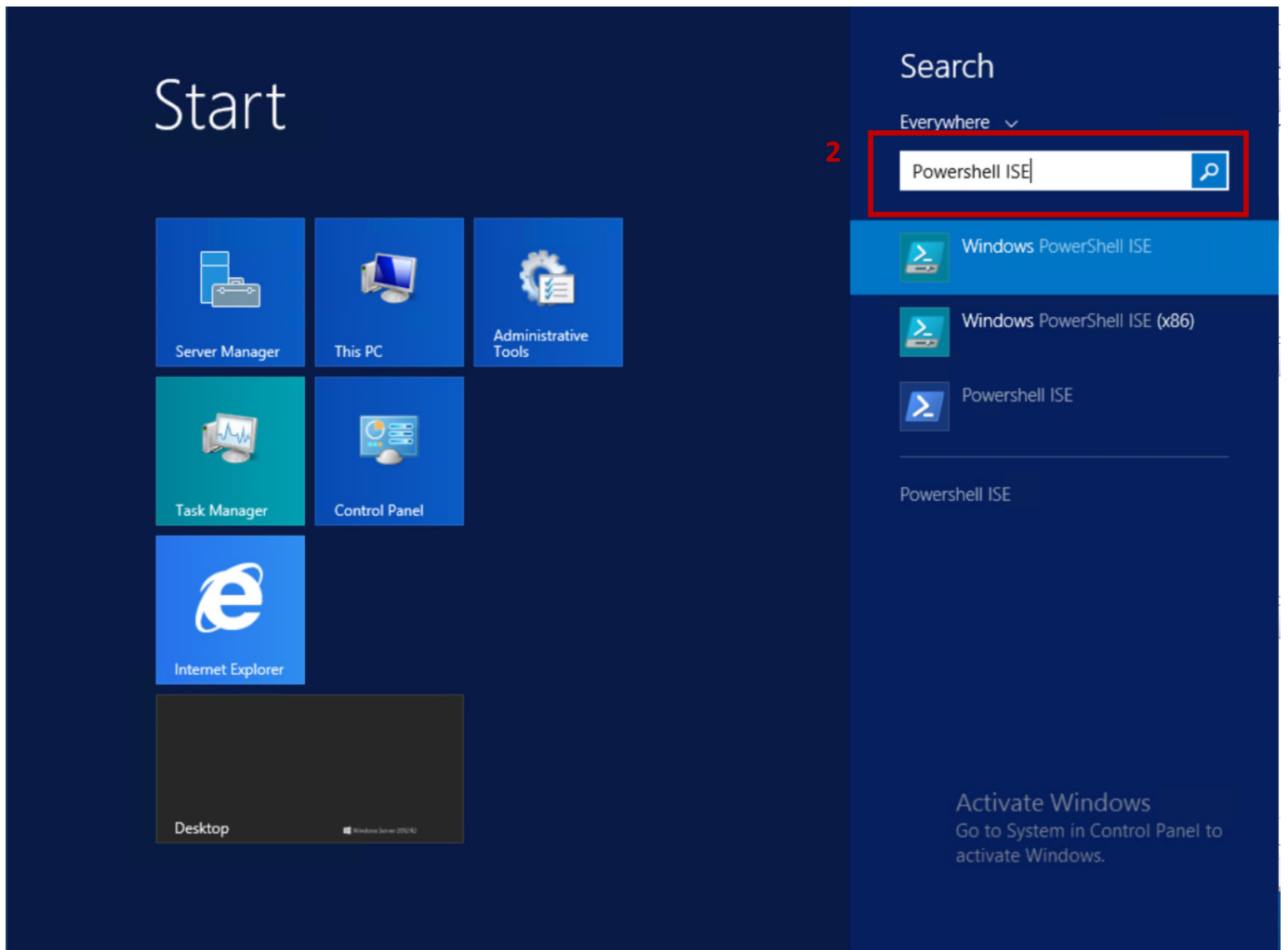


Figure 2: Search for PowerShell ISE.

3. Click on **Run as administrator**.

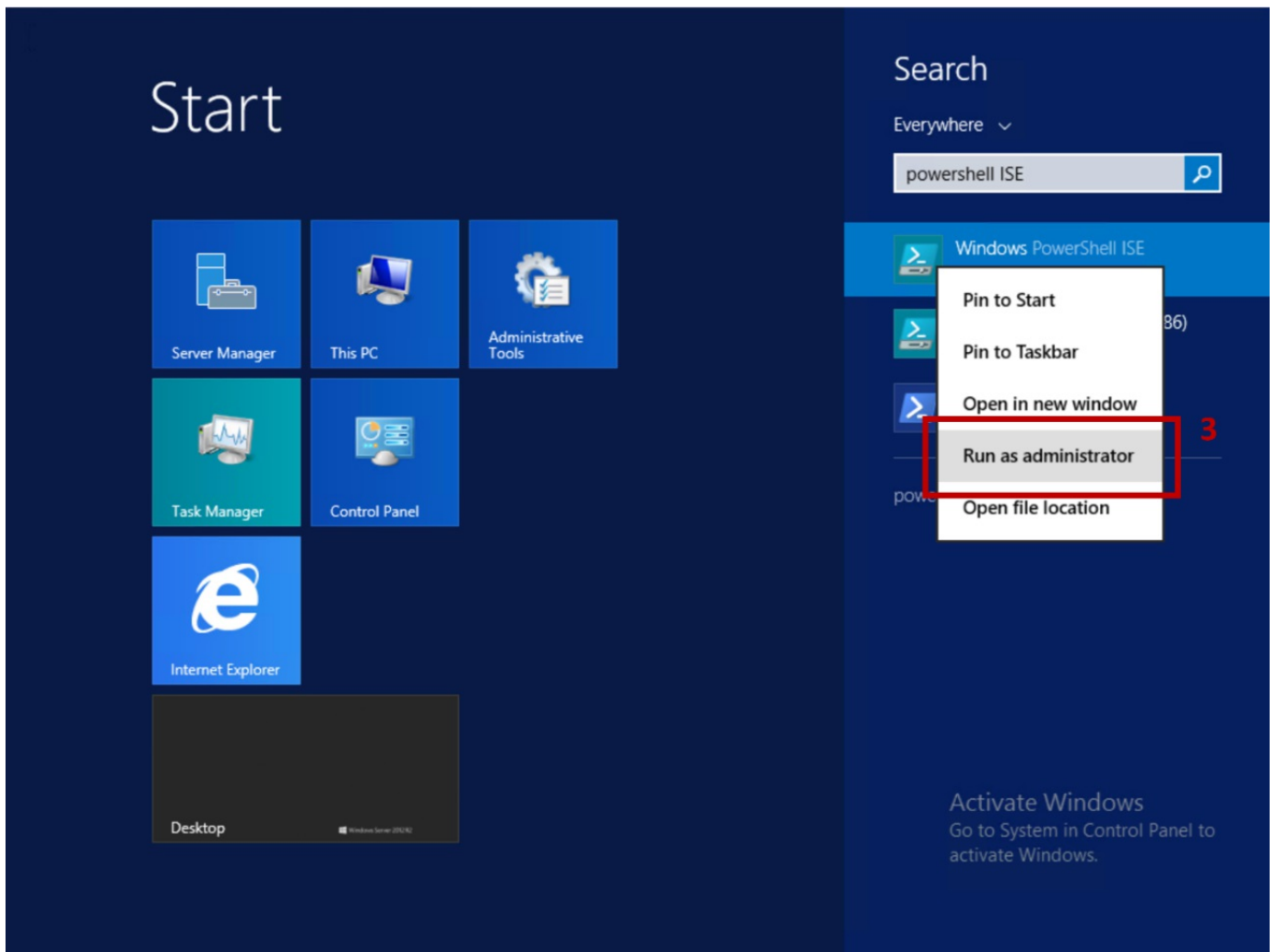


Figure 3: Open PowerShell ISE as Administrator.

Creating an Instance Definition Item

Create new Instance Definition Item by Type Definition ID.

```
New-RpsInstanceDefinitionItem -EntityName $testEntityName -Properties $prop -TypeDefinitionId $typedefId
```

Create new Instance Definition Item by Type Definition object.

```
New-RpsInstanceDefinitionItem -EntityName $testEntityName -Properties $prop -TypeDefinition $typedef
```

Getting an Instance Definition Item

Find an Instance Definition Item by Type Definition ID.

```
Get-RpsInstanceDefinitionItem -Id $lookupId
```

Setting an Instance Definition Item

Updates Instance Definition Item. If it doesn't exist, it will create a New Instance Definition Item.

```
Set-RpsInstanceDefinitionItem -Name $Name1 -TypeDefinitionId $id -Properties @{Prop1 = "Value1"} -EntityName $entityName
```

Removing an Instance Definition Item

Delete Instance Definition by ID.

```
Remove-RpsInstanceDefinitionItem -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"
Remove-RpsInstanceDefinitionItem -InstanceDefinitionItem $RPSInstanceDefinitionItem
```

More Resources

- [RPS Instance Definition](#)
- [RPS Instance Definition Node](#)

RPS Instance Definition Node

Last updated on June 28, 2021.

Last Reviewed and Approved on PENDING REVIEW

Intended Audience

This document is intended for use by a developer.

Introduction

RPS Instance Definition Nodes are definitions for nodes that can be created and associated with Instance Definitions.

Terms and Definitions

- **Instance Definition** A layer on top of existing RPS Types that translates a high-level request into the low-level commands required to perform an operation.
- **Instance Definition Reference** The assignment of the Instance Definition to a root Resource Item, which results in a set of one or more Resource Items to be run.
- **Instance Definition Item** The item on the Instance Definition layer on top of existing RPS Types.
- **Instance Definition Node** A node item that will result in an association of a node with target items that are created when an Instance Definition is invoked.

Creating, Getting, Setting, and Removing an Instance Definition Node

Before activating any software, you must establish your working session. To do so, execute the following PowerShell Command in PowerShell ISE Administrator Mode.

IMPORTANT

Start by establishing your working session in PowerShell ISE Administrator Mode.

1. Click on the **Search Icon** from the Start Menu.

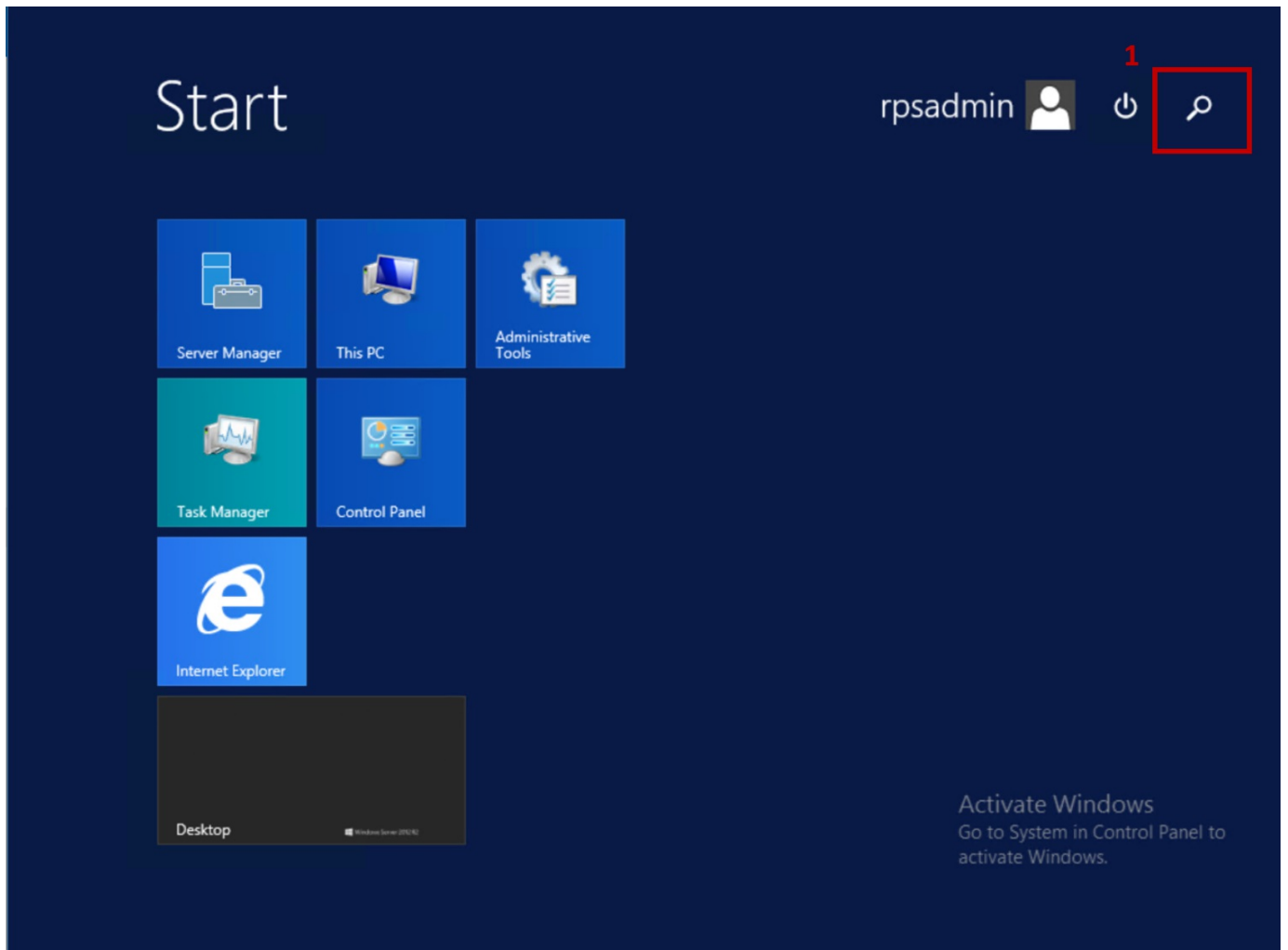


Figure 1: Click on Search Icon.

2. Search for PowerShell ISE by typing **PowerShell ISE** in the Search bar.

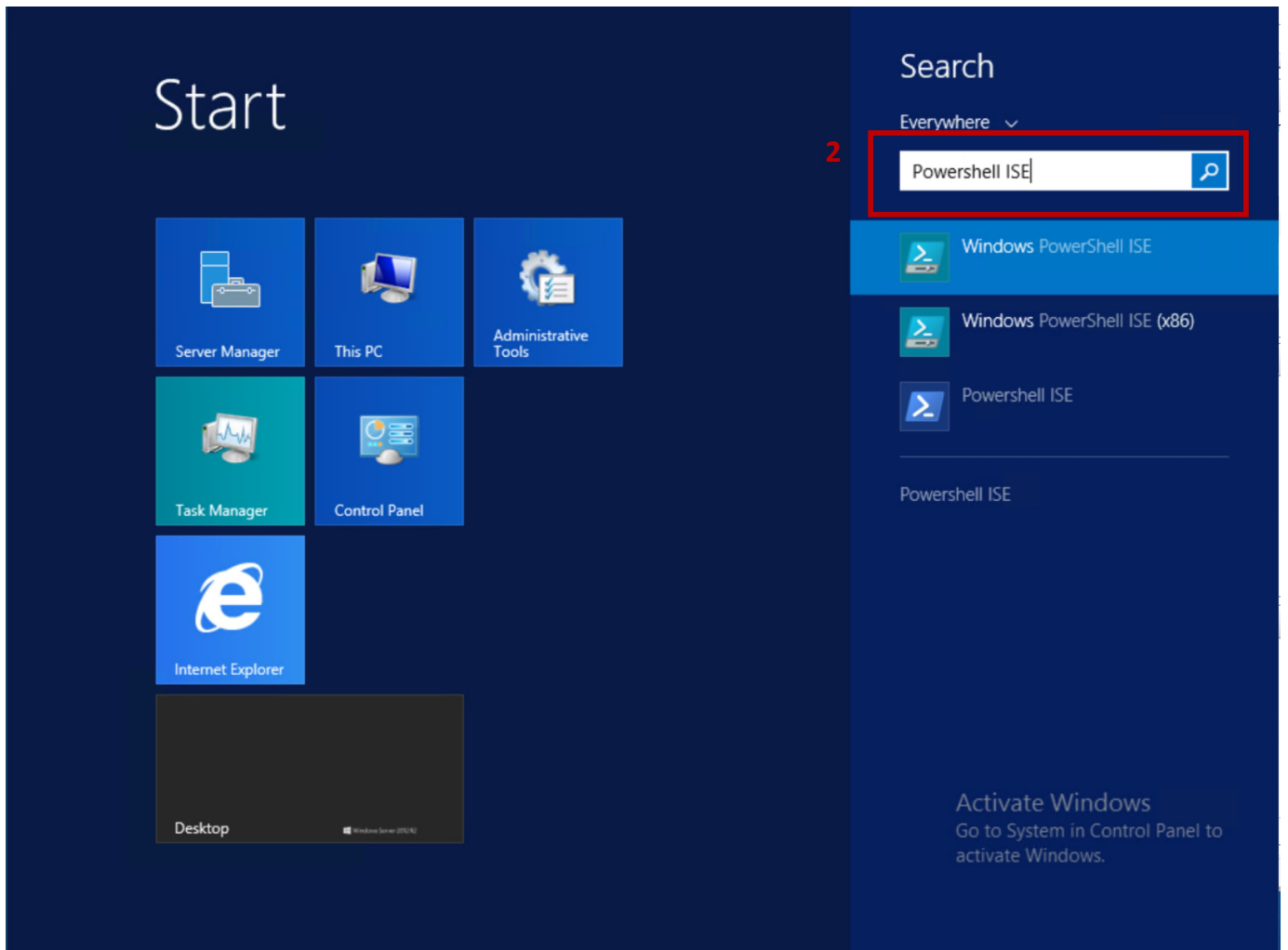


Figure 2: Search for PowerShell ISE.

3. Click on **Run as administrator**.

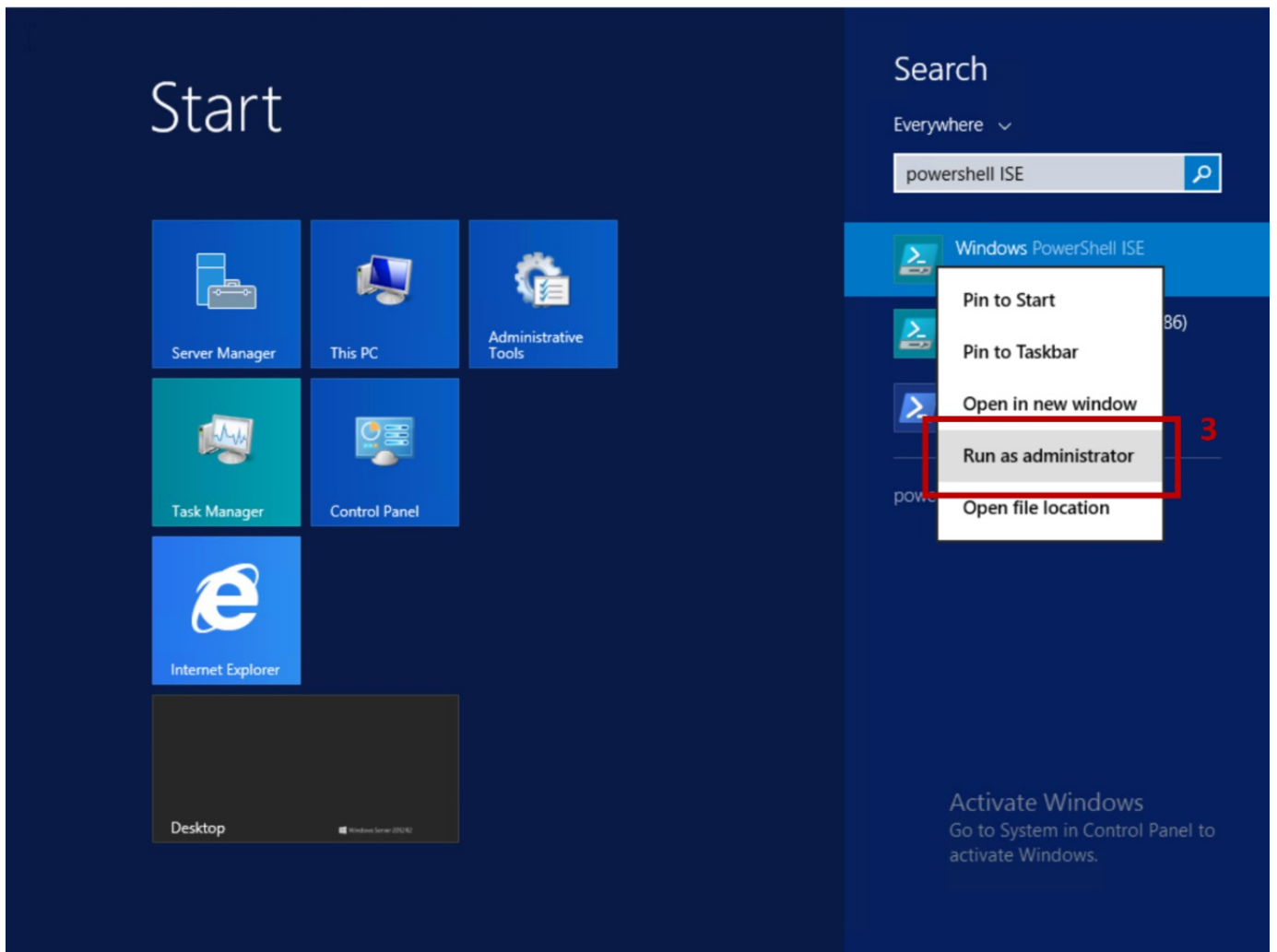


Figure 3: Open PowerShell ISE as Administrator.

Creating an Instance Definition Node

Create new Instance Definition Node with only the required parameters.

```
New-RpsInstanceDefinitionNode -EntityName ConcreteName1 -Name Name1 -HostName microsoft.com -IPAddress 10.10.10.10
```

Create new Instance Definition Node with all parameters.

```
New-RpsInstanceDefinitionNode -EntityName ConcreteName1 -Name Name1 -HostName microsoft.com -IPAddress 10.10.10.10 -SyncEndpointUrl microsoft.com/sync -CertificateThumbprint xyz123 -PollingInterval 500
```

Create new Instance Definition Node with additional properties.

```
New-RpsInstanceDefinitionNode -EntityName ConcreteName1 -Name Name1 -HostName microsoft.com -IPAddress 10.10.10.10 -Properties @{Property1 = 'Value1'}
```

Getting an Instance Definition Node

Find an Instance Definition Node by ID.

```
Get-RpsInstanceDefinitionNode -Id $lookupId
```

Find an Instance Definition Node by name.

```
Get-RpsInstanceDefinitionNode -Name $lookupName
```

Setting an Instance Definition Node

Updates Instance Definition Node. If it doesn't exist, it will create a new Instance Definition Node.

```
Set-RpsInstanceDefinitionNode
```

Removing an Instance Definition Node

Delete Instance Definition Node by ID or by object.

```
Remove-RpsInstanceDefinitionNode -Id "8825A09C-CCE3-4BB0-BCE1-03B4729AC423"  
Remove-RpsInstanceDefinitionNode -InstanceDefinitionNode $instanceDefinitionNode
```

More Resources

- [RPS Instance Definition](#)
- [RPS Instance Definition Item](#)

How to Configure Logging for the RPS API

Last updated on February 27, 2020.

Last Reviewed and Approved on PENDING REVIEW

This guide will provide links for how to configure logging for the RPS API. RPS uses Serilog for logging and log file management. RPS reads an app settings section of the app.config file within RPS to configure Serilog. The app.config file for the RPS API is located in the Source/Rps.Api folder.

- The Serilog website is located at <https://serilog.net/>
- The Serilog documentation for app settings can be found at <https://github.com/serilog/serilog-settings-appsettings>
- RPS will honor all app settings that Serilog supports by passing the app settings section to Serilog upon construction of the logger.

How to Configure RPS-Mapped Parameters

Last updated on September 23, 2021.

Document Status: Document Developer Quality Pending.

Scalar Parameters

Simple `[string]`, `[int]` and `[bool]` parameters, such as `$IPAddress` and `$OutputPath`, are supplied by RPS by convention. When RPS is publishing the Partial assigned to a target, it begins with the **ResourceAssignment** representing the assignment between the partial config and the computer it's publishing to or when Rps is deploying the Packaging Script Provider it begins with the **TargetItem** it is applying the package to.

RPS follows these steps, in order, to find a value for `$OutputPath`:

1. Look for a property on the **ResourceAssignment** named "OutputPath" (When used in a Partial Configuration.)
2. Next, look for a property on the **TargetItem** named "OutputPath"
3. Next, look for a property on the **TargetItem**'s parent named "OutputPath"
4. Traverse the parent items until reaching the root **TargetItem**
5. Finally, look to the Node for a property named "OutputPath"

NOTE

Supplying RPS data to the parameters allows an Administrator to structure data in a more natural way. For example, the OutputPath is common for the Node, so it can be specified once, instead of on every computer within the Node.

PSCredential Parameters

Nearly every partial config will require the use of credentials. To simplify usage, RPS will automatically supply a parameter of type `[PSCredential]` by finding a corresponding Credential ResourceItem in RPS.

Using the example below, RPS follows these steps to find a value for `$DomainAdmin`:

```
[Parameter(Mandatory = $true)]  
[pscredential]  
$DomainAdmin,
```

1. Look for a "Credential" ResourceItem assigned to the Target computer with a "Role" matching the parameter's name, "DomainAdmin"
2. Next, look for a "Credential" ResourceItem assigned to the Target computer with a "ResourceState" matching the parameter's name

Hashtable Parameters

`[Hashtable]` parameters allow mapping configuration items in RPS into structured data that PowerShell can easily consume. Because there are many options for supplying complex data to a parameter, a parameter uses attributes to indicate to RPS how to supply the values.

The required `$DSCEncryptionCertificate` parameter is an example. RPS will find a Certificate ResourceItem assigned to the computer with a Role which includes "DSCEncryption". That Certificate info, if found, will be supplied to the partial automatically as a Hashtable.

```
[Parameter(Mandatory = $true)]
[ResourceItemMapping(EntityType = [ResourceTypes]::Certificate, Role = 'DSCEncryption')]
[Hashtable]
$DSCEncryptionCertificate
```

Entity Mappings

The `[ResourceItemMapping]` attribute indicates that this parameter is mapped to a **ResourceItem** in RPS. Supported mappings are:

ENTITY	ATTRIBUTE	EXAMPLE
ResourceItem	<code>[ResourceItemMapping]</code>	<code>[ResourceItemMapping(EntityType = [ResourceTypes]::Certificate, Role = 'DSCEncryption')]</code>
TargetItem	<code>[TargetItemItemMapping]</code>	<code>[TargetItemMapping(EntityType = "NetworkConfig")]</code>
ResourceGroup	<code>[ResourceGroupMapping]</code>	<code>[ResourceGroupMapping(EntityType = "ADGroup", IsAssigned = \$true)]</code>
ResourceAssignment	<code>[ResourceAssignmentMapping]</code>	<code>[ResourceAssignmentMapping(EntityType = "SoftwarePackage")]</code>

Mapping Properties

The following properties are used to filter the mappings:

PROPERTY	DESCRIPTION	REQUIRED	APPLIES TO
EntityType	Indicates to RPS to filter based on the item's type.	Yes	All
Role	Indicates to RPS to filter based on items that contain the specified Role.	No	ResourceItems
IsAssigned	Indicates to RPS to only consider items assigned to the target. Defaults to <code>\$true</code>	No	ResourceItems, ResourceGroups
EntityIsActive	Indicates to RPS to only consider active items. Defaults to <code>\$null</code>	No	ResourceItems, TargetItems, ResourceGroups

Hashtable Array Parameters

Similar in behavior to Hashtables, except these parameters will find zero or more matches in RPS and return them as an array of Hashtables.

Creating Dynamic Resource and Target Groups

Last updated on February 12, 2021.

Last Reviewed and Approved on PENDING REVIEW

Dynamic Groups allows you to add filter conditions to a group. Then any Resource or Target that matches the conditions will be automatically added to those groups.

Use Cases

1. Create a Target Group of Windows Machines by filtering on OsType = 'Windows'. Then you can assign a Resource Item to all Windows machines.
2. Create a Resource Group of Admin Accounts where Role Contains 'Admin'.

Steps

```
$condition1 = New-RpsGroupCondition -ConditionOperator "Eq" -Property "Type" -Value "VirtualMachine"
$condition2 = New-RpsGroupCondition -ConditionOperator "Contains" -Property "Role" -Value "RpsAdmin" -
Delimiter "|"

New-RpsResourceGroup -Name DynamicGroup1 -Type DynamicGroup -Operator "And" -Condition $condition1,
$condition2
```

The first command `New-RpsGroupCondition` describes the condition. This uses the same syntax as [PowerShell Operators](#).

We support strings, booleans, integers, containments, and regex comparisons. If a property on an entity is an array, then you can supply the delimiter value to split the string into an array.

`New-RpsResourceGroup` and `New-RpsTargetGroup` have two new additional fields.

- "Conditions", which takes a list of the conditions object created above.
- "Operator", with values 'And' or 'Or'.

NOTE

The operator value 'And' says all conditions must match; 'Or' says only one condition must match.

How to Import RPS Data Into the CMDB

Last updated on August 23, 2021.

Document Status: Document Developer Quality Complete.

Introduction

RPS provides the ability to export various data to an XML file, which can then be imported. To see more information on how to export RPS data and which kinds of data can be exported, see [How to Export the CMDB](#).

RPS data can be imported using the following two methods: using the [RPS GUI](#) or using [PowerShell](#).

Import RPS Data Using the RPS GUI

1. From the RPS header, select **Admin** → **Import**.

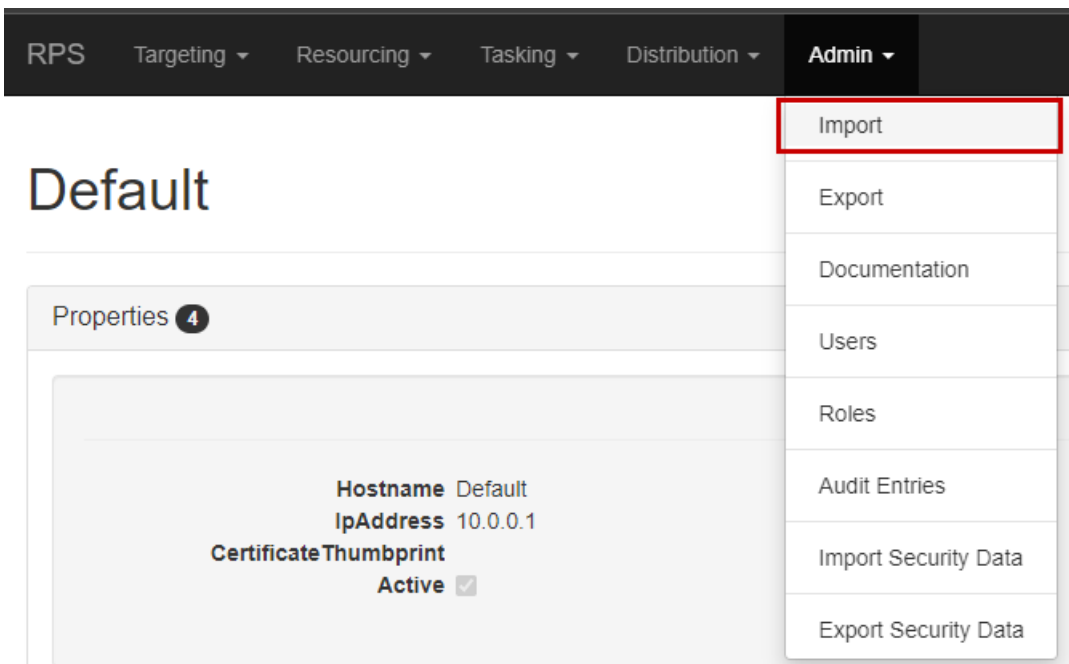


Figure 1: Navigating to the RPS Import screen.

This will take you to the "Upload Import File" view.

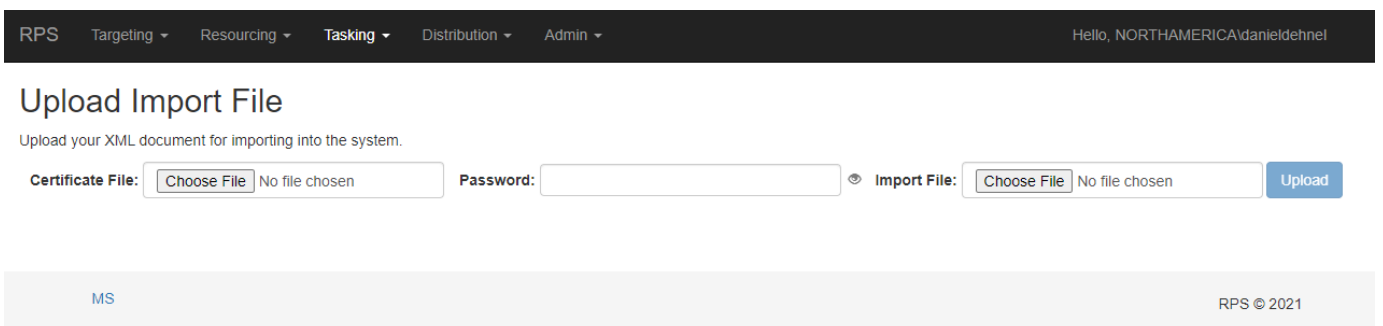


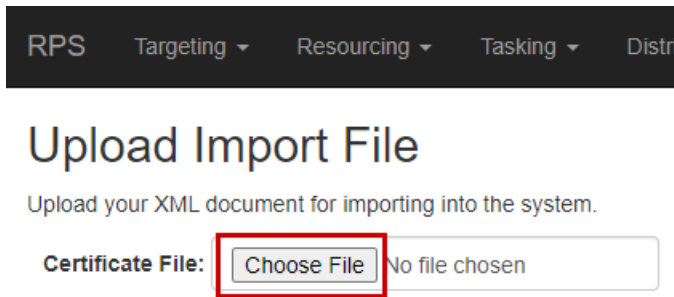
Figure 2: The RPS Import screen.

NOTE

RPS supports importing both encrypted and unencrypted files. In order to import encrypted files, the encrypting certificate and password must be provided alongside the encrypted data file. If importing an unencrypted file, only the data file needs to be provided.

For encrypted files, proceed to Step 2. For unencrypted files, skip ahead to Step 5.

2. If importing an unencrypted file, skip to Step 4. If importing an encrypted file, click the **Choose File** button next to "Certificate File". Select the certificate needed to decrypt the encrypted data file.



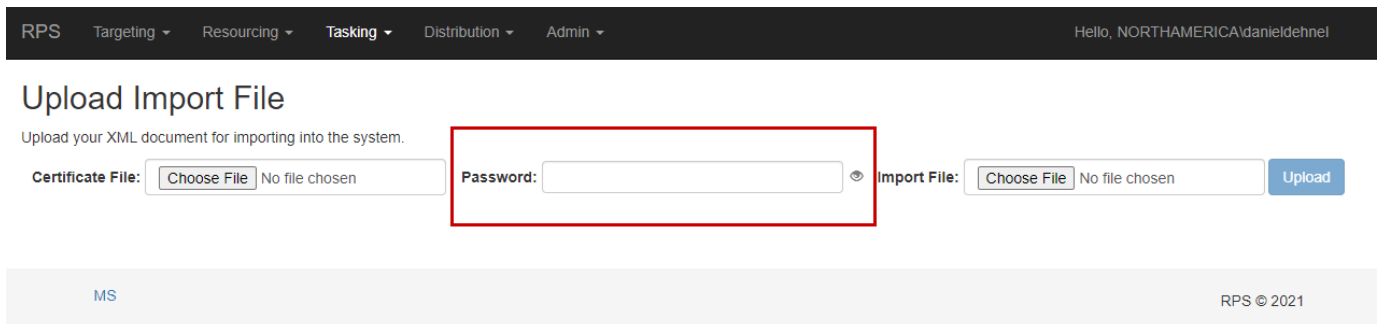
The screenshot shows the top navigation bar with 'RPS' and dropdown menus for 'Targeting', 'Resourcing', 'Tasking', and 'Distribution'. Below the navigation is the heading 'Upload Import File' and the instruction 'Upload your XML document for importing into the system.' The 'Certificate File:' label is followed by a 'Choose File' button and the text 'No file chosen'. A red rectangular box highlights the 'Choose File' button.

Figure 3: Select the certificate file.

IMPORTANT

Only .pfx files are supported for certificate file upload. Attempting to upload a different file type will result in an error message on upload (Step 5).

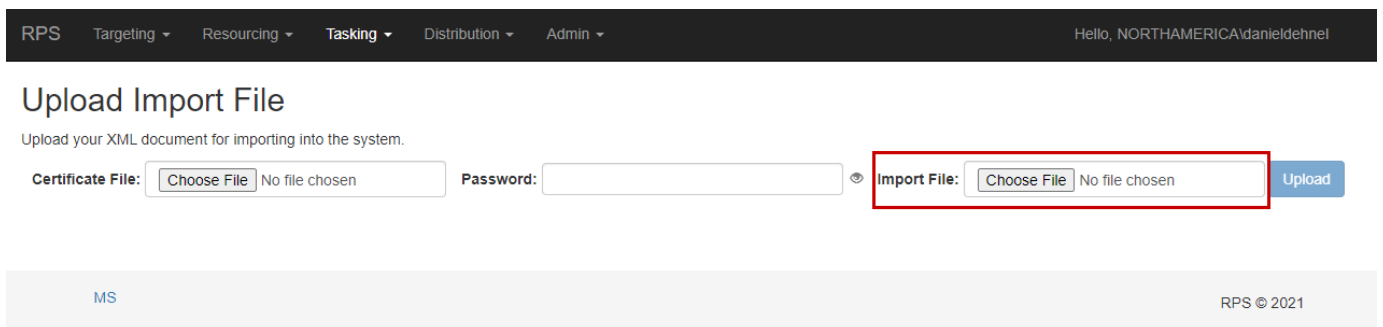
3. If importing an unencrypted file, skip to Step 4. If importing an encrypted file, the password must also be provided. Type in the password in the **Password** field. The eye icon to the right of the field will toggle showing/hiding the password characters.



The screenshot shows the top navigation bar with 'RPS' and dropdown menus for 'Targeting', 'Resourcing', 'Tasking', 'Distribution', and 'Admin'. The user is logged in as 'Hello, NORTHAMERICA\danieldehnel'. Below the navigation is the heading 'Upload Import File' and the instruction 'Upload your XML document for importing into the system.' The 'Certificate File:' field has a 'Choose File' button and 'No file chosen'. The 'Password:' field is highlighted with a red box and contains an eye icon to its right. The 'Import File:' field has a 'Choose File' button and 'No file chosen'. An 'Upload' button is located to the right of the 'Import File:' field. At the bottom, there is a footer with 'MS' on the left and 'RPS © 2021' on the right.

Figure 4: Type in the password.

4. Click the **Choose File** button next to "Import File" and select the data file to import.



The screenshot shows the top navigation bar with 'RPS' and dropdown menus for 'Targeting', 'Resourcing', 'Tasking', 'Distribution', and 'Admin'. The user is logged in as 'Hello, NORTHAMERICA\danieldehnel'. Below the navigation is the heading 'Upload Import File' and the instruction 'Upload your XML document for importing into the system.' The 'Certificate File:' field has a 'Choose File' button and 'No file chosen'. The 'Password:' field is empty and has an eye icon to its right. The 'Import File:' field is highlighted with a red box and has a 'Choose File' button and 'No file chosen'. An 'Upload' button is located to the right of the 'Import File:' field. At the bottom, there is a footer with 'MS' on the left and 'RPS © 2021' on the right.

Figure 5: Select the data file to import.

IMPORTANT

Only .xml files are supported for import data file upload. Attempting to upload a different file type will result in an error message on upload (Step 5).

The **Upload** button should now be enabled.

5. Once the Import File and Certificate/Password (if encrypted) have been entered, click **Upload**.

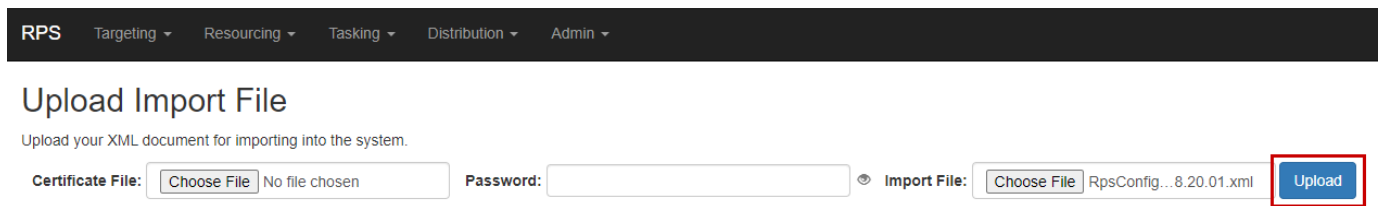


Figure 6: Click **Upload**.

If no errors display, proceed to Step 6. Otherwise, go to the [troubleshooting](#) section below for support.

6. If no errors are encountered, the "Data To Be Imported" view will display. The following data can be selected to import into the CMDB: Root Node(s), Target Item(s), Target Group(s), Resource Item(s), Resource Group(s), Task Map(s), and Task Item(s).

Alternatively, if all items in the data file should be imported, check the **Select All** checkbox at the top of the view to select all items in the data file for import.

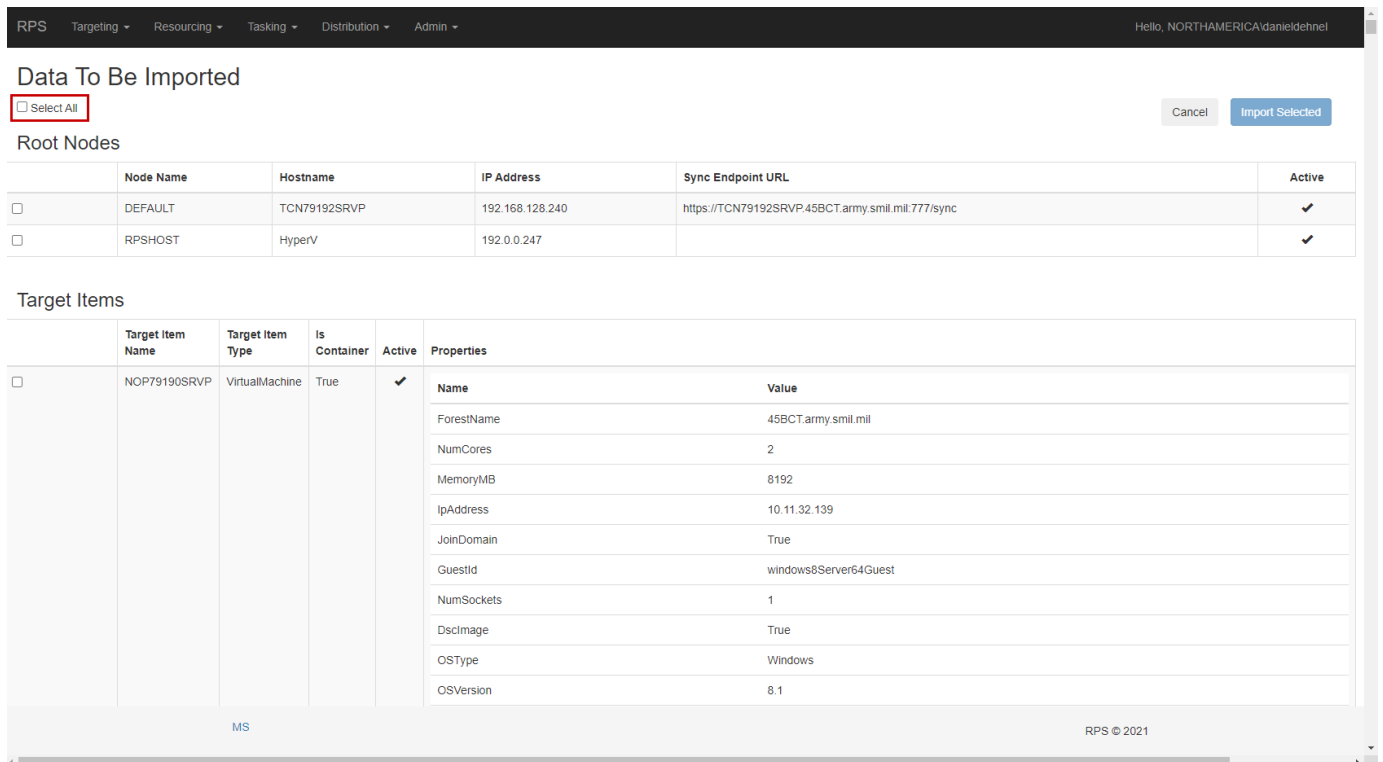
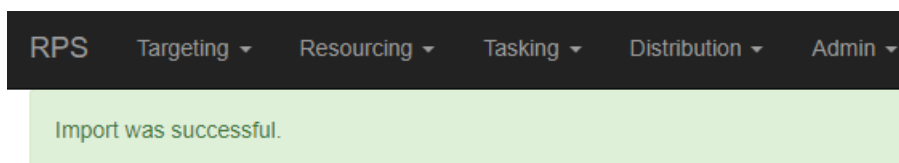


Figure 7: Select all, or choose individual items to import.

Once at least one item is selected for import, the **Import Selected** button will be enabled.

7. Once everything to import is selected, click the **Import Selected** button to begin the import process. Upon completion, the message "Import was successful." will display.



[Return Home](#)

Figure 8: Import was successful.

RPS GUI Import Troubleshooting

Invalid File Type

The screenshot shows the RPS GUI navigation bar with 'RPS', 'Targeting', 'Resourcing', 'Tasking', 'Distribution', and 'Admin' menus. The main heading is 'Upload Import File' with the instruction 'Upload your XML document for importing into the system.' Below this are three input fields: 'Certificate File:' with a 'Choose File' button and 'No file chosen' text; 'Password:' with an empty text box; and 'Import File:' with a 'Choose File' button and 'No file chosen' text. An 'Upload' button is to the right. A red error message box states: 'Invalid File Type! This is not a supported import type. You must upload an XML document for the import file and a .pfx file for the Certificate.'

Figure 9: "Invalid File Type!" error message.

- Only .pfx files are supported for certificate file upload (see Step 2).
- Only .xml files are supported for import data file upload (see Step 4).

The XML Was Not Recognized

The screenshot shows the RPS GUI navigation bar. The main heading is 'Upload Import File' with the instruction 'Upload your XML document for importing into the system.' Below this are three input fields: 'Certificate File:' with a 'Choose File' button and 'No file chosen' text; 'Password:' with an empty text box; and 'Import File:' with a 'Choose File' button and 'No file chosen' text. An 'Upload' button is to the right. A red error message box states: 'There were error(s):' followed by a bullet point: 'The xml was not recognized. Please check if the data was encrypted with a certificate.'

Figure 10: "The xml was not recognized" error message.

There were errors parsing the XML file. Possible reasons are:

- The file is not valid.
- The data file is encrypted, and the correct certificate was not provided (see Step 2).

A Password is Required

The screenshot shows the RPS GUI navigation bar. The main heading is 'Upload Import File' with the instruction 'Upload your XML document for importing into the system.' Below this are three input fields: 'Certificate File:' with a 'Choose File' button and 'No file chosen' text; 'Password:' with an empty text box; and 'Import File:' with a 'Choose File' button and 'No file chosen' text. An 'Upload' button is to the right. A red error message box states: 'There were error(s):' followed by a bullet point: 'A password is required if you specify a Certificate.'

Figure 11: "A password is required" error message.

If a certificate is provided, then the password must also be provided (see Step 3).

Import RPS Data Using PowerShell

Importing via PowerShell is accomplished with the `Import-RpsData` cmdlet.

Like importing using the RPS GUI, both [unencrypted](#) and [encrypted](#) data files can be imported using PowerShell. In addition,

`Import-RpsData` can also [import text](#), such as XML content generated from XML conversion cmdlets.

NOTE

The values used for the parameters in the example code snippets throughout this section are for example only, and should be changed to real values when used.

How to Import an Unencrypted Data File

In order to import an exported XML data file, the `-Path` parameter must be passed.

```
Import-RpsData -Path "C:\Users\currentuser\Documents\rps-export-test-node.xml"
```

See *Figure 12* below as an example for importing an unencrypted data file containing a single node.

```
PS C:\Users\danieldehne1\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Get-RpsNode -Name "TestNode"
WARNING: No object matching parameters was found.

PS C:\Users\danieldehne1\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehne1\OneDrive - Microsoft\Documents\rps-export-test-node.xml"

HasAny          : True
HasErrors       : False
LocalNodeId     :
LogItems       : {}
Nodes          : {TestNode}
ResourceGroups : {}
ResourceItems  : {}
TargetGroups   : {}
TargetItems    : {}
TaskItems      : {}
TaskMaps       : {}
v1TaskAssignments : {}

PS C:\Users\danieldehne1\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Get-RpsNode -Name "TestNode"

ParentNodeLazy : Value is not created.
ChildNodesLazy : Value is not created.
Id             : 314e6caa-9962-497e-b847-c86fa32dace2
ParentId      :
ParentNode    :
Name         : TestNode
Hostname     : Test
IpAddress    : 1.1.1.1
SyncEndpointUrl :
SyncReceivedVersion :
SyncSentVersion :
SyncDateUtc  :
SyncExpirationDuration :
CertificateThumbprint :
PollingInterval :
IsActive     : True
ChildNodes   : {}
CreatedUser  : NORTHAMERICA\danieldehne1
CreatedDateUtc : 8/23/2021 2:50:48 PM
UpdatedUser  : NORTHAMERICA\danieldehne1
UpdatedDateUtc : 8/23/2021 2:50:48 PM
CurrentVersion : 481379
ModifiedDateUTC : 1/1/0001 12:00:00 AM
CreatedVersion :
```

Figure 12: Importing an unencrypted data file using PowerShell.

Refer to [Import-RpsData Output](#) at the end of this article to learn more about the returned results.

How to Import XML Data Contained Within PowerShell

`Import-RpsData` can import straight text, such as XML content generated from XML conversion cmdlets, using the `-Text` parameter.

```
Import-RpsData -Text $testNode
```

Here is an example snippet that is storing `RpsData` XML content to the `$testNode` variable, then importing it via the `-Text` parameter.

```

$testNode = '<?xml version="1.0" encoding="utf-16"?>
<RpsData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <LocalNodeId xsi:nil="true" />
  <LogItems />
  <Nodes>
    <Node
      Hostname="Test"
      Id="314e6caa-9962-497e-b847-c86fa32dace2"
      IPAddress="1.1.1.1"
      IsActive="true"
      Name="TestNode">
        <ChildItems />
        <LastSyncTimeUTC xsi:nil="true" />
        <PollingInterval xsi:nil="true" />
        <Properties />
        <SyncExpirationDuration xsi:nil="true" />
        <SyncVersion xsi:nil="true" />
      </Node>
    </Nodes>
  <ResourceGroups />
  <ResourceItems />
  <TargetGroups />
  <TargetItems />
  <TaskItems />
  <TaskMaps />
  <TaskAssignments />
</RpsData>'

Import-RpsData -Text $testNode

```

See *Figure 13* using the above example to import XML using the `-Text` parameter and the resulting output.

```

PS C:\Users\danie\dehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> $testNode = '<?xml version="1.0" encoding="utf-16"?>
<RpsData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <LocalNodeId
    xsi:nil="true" />
  <LogItems />
  <Nodes>
    <Node
      Hostname="Test"
      Id="314e6caa-9962-497e-b847-c86fa32dace2"
      IPAddress="1.1.1.1"
      IsActive="true"
      Name="TestNode">
        <ChildItems />
        <LastSyncTimeUTC
          xsi:nil="true" />
        <PollingInterval
          xsi:nil="true" />
        <Properties />
        <SyncExpirationDuration
          xsi:nil="true" />
        <SyncVersion
          xsi:nil="true" />
      </Node>
    </Nodes>
  <ResourceGroups />
  <ResourceItems />
  <TargetGroups />
  <TargetItems />
  <TaskItems />
  <TaskMaps />
  <TaskAssignments />
</RpsData>'

PS C:\Users\danie\dehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Text $testNode

HasAny          : True
HasErrors       : False
LocalNodeId     :
LogItems        : {}
Nodes           : {TestNode}
ResourceGroups  : {}
ResourceItems   : {}
TargetGroups    : {}
TargetItems     : {}
TaskItems       : {}
TaskMaps        : {}
v1TaskAssignments : {}

```

Figure 13: Importing XML text directly within PowerShell.

Refer to [Import-RpsData Output](#) at the end of this article to learn more about the returned results.

How to Import an Encrypted Data File

When importing an encrypted data file, the certificate must be provided, or an error will be thrown:

```
PS C:\Users\danieldehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\rps-export-test-node-encrypted.xml"
Import-RpsData : The xml was not recognized. Please check if the data was encrypted with a certificate.
At line:1 char:1
+ Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Docu ...
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Import-RpsData], XmlException
+ FullyQualifiedErrorId : System.Xml.XmlException,Rps.Api.PowerShell.ImportRpsData
```

Figure 14: Error thrown when importing an encrypted data file without a certificate.

When importing an encrypted data file, the certificate can either be passed as a [.pfx file and password](#), or the [thumbprint](#) of a locally installed certificate.

Using Certificate File and Password

In order to import an exported XML data file, the `-Path` parameter must be passed. The file path to the .pfx certificate needed to unencrypt the data file is passed using the `-Certificate` parameter, and the password with the `-Password` parameter.

IMPORTANT

The password must be passed as a SecureString object. A String password can be converted to a SecureString using the

`ConvertTo-SecureString` cmdlet.

```
Import-RpsData -Path "C:\Users\currentuser\Documents\rps-export-test-node-encrypted.xml" -Certificate
"C:\Users\currentuser\Documents\test_cert.pfx" -Password $securePassword
```

An example of importing while passing the .pfx certificate and password and the resulting output is shown below.

```
PS C:\Users\danieldehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\rps-export-test-node-encrypted.xml"
-Certificate "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\test_cert.pfx"
-Password $securePassword

HasAny          : True
HasErrors       : False
LocalNodeId    :
LogItems       : {}
Nodes          : {TestNode}
ResourceGroups : {}
ResourceItems  : {}
TargetGroups   : {}
TargetItems    : {}
TaskItems      : {}
TaskMaps       : {}
v1TaskAssignments : {}
```

Figure 15: Import an encrypted data file with .pfx certificate and password.

Refer to [Import-RpsData Output](#) at the end of this article to learn more about the returned results.

Using Certificate Thumbprint

In order to import an exported XML data file, the `-Path` parameter must be passed. The thumbprint of the installed certificate needed to unencrypt the data file is passed using the `-CertificateThumbprint` parameter.

```
Import-RpsData -Path "C:\Users\currentuser\Documents\rps-export-test-node-encrypted.xml" -
CertificateThumbprint "f53bdaaae346627bd2657bce9e5fe81d07219c9dd"
```

An example of importing an encrypted data file using the thumbprint of an installed certificate and the resulting output is shown in *Figure 16* below.

```
PS C:\Users\danieldehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\rps-export-test-node-encrypted.xml"
-CertificateThumbprint "f53bdaaae346627bd2657bce9e5fe81d07219c9dd"

HasAny          : True
HasErrors       : False
LocalNodeId    :
LogItems       : {}
Nodes          : {TestNode}
ResourceGroups : {}
ResourceItems  : {}
TargetGroups   : {}
TargetItems    : {}
TaskItems      : {}
TaskMaps       : {}
v1TaskAssignments : {}
```

Figure 16: Import an encrypted data file with installed certificate thumbprint.

Refer to [Import-RpsData Output](#) at the end of this article to learn more about the returned results.

Import-RpsData Output

The object returned by the `Import-RpsData` cmdlet contains several fields that describe the import results.

HasAny

Type: Boolean

Returns *True* if any RPS data was found and imported. Returns *False* if no data was found.

See *Figure 17* for an example where an empty import file was imported.

```
PS C:\Users\danieldehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\rps-export-empty.xml"
HasAny          : False
HasErrors       : False
LocalNodeId    :
LogItems       : {}
Nodes          : {}
ResourceGroups : {}
ResourceItems  : {}
TargetGroups   : {}
TargetItems    : {}
TaskItems      : {}
TaskMaps       : {}
v1TaskAssignments : {}
```

Figure 17: Importing an empty data file.

HasErrors

Type: Boolean

Returns *True* if any errors were encountered while importing, such as if the exported data does not contain the required fields.

See *Figure 18* for an example of an invalid import attempt.

```
PS C:\Users\danieldehnel\source\repos\Core\Source\Rps.Api.PowerShell\bin\Debug> Import-RpsData -Path "C:\Users\danieldehnel\OneDrive - Microsoft\Documents\rps-export-invalid.xml"
HasAny          : True
HasErrors       : True
LocalNodeId    :
LogItems       : {}
Nodes          : {TestNode}
ResourceGroups : {}
ResourceItems  : {}
TargetGroups   : {}
TargetItems    : {}
TaskItems      : {}
TaskMaps       : {}
v1TaskAssignments : {}
WARNING: Encountered one or more issues during import. Check the RPS logs for more information.
```

Figure 18: Importing with errors.

How to Export the CMDB

Last updated on August 19, 2021.

Document Status: Document Developer Quality Complete.

Intended Audience

This document is intended for use by the LSI or developer.

Introduction

The Rapid Provisioning System (RPS) allows users to export all the data sorted in the CMDB using both the RPS web GUI and PowerShell.

The exported data is exported in an XML or JSON format and allows a user to re-import the data at a later time or on another system that has RPS installed.

To learn more about the re-import process, see [How to Import RPS Data Into the CMDB](#).

Export the CMDB Using the RPS Web GUI

1. From the RPS header, select **Admin** → **Export**.

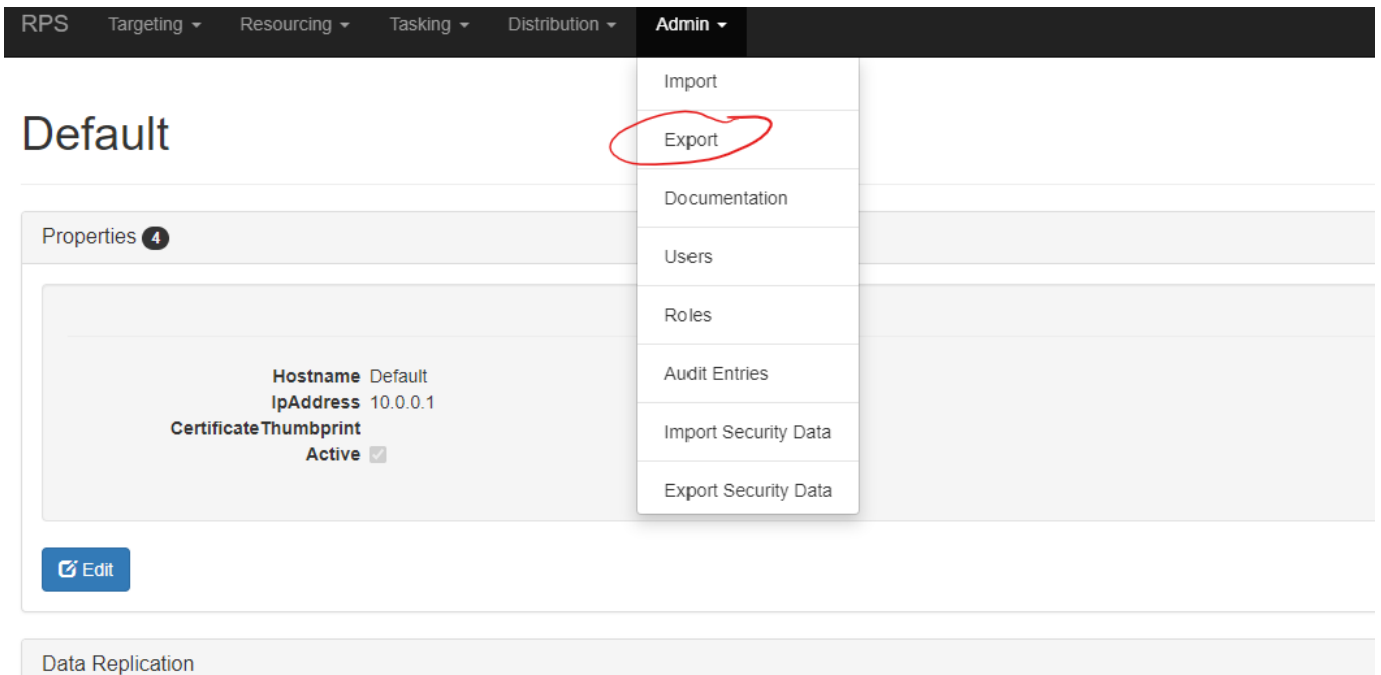


Figure 1: Navigating to the RPS Export screen.

This will take you to the "Export Data" view.

2. If the export must be encrypted, click the **Choose File** button next to "Certificate File". Select the certificate to encrypt the file with.

Export Data

You can select what data you would like to export below.

Certificate File:

Figure 2: Select the certificate file.

3. Select the data to be exported. The following individual data can be selected to export from the CMDB: Root Node(s), Target Item(s), Target Group(s), Resource Item(s), Resource Group(s), Task Map(s), and Task Item(s).

Root Nodes					
	Node Name	Hostname	IP Address	Sync Endpoint URL	Active
<input type="checkbox"/>	Default	Default	10.0.0.1		✓

Target Items					
	Target Item Name	Target Item Type	Is Container	Active	Properties
<input type="checkbox"/>	Test	Test	True	✓	

Figure 3: Select individual data to export.

Alternatively, if all data in the CMDB should be exported, check the **Select All** checkbox at the top of the view to select all items in the CMDB for export.

Export Data

You can select what data you would like to export below.

Certificate File: No file chosen

Select All

Figure 4: Select all data to export.

Once at least one item is selected for export, the **Export Selected** button will be enabled.

4. Once everything to export is selected, click the **Export Selected** button to begin the export process.

Export Data

You can select what data you would like to export below.

Certificate File: No file chosen

Figure 5: Click **Export Selected**.

NOTE

The export file will download to the default local download location configured in the current browser.

Export the CMDB Using PowerShell

1. Import the RPS API module.

```
Import-Module C:\ContentStore\Modules\Rps-Api
```

2. Run the `Export-RpsData` cmdlet.

Parameter options for the `Export-RpsData` cmdlet are:

PARAMETER NAME	TYPE	REQUIRED	DESCRIPTION
Path	String	True	Path and filename where the output file will be saved to. Ex: <code>C:\outputs\cmdbExport.xml</code>
Format	String	True	Output format for the export file. Only XML and JSON are currently supported. Default: XML.
TargetItems	TargetItem[]	False	Supply an array of target items to export.
ResourceItems	ResourceItems[]	False	Supply an array of resource items to export.
TaskMaps	TaskMap[]	False	Supply an array of task maps to export.
IncludeAll	SwitchParameter	True	When selected will export all data from the CMDB, excluding security data.
IncludeGlobal	SwitchParameter	False	Switch to export all global resources, task maps, and tasks.
NodeId	Guid	True	ID of the node to be exported. All related targets and descendant targets will be included.
SetLocal	SwitchParameter	False	Set the provided NodeId as the Local in the exported data.
SetLocalNodeId	Guid	False	Set the provided SetLocalNodeId as the Local in the exported data.
Force	SwitchParameter	False	Overwrite the previously exported file, if it exists.
CertificateThumbprint	String	False	The thumbprint of the certificate that will be used to encrypt the resulting configuration file, if provided.
Certificate	String	False	The certificate that will be used to encrypt the resulting configuration file, if provided.

Export-RpsData Usage Examples

Some common ways to use the cmdlet include:

Export all data and encrypt it with a certificate thumbprint:

```
Export-RpsData -IncludeAll -Path "c:\temp\encryptedrps.xml" -CertificateThumbprint
"2b8a73beb5da48d8ae612075428437bd677b86bb"
```

Export all data and encrypt it with a certificate:

```
$exported = Export-RpsData -IncludeAll -Path "c:\temp\encryptedrps.xml" -Certificate
"C:\ContentStore\Certificates\APP.rps.local_NodeEncryption.cer"
```

Export a specific node in JSON format:

```
Export-RpsData -IncludeAll -Format json -Path "c:\temp\rpsdata.json" -NodeId f0386465-ed03-4f60-aec4-
c9745fce0f7d
```