RMM Suite for Kaseya VSA
Operations & Customization Guide

Intelligent
Ticket Processing (ITP)

MSP Builder, LLC
Version 2.5 / Release 21-060
Glenn Barnas

Last Updated: 2021/09/29

MSP Builder RMM Suite for Kaseya VSA

The MSP Builder RMM Suite contains proprietary software, including unpublished source code. All software is (and remains) the property of MSP Builder, LLC and no transfer of ownership is granted or implied. .

The MSP Builder RMM Suite software is designed to audit, monitor and manage computers that use a Kaseya Agent application. It is not designed or configured to collect personally identifiable information and should not be configured to do so without the consent of the individual or to be used in any unlawful manner, or in a manner that requires the consent of an individual.

MSP Builder LLC ("COMPANY") CONFIDENTIAL

NOTICE: All information contained herein is and remains the property of COMPANY. The intellectual and technical concepts contained herein are proprietary to COMPANY and may be covered by U.S. and Foreign Patents, patents in process, and are protected by trade secret or copyright law.

Dissemination of this information or reproduction of this material is strictly forbidden unless prior written permission is obtained from COMPANY. Access to the source code contained herein is hereby forbidden to anyone except current COMPANY employees, managers or contractors who have executed Confidentiality and Non-disclosure agreements explicitly covering such access.

The copyright notice above does not evidence any actual or intended publication or disclosure of this source code, which includes information that is confidential and/or proprietary, and is a trade secret, of COMPANY. ANY REPRODUCTION, MODIFICATION, DISTRIBUTION, PUBLIC PERFORMANCE, OR PUBLIC DISPLAY OF OR THROUGH USE OF THIS SOURCE CODE WITHOUT THE EXPRESS WRITTEN CONSENT OF COMPANY IS STRICTLY PROHIBITED, AND IN VIOLATION OF APPLICABLE LAWS AND INTERNATIONAL TREATIES. THE RECEIPT OR POSSESSION OF THIS SOURCE CODE AND/OR RELATED INFORMATION DOES NOT CONVEY OR IMPLY ANY RIGHTS TO REPRODUCE, DISCLOSE OR DISTRIBUTE ITS CONTENTS, OR TO MANUFACTURE, USE, OR SELL ANYTHING THAT IT MAY DESCRIBE, IN WHOLE OR IN PART.

MSP Builder, LLC
385 Falmouth Ave
Elmwood Park, NJ 07407
201-300-8277

Contents

Introduction

The MSP Builder Intelligent Ticket Processing (ITP) module for Kaseya VSA provides Kaseya VSA with an intelligent method to handle alert events, filter out unwanted events, interface with various PSAs, and interface with notification services for priority event alerting. The ITP software can run directly on the VSA server or on any other server in the MSP's environment, and supports On-Prem, Hosted, and SAAS implementations of VSA.

The solution presents a very low impact on the host where it runs. The service uses an average of 0.18% of the CPU during its cycle to check for and process alerts, and the process engine will utilize less than 20% of CPU for the roughly half-second of process time typically needed per event.

Technical Overview

The ITP Module consists of three distinct components.

- A Windows System Service that monitors the VSA for new events, dispatches the processing engine for each event, and performs general housekeeping tasks. A unique feature of the service is live software updating. The service checks daily for updates, and automatically downloads the new software. The service detects the updated version and will force a service stop/start in between detection cycles to load the updated version with no operational interruption.
- An Event Processing Engine, which evaluates a single alert event (Alert, Ticket, or Service Desk Ticket), and can then rewrite the event data, determine if a remediation procedure can be run, decide if the event is worthy of having a ticket created and send a ticket request to the PSA, and then initiate a notification process when a high-priority event is detected and a ticket was generated. The EPE also performs event deduplication and detects when auto-remediation events are repeating, escalating these events appropriately. Almost all of the MSP-specific configuration is based on the Event Processing Engine and defined in the ITPConfig.ini file.

  The EPE can interface with several external modules that provide a simple means to add custom process logic, which are described below.

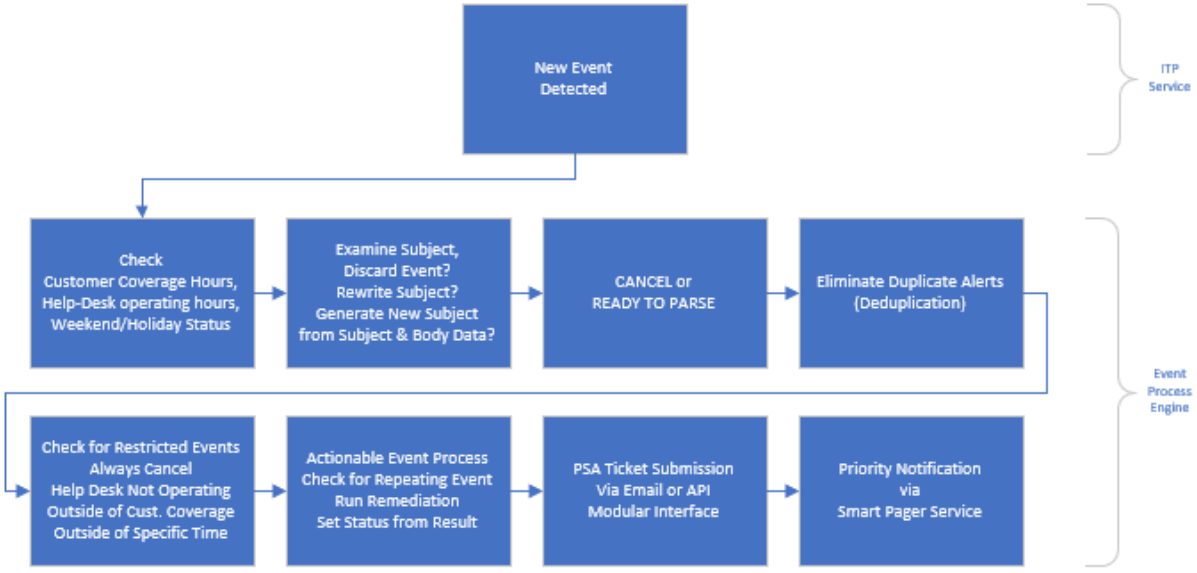
- Processing Modules that allow the EPE to adapt to custom needs without extensive code. These modules can:
  - Rewrite the subject based on content in the original subject and event body, allowing consistent parsing of event data.
  - Evaluate events and cancel or close them if the event is no longer active.
  - Summarize multiple patch failure events into a single ticket.
  - Interface with "Smart Paging" and voice notification services via email or APIs to notify on-call teams of priority events.
  - Interface with virtually any PSA via email or API.

  Additional modules can be provided by MSP Builder on request to meet specific requirements.

The ITP module does depend upon the MSP Builder Core Automation tools and requires that the Kaseya agent be installed on the host where it will run from. The KWorking folder is used to store configuration files, the ITP application and modules, and logs. The ITP software is installed to the KWorking\ITP folder and uses the standard KWorking\Logs folder to maintain its logs. Like all other MSP Builder products, logs are self-rotating on a daily basis and 7 days of log data are retained before being overwritten.

Process Flow

The ITP_Supervisor service constantly monitors the VSA for new alert events. These events can come from agent monitors and external emails. Targeting VSA for all alerts allows the ITP to process and evaluate every event. When an event is detected, the Event Process Engine is invoked and given the ID of the event that was found in VSA.



The Event Process Engine (EPE) then performs the eight general logic steps defined above for each event. The general logic concept is based on the "Lazy Admin" principal – work really hard to not do any work. This effort filters the events before they reach your help desk, eliminating or reducing the time that the staff needs to take to evaluate and then discard/close informational type of events.

1. The current day/time is compared against the customer's coverage hours, and the help desk operating days/hours. This sets the baseline for time-based filtering of events.
2. The event subject text is examined to determine if it can be safely ignored, rewritten, or whether a modular process should be loaded to more deeply evaluate the subject, body, and other event data. This can result in a subject rewrite or event discard.
3. At this point, the decision to cancel or parse the event subject data for additional processing is made. The VSA Admin can be notified by email of cancelled events.
4. The event is checked for duplicates that have been received within a custom-defined period. Duplicate events are dropped and the VSA admin can be optionally notified.
5. The event data is compared to a table in the configuration data. This allows certain types of events to be dropped (log/monitor only) or dropped only under certain time constraints.
6. If the event is defined as "actionable", the EPE verifies that the event is not in a repeating loop. If it is, the remediation is cancelled, the priority elevated, and the PSA is given an event in "Repeating" status. If the event is not repeating, the defined Agent Procedure will be invoked with any necessary arguments. The result is monitored and will result in a New or Completed status ticket being sent to the PSA. The "request" class of actionable event does not result in a PSA ticket.
7. The custom PSA interface module defined in the configuration is loaded and the event delivered to the PSA to create a new ticket.
8. If the event priority is above the set threshold, the EPE will determine if an after-hours notification is to be made. These can be immediate if the event is within the customer's coverage

hours or deferred to early morning if outside of coverage hours. This also triggers the "3rd-Party NOC Service" notification.

1-Way vs. 2-Way Ticket Management

There is much discussion in the RMM/PSA industry about 1-Way and 2-Way ticket management.

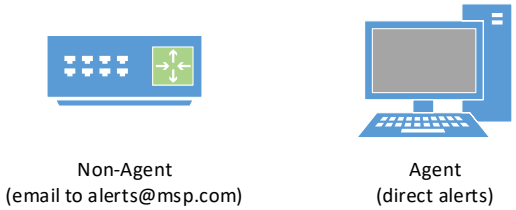
In 2-Way processing, the alert event becomes a "ticket" in the RMM *and* in the PSA. Updates to the ticket in one system are reflected in the other system. While this may seem useful, the ticketing systems built into VSA are fairly limited in comparison with "full-fledged" PSA platform capabilities. This makes synchronization of information challenging, particularly when ticket notes in one system are "public" only while on the other system they can be either "public" (visible to the customer in the invoicing) or "private" – for MSP use only.

The ITP system uses 1-Way processing, which greatly simplifies and streamlines operation. This leverages VSA as a true RMM platform. It becomes a clearinghouse for all alerts from any source. As alerts arrive, they are evaluated by ITP, and the event is closed on VSA, indicating that it has been accepted into the system. ITP actively manages the event until such time that it determines that an actual ticket must be created on the PSA. The event is then managed on the PSA from that point forward.

With the ITP being able to evaluate and cancel alerts, the need for 2-way sync is unnecessary and helps to reduce the tickets recorded in the PSA. Only events associated with automatic or manual actions become PSA tickets – notifications and other informational-type events are suppressed.

VSA / PSA Prerequisite Configuration

For integration between VSA and the PSA to work properly, the MSP must configure two distinct email accounts. Refer to the diagram below to visualize this data flow:



How Email Communications are set up between VSA and a PSA

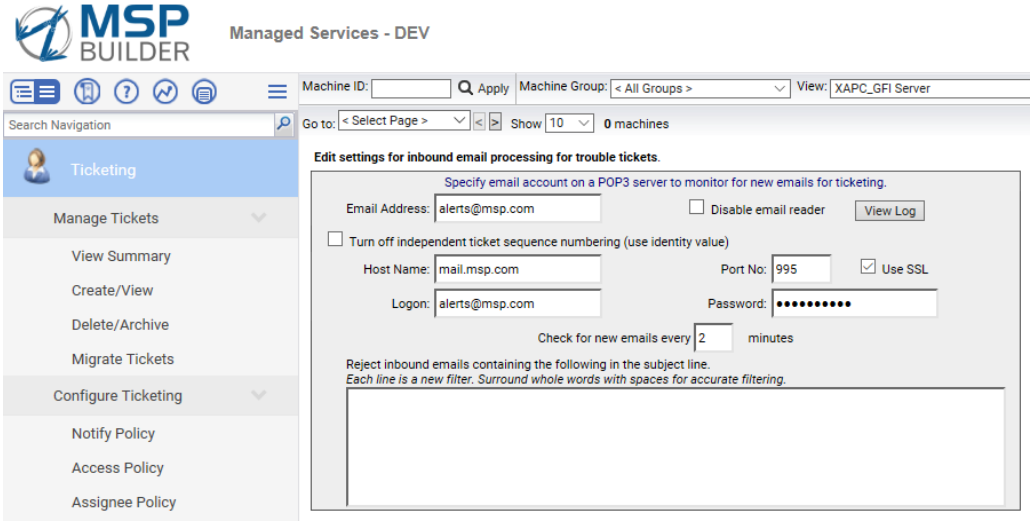
Non-Agent devices send emails to VSA. This is configured in the Ticketing component.
VSA sends emails* to the PSA. The account is defined in the System – Outbound Email section.
The PSA uses a different account to read the incoming emails and turns them into tickets.
*ITP sends these through VSA

VSA Both SENDS and RECEIVES Email.
MUST use a unique email account.
ALERTS@MSP.COM is recommended.

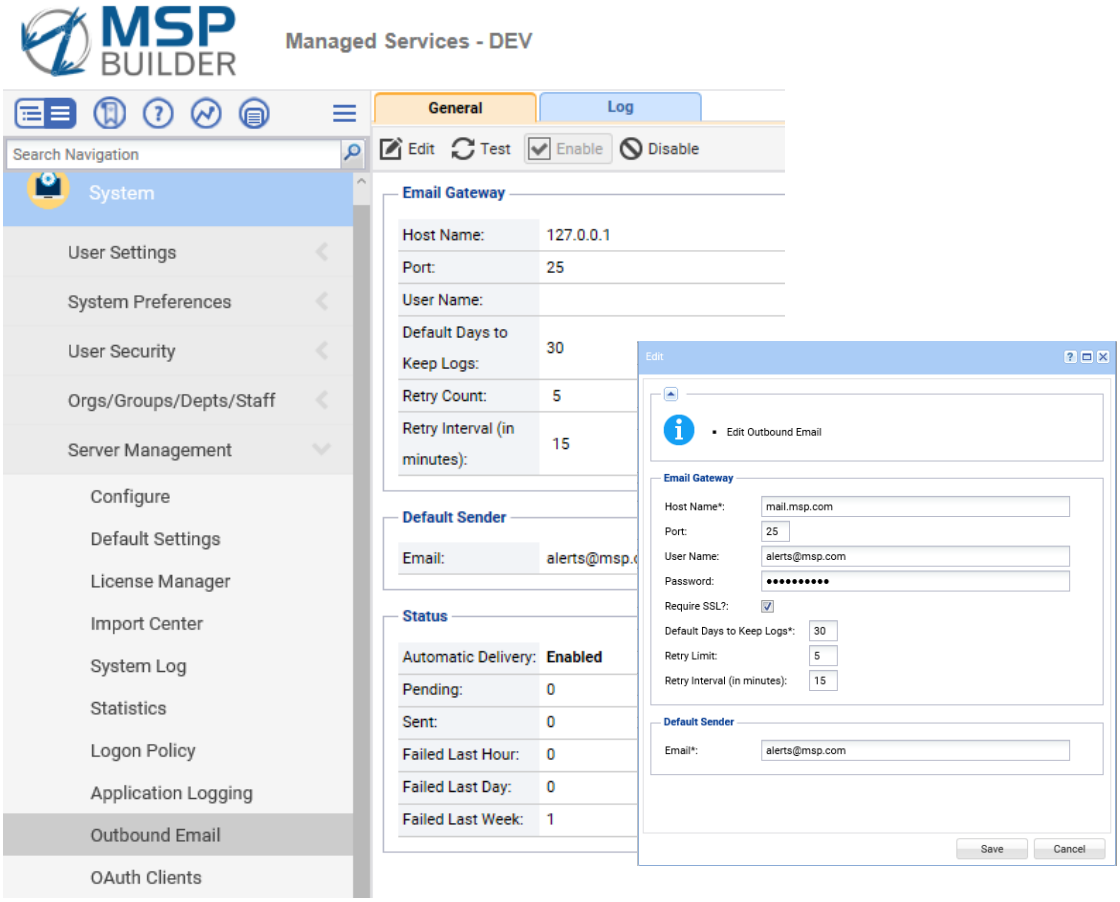
PSA only RECEIVES Email.
MUST be a unique email account.
TICKETS@MSP.COM is recommended.

VSA Email Account

The first account, which we refer to as the "alerts@" email, is used by VSA to send and receive emails. Non-agent devices that send alerts via email should use this account so that the alerts are ***received*** by VSA using the Classic Ticketing module.



Unless the VSA is using a Smart Host with Relay, the outbound relay would use the same account to ***send*** email, as shown below:



Note: The above configuration assumes use of an internal email relay server. If your VSA is on-prem and you will use Office 365, review the **How Do I?** ***Configure VSA & O-365 for Mail Relay*** document for proper configuration of mail relay services.

PSA Email Account

The second account is used by the PSA to accept incoming email ticket requests. We refer to this as the "tickets@" email. This account is often aliased to "help@" to allow end-users to submit ticket requests by email. The most important thing to verify is that there are two distinct email accounts and the accounts are not shared by the VSA and PSA. To confirm this:

- Send an email to the "alerts@" account and verify
  - It creates a ticket in VSA in Classic Ticketing.
  - It does not create a ticket in your PSA
- Send an email to the "tickets@" account and verify
  - It creates a ticket in the PSA.
  - It does not create a ticket in VSA Classic Ticketing.

While you can use any email account name that you wish, the "alerts@" and "tickets@" account names make the purpose of each account crystal clear.

Operational Overview

The ITP folder

The ITP folder contains the service (ITPSVC.BMS), the Event Processing Engine (ITPEPE.BMS), and the various modules that provide specific operational capabilities. The ITP folder contains a "Processing" subfolder where individual event processing data is stored temporarily. This allows all log data associated with a specific event to remain together until processing has completed. These logs are renamed from ".xxx" to ".log" when the processing completes. This allows the service to detect completed logs and consolidate the log data into a single ITP_Process log file.

The .MOD files are small, executable code blocks that merge into the processing engine at run-time. This allows the processing engine to remain small yet be flexible enough to accommodate any new method to interface to PSAs and notification services or interpret alert data to create a parsable event header.

The Service

The top-level of the ITP module is a Windows System Service that runs on the VSA (or another server, required for SAAS environments). This service looks for new alert and ticket events on the VSA on a regular basis, typically every 30 to 120 seconds, and up to a maximum of 300 seconds. When new events are detected, the service invokes an Event Processing Engine thread to process the event. (We call them "events" because the ITP will determine if they are worthy of becoming "alerts" and then "tickets".)

In addition to checking for new events, the service also performs some overall "housekeeping" tasks. Some tasks run daily and use both a Global and Local concept of "daily". Any task that could affect other components will run at Midnight UTC, while other tasks like log rotations and checking for updates run at Midnight local time. Log consolidation runs during each event check cycle, and the other tasks run every 5 minutes.

Configuration File

The configuration file – ITPSvcCfg.INI – is located in the KWorking folder. This is the standard configuration file location for all MSP Builder applications. The contents of the configuration file will be documented in detail in the next section.

Service Tasks

The service performs the following tasks on the defined cycle:

Category	Frequency	Task
EventScan	30-300 seconds	Query the VSA for new events
EventScan	30-300 seconds	Verify API authentication and re-authenticate if needed
EventScan	30-300 seconds	Log Consolidation
DailyUTC	Midnight UTC	API Password Change
DailyLCL	Midnight Local	Log Rotation, License Auth, LiveUpdate
Housekeeping	5 minutes	Reload configuration data if the configuration file has been modified since the last load.
Housekeeping	5 minutes	Deduplication Data cleanup – any expired data used by the deduplication logic is removed.
Housekeeping	5 minutes	Repeating Event Monitor cleanup – expired data used to detect repeating remediation events is removed.
Housekeeping	5 minutes	Notification Check – identifies when a notification has been queued and initiates the notification process. This check and notification process is Tenancy aware.

Service Logs

The service writes all of its events to logs in the KWorking\Logs folder. This is the standard log location for all MSP Builder products. The log is named "ITPSVC_#-DAY.log", where "#-DAY" is the day number (0=Sunday) and short name of the day that the events occurred.

The log writes at least one entry per scan cycle, reporting how many events were reported. When events are reported, it logs the initiation of the Event Processing Engine to handle the event. It also logs housekeeping events that were performed.

The logs are overwritten each week, so a full 7 days of data are available under normal operation. The MSP can copy the logs to another location and rename them to maintain them for a longer period if desired.

Communication Failures

The supervisor service monitors the ticket processing sub-service. If the sub-service fails to communicate with the VSA, the supervisor will detect that the sub-service ran for less than 90 seconds. When this occurs, it will defer the restart of the sub-service for 5 minutes. It will continue to do this for one hour. If the sub-service is still failing to communicate with VSA, the supervisor will increase the restart delay to 15 minutes for the next hour. If, after 2 hours, the supervisor determines that the sub-service cannot establish communication with the VSA, it will terminate. The communication issue will need to be resolved before the service is manually restarted. This can be the result of invalid credentials or a fault with the VSA (service failure, SQL failure, etc.).

The Event Process Engine

The Event Process Engine (EPE) provides the logic to process each event. A separate instance of the EPE is invoked for each event. Each event is assigned a GUID, and this value is prefixed to each log entry. The EPE obtains information about the event, agent, and customer organization directly from the VSA to help it make its processing decisions. The primary goal of the EPE is to not process the event! It employs several methods to evaluate the event and filter out unwanted or inappropriate events. These decisions can be made based on the subject or body content, the time of day, the customer where the event originated, whether the help desk is staffed, or the customer is within their coverage hours. If the event is determined to be valid, the EPE then examines the subject to insure it is a valid, parsable format. The configuration settings can map intake subject data to proper alert data or run a process module that evaluates the subject and body content to generate a proper event subject.

Improper subjects cause the event to be rejected, noting that a lookup or parsing module may be required. When a proper subject is available, the event is treated as an alert, and parsing rules then take effect. The alert data is compared to information in a remediation table. This associates a specific alert or alert class to an agent procedure that will attempt to resolve the issue. If the alert was successfully remediated, notification is disabled, the status is set to Complete, and the alert is sent to the PSA via the defined module to become a ticket.

If remediation failed to resolve the ticket, the process allows the PSA ticket to be completed with "New" status.

The priority of the alert is assessed. If it exceeds the notification threshold, then an email is sent to the on-call team as a "heads-up" of a high-priority alert. This alert can also be sent to a customer if the organization has a staff member so assigned. If a notification module is defined, it is loaded and executed to initiate a third-party notification. The system currently has modules for On-Page and Ops Genie, which offer "Smart Paging" services.

Configuration File

The configuration file – ITPConfig.INI – is located in the KWorking folder. This is the standard configuration file location for all MSP Builder applications. The contents of the configuration file will be documented in detail in the next section.

Process Engine Logs

The EPE Process Logs are consolidated by the ITP Service at the end of each process cycle. These are located in the KWorking\Logs folder, which is the standard log location for all MSP Builder products. The log is named "ITPprocess_#-DAY.log", where "#-DAY" is the day number (0=Sunday) and short name of the day that the events occurred.

Process Logs

Each EPE instance creates a distinct log file in the ITP\Processing folder. While the event is active, the log has a ".xxx" file extension. When the process completes, the file is renamed to have a ".log" extension. The ITP Service collects the data from these log files at the end of every cycle and appends them to the primary log. The interim logs are removed from this folder.

Task Detail Logs

A log containing all the data related to the event is written to the TEMP folder (usually C:\Temp). This information can be used to aid in troubleshooting or developing custom processing modules. These logs are not maintained by the ITP Service or EPE, but standard MSP Builder Maintenance tools – specifically the Temp File Cleanup utility – will remove files from this location that are older than the threshold.

Modules

Modules contain logic that is loaded dynamically into the EPE when specific conditions are met. Modules allow new features and capabilities to be added without changing the EPE software itself. The currently available modules are listed here. (Each module has an "ITP-" prefix and ".mod" file extension, not shown here.)

Module Name	**Purpose**
PSA-GEN	PSA Interface, Generic email based
PSA-*xxx*	PSA Interface - "xxx" represents the platform-specific module. Modules are available for ConnectWise Manage (CWF), Kaseya BMS, and Datto Autotask.
STS	Checks STS events, cancels workstation events, cancels server events if it can confirm the agent is online.
PFS	Patch Fail Summarization – condenses multiple patch installation failures from a single agent into a single alert/ticket.
ISR-Webroot	Intelligent Subject Rewrite for Webroot Antivirus events.
INP-OpsGenie	Incident Notification Process for the Ops-Genie Smart Paging service.
INP-OnPage	Incident Notification Process for the On-Page Smart Paging service.

Module Configuration

No specific configuration is required by most modules as the entire configuration data set is available within the module, as well as all event data values related to the Event, Agent, and Organization. The PSA Integration modules will typically utilize a parameter section in the ITPConfig.ini file to define the specific configuration settings. This will be documented in the appendix related to that specific PSA integration module.

Module Logging

Modules do not write their own logs. When a module needs to write a log event, it uses the active log file of the EPE instance that invoked it, including the current Event GUID.

Installation

Prerequisites

The ITP Service and Engine components must be installed on a server that has a Kaseya Agent and the RMM Suite Core Tools deployed. The core tool components are required for the ITP components to function. The server where the ITP components are installed can be the VSA itself (minimizes network traffic), the VSA SQL server, the PSA server (Windows-based), or any Windows 10/Server 2012 or higher platform that meets the criteria of having the VSA Agent installed and the RMM Suite tools deployed. If the ITP Service is installed on a Windows 10 platform, it must be managed and maintained like it was a server with full uptime and scheduled patching.

System Requirements

The ITP Service and Engine components consume 520 KB including all current modules and the configuration files. Processing modules range from 313 *bytes* to about 1.4 KB, so new modules added to the platform require an insignificant amount of storage. Logging can require an additional 100-150KB per day with 7 days of logs retained and trimmed automatically. If individual process logs are enabled, they consume about 1.5K each, are written to the C:\Temp folder (where the Daily Temp File Cleanup process removes them after 5-7 days). These logs are usually only used when debugging or developing a new rewrite or processing module.

The overhead of the ITP Service and Engine components is minimal as well. A server with 1 core and 1 GB of RAM will easily support the ITP requirements.

Install and Setup

On the server where the ITP components will be installed:

- Download the ITP.ZIP file and unblock the ZIP file before extracting.
- Create the ITP folder in the Kaseya "KWorking" folder (the actual folder name may be different in your environment depending on MSP configuration settings).
- Extract the contents of the Zip into the ITP folder.
- Open a command prompt, CD to the "KWorking" folder, then execute the following command:

```
Bin\rmmkse.exe ITP\itpsvc.bms --install [FORCE]
```

The remaining components will be downloaded from the MSP Builder file distribution server using HTTPS protocol, the service will be configured, and the service configuration file will be created. You will be prompted to set the cycle time, in seconds, or accept the default of 90 seconds. Values between 30 and 300 seconds can be defined. Verify that the service is installed and not running (use the XNET LIST command).

Once this is completed, the ITPconfig.ini file will need to be reviewed and configured after reviewing the section titled "ITP Event Process Engine Configuration" starting on page 11. Once the engine configuration is completed, the ITP_Supervisor service can be started.

The FORCE option is useful to upgrade an existing ITP to use the new service manager software. This will completely uninstall the service and then reinstall it – using the new executables and registry service parameters.

ITP Service Configuration

The ITP Service configuration file is a simple INI-format file located in the KWorking folder. This folder name is specific to the MSP's configuration but is referenced generically as "KWorking" in this guide.

There are three key sections in the configuration file, and the default file requires a minimal amount of configuration to define MSP-specific requirements. The configuration data is loaded at startup and then checked every 5 minutes for changes. The configuration data is reloaded if a change is detected.

GLOBAL Section

This section defines service-wide configuration parameters.

Interval

The interval between checks for new events on the VSA, in seconds. An interval between 90 and 150 seconds is recommended. The interval cannot be set below 30 or above 300 seconds. If the value is not defined, it defaults to 120 seconds. Values above 150 are commonly used for testing and if used in production could delay the delivery of alerts to the PSA. The default value of 120 seconds is recommended for most implementations.

ServiceDesks

Defines which Service Desks can be interrogated for events. This is required for implementations that use Service Desk as the MSP's PSA, and this configuration requires at least two Service Desk definitions – one for intake of alerts and the second for managing PSA tickets. If not defined, all service desks will be queried and any ticket with "New" status will be processed. The standard RMM Suite service desk name is "MSPB".

Debug

Forces the service to run in DEBUG MODE. When active, the service will not invoke the Event Processing Engine and will write additional data to the log files. This should never be used in production environments.

SuppressAlarms, SuppressTickets, SuppressSDTickets

These Boolean parameters are used to disable queries for Alarms, Tickets, and Service-Desk Tickets. They are sometimes used to troubleshoot API communication issues, but generally will allow the service to only query the type of events being generated in VSA. For standard MSP Builder monitors, these are Alarms. KNM and other external, email-based events utilize one of the ticketing platforms.

As of July 2021, Kaseya has removed the Ticketing module and ITP has been updated to no longer query the ticketing module. Only Service Desk is currently available for email-based alert notifications. Note that VSA Service Desk is not available for use with BMS.

SelfUpdate

This Boolean value allows the self-update feature to be disabled. It is normally enabled, and as new ITP versions are released, the software self-updates. The feature may be disabled when initially deploying ITP or when other platform updates have not yet been completed and you want to control the update process.

UpdateURL

These parameters allow the system to receive updates from an alternate source.
This should only be defined under the direction of MSP Builder Support staff members as changing these could affect the nightly auto-update capability.

NightlyTasks

A comma-delimited list of commands to run during the LOCAL start-of-day cycle.

LAUSER Section

The ITP Service can be used to schedule the MSP Builder Local Admin User utility to generate local admin credentials on workstations. The utility must be present in the KWorking\Bin folder and the schedule defined. When the utility is scheduled through the ITP Service, it runs at midnight of the scheduled day (local time). This utility is a standard component of the RMM Suite and no additional configuration or download action is necessary.

LAUserCycle

Defines the schedule for running the LAUser utility. This is when the procedure will be scheduled – it may not run until an offline agent checks into the VSA.

- 0 Disabled – do not schedule.
- 1 First of every month.
- 2 First and Fifteenth of every month.
- 3 Every Monday

The value can have a suffix of "!" to request an immediate run. This flag will be removed from the configuration file after the first time the LAUser account is updated. Using a value of "1!", for example, will run at midnight of the current day and then again on the first of each following month.

APIPASS

This defines when the MSP Builder API Password should be changed. The change is made at midnight UTC, not local time, and all MSP Builder tools must be configured to use the same password change configuration setting. The available options are:

- Never Never change the password via automation.
- Daily Change the password every day at 00:00 UTC.

This option is currently unavailable due to a software defect in the VSA API and is internally disabled. When this defect is resolved, this feature will be enabled in all MSP Builder components that utilize the API interface to VSA.

Special Features

The ITP service has several special features that can be enabled to perform tasks on a schedule.

- Website Monitoring and Keep-Alive
- Outdated Agent Cleanup
- Server Availability Monitoring

See the Special Features section for details on these capabilities.

ITP Event Process Engine Configuration

The ITP Event Processing Engine has a rather comprehensive set of configuration parameters that allow control over nearly every aspect of event processing. It also uses an INI-format file located in the KWorking directory. There are several required sections, and many of these can be duplicated to apply specific control over alerts by customer organization. This allows the VSA to be shared by multiple MSPs or Customer IT organizations, with alerts being processed and delivered uniquely for each organization.

GLOBAL Section

Defines the general operation of the Event Processing Engine.

DeDupPeriod & ActDeDupPeriod

The period, in minutes, where repeating events from the same agent are considered duplicates. There are no minimum or maximum values, but this should be a reasonable value based on the MSP and typical customer environment. Values between 30 minutes and 4 hours (240 minutes) are typical. A separate de-dup period can be defined for Actionable remediation events. This value should be fairly short, and 30 minutes is a recommended value.

PFSummaryPeriod

The period, in minutes, where patch install failures will be summarized for specific agents. Each agent is tracked individually. When the first patch failure for an agent arrives, it is logged and the event process waits the defined PFSummaryPeriod time. Any additional patch failures that arrive from the same agent are appended to the log and the alert is dismissed. After the timer expires, the summary of failures is written to the initial event and that event is allowed to finish processing, posting to the PSA. This value is restricted to a minimum of 15 minutes, a maximum of 90 minutes, and defaults to 30 minutes if not defined.

VsaEmail

This is simply the default email address used when sending email from the VSA. This value can be overridden in the RMM_SETTINGS section on a per-MSP basis.

VsaAdminEmail

The email address of the VSA Administrator. Notices regarding failures and event cancellations are sent to this user if it is defined. This can be a distribution list and should NOT be your ticketing system.

NotifyFailCancel

Controls sending of failure and cancellation messages to the VSA Admin. This is a BINARY value, bit 0 controls sending Failure messages, bit 1 controls sending Cancellation messages. The default value is 3, which sends both messages.

DirectPass

This parameter will pass unparsable events directly to the PSA when enabled. If disabled, these events are dropped, and the VsaAdminEmail is used to send a process failure notice. The admin is always notified, even if the event is passed unmodified to the PSA.

SDNoteIgnore

A comma-delimited list of phrases that, when found in Service Desk notes, are ignored. This helps to identify data to be contained in the message body. Service Desk adds internal notes that are not needed in the PSA ticket body.

ServerOS & WkstnsOS

List server class OS versions (2008, 2012, 2016, 2019,linux) that are eligible for after-hours alerting. This and the Workstations list are also used to aid in platform identification.

ShowSummary

A Boolean configuration setting that, when true, will write all of the event data to a log file in the TEMP folder. Primarily a debugging tool, it is useful when developing a custom module to expose all of the available data names and values.

RMM_SETTINGS Section

This section defines the MSP-specific settings. The "RMM_SETTINGS" section is required and its settings represent the primary MSP. Additional settings, with a "_{MSPID}" suffix, can provide additional settings on a per-organization basis, allowing events from one organization to be sent to a separate PSA, or a collection of organizations sent to another MSP that leases space on your VSA platform. This feature *requires* that the first Organization Custom Field be named "MSP". The data in this field, when present, will replace "{MSPID}". Any number of organizations can have this field populated, and any number of "RMM_SETTINGS_{MSPID}" corresponding sections can be defined.

Name

Defines the MSP name. This helps identify the organization served by this configuration section. If the NOC Notify service us used, this name is used to identify the MSP that initiated the event notification.

RMM_EmailFrom

The email address that is used as the default "from" address for mail sent by this configuration set.

UseUniqueFrom

When this Boolean value is true, the From email address is modified to include the Kaseya OrgID in the domain, replacing "from@msp.tld" with "from@*ORGID*.msp.tld". This is required by some PSAs to associate an event with a specific customer. This may require creating a contact for each customer in the PSA with this unique email. Each contact can have the same name – RMM Alert – as the first/last name.

UseUniqueSender

This is similar to the UseUniqueFrom option except that it changes the user instead of the domain. The email From address is modified to be "*ORGID*@msp.tld", which may be easier to configure using O-365. Not all PSAs support this method, so both formats are provided.

PSA_Type

The ID of the PSA used by this configuration.

- GEN Generic, email-based notification.
- CWF ConnectWise Manage (Full Integration)
- BMS *Kaseya BMS*
- ATK *Datto Autotask*

PSA platforms listed in *italics* are under development. New ticket delivery always uses email to eliminate the need to duplicate configuration and classification data. These PSA-specific interfaces use the appropriate APIs to update existing tickets or retrieve the ticket number of new tickets for inclusion in priority alert notifications.

PSA_Email

The email address of the PSA, used to send ticket data. Used as a backup to API integrations.

Company Match Method

The default method for the API integrations to associate Kaseya organizations to the PSA is to use the Kaseya Org Name and attempt to match it to the PSA Company Name and then the Company Identity (text ID). The company name text can usually be changed without impact to ensure a match between the platforms. If the MSP has instead matched the Kaseya Org ID to either the Company Identity or Company Name, then the PSA_Org_MatchByID should be enabled.

```
PSA_Org_MatchByID=Y/N
```

Note that not every PSA uses both Company Name and Identity, and not every PSA platform is supported by an API integration. This parameter is ignored when the generic interface module is used.

PSA_URL, PSA_AV1-9*

The URL and additional parameters used for API communication with the PSA. Must be the full API URL. These parameters are not used by the Generic interface module.

*Each PSA module will have a unique set of parameters documented in the appendix for that module.

PSA_Subject

The subject template for alerts sent to the PSA. The subject can be a mix of text, delimiters, and macros that represent event data. See **Macros** in the Special Features section of this guide for information on how these can be used.

The standard MSP Builder PSA Subject definition is:

RMM|<agent>|<estatus1>|<hostclass>|<altsubject>

The "<altsubject>" macro can be safely used in all cases. If a PSA REWRITE option is not defined, the AltSubject value will contain the original subject text, the same as "<subject>". Using "<subject>" in this definition will effectively disable any PSA REWRITE configurations. Note that the alternate subject will only replace the original subject if defined in the PSA REWRITE section.

CleanBody

A Boolean value that, when true, removes the "For more information…" data and URL from Windows Event Log alerts. There are many custom events in the MSP Builder RMM Suite and this option makes for a much cleaner event message.

Help Desk Operating Hours

WD_Start / WD_End, WE_ Start / WE_End, & HO_ Start / HO_End

- The start and end times that the help desk is operational on non-holiday weekdays.
- The start and end times that the help desk is operational on non-holiday weekends.
- The start and end times that the help desk is operational on holidays.

These values are collectively used to define when the help desk is staffed. If the help desk is staffed, then certain alert notifications are suppressed. If the help desk does not operate on weekends or holidays, the start and end times should both be set to "00:00". The end time should be 1 minute prior to the clock-time. For 5 PM, enter 16:59.

HolidaySets

A comma-delimited list of holiday sets that are observed by this support organization. Holidays are defined by month and day only, and name the holiday observed. Multiple holiday sets can be defined for standard and annual holidays, or holidays observed by other support teams without duplicating effort. This will be explained further in the HOLIDAYSET_# section.

Notify Times

WD_NotifyTime, WE_NotifyTime, & HO_NotifyTime

The time, in HH:MM format, that deferred notifications will be delivered. When a priority alert occurs outside of a customer's notification hours, a flag is set to send a notification in the morning to notify the on-call team of an overnight priority event. This allows an early start to resolve any potential issue before the customer operating hours begin.

NotifyProcess

The name of the module that performs the after-hours notification. These modules typically interface with a "Smart Paging" system such as On-Page, PagerDuty or Ops Genie, or voice notification services such as Dial My Calls. Ops Genie, PagerDuty and On-Page are supported in the core product, and additional modules are under consideration.

NotifyControl

This setting controls whether priority notifications are sent immediately or queued and sent on 5-minute cycles. A value of 1 causes queuing, 0 or not defined results in immediate sending. The queuing option consolidates multiple notification events per cycle into a single notification. This can reduce cost as well as eliminate continuous notifications from a flurry of failures.

NotifyPriority

The event priority, as defined in the MonSetID, to generate a priority notification event. Priority 1 is the highest, and NotifyPriority is usually set to 1 or 2. MSP Builder monitor sets consider Priority 2 events as potentially worthy of after-hours notification, and priority 1 as requiring after-hours notification.

NotifyOCEmail

This is the email address of the On-Call team and is usually a distribution list. This address is notified of EVERY high-priority event, controlled by the NotifyOCAlways parameter. A customer staff member can have its "Function" value set to "PriNotify" to have its email address included in this email notification.

NotifyOCAlways

A Boolean value that, when true, sends a priority notification email to the NotifyOCEmail address even when the help desk is staffed. If set to false, the email is sent only when the help desk is not staffed.

NotifyEmail

The email address of the notification service used when the service is email-based.

NotifyURL

The URL of the notification service when API-based communication is used.

NotifySubject

The subject line used for after-hours notifications. This is usually a generic message.

NotifyBody

This is the body text sent for after-hours notifications. This is usually a generic message reminding the recipient to check for specific priority event emails.

DropCodes

The last field of the Monitor Set ID (MonSetID) contains an "Action Code" that defines the actions that the ITP takes to process the event. "REQ" and "ACT" are "actionable" and result in running an Agent Procedure, while "ALM" simply generates an alarm ticket. Other codes are permitted to generate notices, particularly for testing, and it might be desirable to cancel these events. DropCodes can define a comma-delimited list of codes for which to cancel processing.

AdvanceWarn (deprecated)

An email address list representing users that should be alerted to priority events that can be remediated via procedure *when they arrive*. Normally, the ITP module will allow up to 15 minutes for automatic remediation procedures to be performed and completed. There are situations where you may want to be notified as soon as the condition is detected, even if auto-remediation is in progress. This allows you to be aware of the event if the customer calls and lets you tell them that your team is already working on it.

This setting is rarely required and is a holdover from Service-Desk based event processing. Service Desk required several minutes to process an event, while ITP can usually process an alarm-only event in a fraction of a second, and remediation-based events usually complete within 1-2 minutes. This functionality may be removed in a future release.

NOC-Notify

This is a Boolean value that enables notification of events to a third-party NOC service. This is often used when an external NOC service is used for after-hours support.

NOC_Email

The email address to send the NOC notifications to.

NOC_AlertLevel

The priority level (and above) for which alerts are sent. This is a numeric value between 1 (highest) and 5.

NOC_Restrict

A Boolean value that, when true, restricts alert notices to non-workstation devices. This would include servers and network devices.

NOC_Always

A Boolean value that, when true, always sends the email notices. When false (the default), it only sends the notices when the MSP's help desk is not staffed.

HOLIDAYSET Sections

These sections, each with a numbered suffix, identify holidays observed by the help desk.

Each section contains a list of dates in MMDD format and defines the name of the holiday. The HOLIDAYSET_0 section defines common holidays that do not change from year to year, as shown here:

```
[HOLIDAYSET_0]
0101=New Year's Day
0704=Independence Day (US)
1225=Christmas Day
```

Additional HOLIDAYSET sections can be defined for current-year holidays. These should be updated each January. The holidays defined can represent any country or organization, and any number of HOLIDAYSET sections can be created. The primary MSP should define their holidays in HOLIDAYSET_1, with additional holiday sets used for tenants as required.

PROCESSING Section

This section maps MonSetID field data to specific processing modules. The section uses a format of "Field2:Field1" (category, name) to map to a processing module. There are two standard modules defined here, and others can be created on customer request.

PAT:PatchFail	This invokes the Patch Fail Summarization process via an external module.
STS:AgentStatus	This module checks the agent class. It drops the alert if it is a workstation class. If it detects a server class system, it verifies the Last Check In time, dropping the event if the agent has checked in within the past 2 minutes.

RESTRICTED Section

This section defines restricted processing based on the MonSetID category field (field 2). This allows certain monitors to log data but never generate alerts or permit alerts only during select time ranges. This section can be made MSP-specific, but if a value is defined in the RESTRICTED section and not in the MSP-specific section, the setting in the RESTRICTED section will be used as the default. If the default restriction is not desired by the MSP, then the Class 0 setting must be defined in the MSP section to override the default restriction.

Note that all advanced processing is terminated when a restriction is matched. If full processing is desired, but suppression of PSA ticketing and notification for specific agents or machine-groups is desired, see the Blacklist section for this feature.

There are five classes of restriction available:

0 (or undefined) Allow alerts to be delivered to the PSA at all times.
1 Never allow alerts to be sent to the PSA.
2 Allow alerts to be sent only when the help desk is staffed.
3 Allow alerts to be sent only during customer coverage hours.
4 Allow alerts to be sent only during the times defined in this section (see below).

Class4_WD_Start / Class4_WD_End

The start and end times that class 4 event can send a ticket to the PSA on a weekday.

Class4_WE_Start / Class4_WE_End

The start and end times that class 4 event can send a ticket to the PSA on a weekend *or* holiday.

Example: PERF=1 *Unoptimized systems never alert for performance.*
PMON=2 *Optimized systems alert while the help desk is staffed.*

ALERT REWRITE Section

These parameters help reduce tickets by examining the subject line, dropping informational events, rewriting events into a parsable format, or even loading and running specific modules to generate parsable subject lines from data in the subject and body of the event. The parameter is a fragment of the event subject that is matched and associated with a control action and possibly an additional value. These alerts are most commonly from agentless devices that send alert notifications via email. These are received in the Ticketing (or Service Desk) module.

Format: <subject fragment>=Control Action; Optional Argument

Control Actions are one of the following:

- CANCEL – the event is informational and processing can be cancelled. No ticket will be submitted to the PSA, and no further action will be taken. A cancellation notice can be sent to the VSA administrator to document the action.
- REWRITE – the subject is replaced with the second argument. Processing is allowed to continue with the new, properly formatted subject line. This is used to directly translate certain events that can't have custom subject lines defined. ITP processing continues, including the ability to perform remediation tasks.
- PROCESS – the second argument defines a processing module that can create a custom subject and even update event parameters. This is useful for re-parsing events from Kaseya add-in modules or even external applications.

Additional process modules are developed and supplied by MSP Builder by customer request.

REMEDIATION Section

The REMEDIATION section maps MonSetID name and category field data to an agent procedure that is run to attempt to resolve the issue. Up to four Arg/Value pairs can be specified, and the standard macros can be used to inject event-specific data. Field 1 and 2, delimited with a period, form the identifier. This is followed by the full name of the procedure to invoke, then followed by up to 4 Arg,Value pairs, delimited by commas. The "Arg" must match the prompt/variable name, and the Value is what will be inserted into the variable in the procedure.

There are two types of remediation events – ACT and REQ. ACT, or Actionable Events, will generate a New (open) ticket if the remediation is not successful or a Closed ticket if remediation was successful. ACT events can also generate a Repeating ticket and prevent further remediation if the same action is taken four times in a four-hour period. This prevents the automation from continuously applying a temporary fix to a significant problem. REQ type of events are requests that simply run a procedure. These are not monitored for success or failure, and no ticket is ever sent to the PSA for these requests.

Once a remediation procedure is defined, if the procedure is renamed, the remediation table in the configuration file must be updated to reflect the new name! The procedure is invoked via it's numeric ID, but the ID is found by a procedure name lookup process.

When developing remediation procedures, the procedure should be designed specifically for remediation purposes. It must meet the following requirements:

- Input Prompts – The input prompt and the variable name that the data is saved to must be identical. In the example below, both the prompt field and the variable name field contain "DOMAIN".

```
getVariable("Prompt When Procedure is Scheduled", "DOMAIN", "DOMAIN", "All Windows Operating Systems", "Halt on Fail")
```

- The procedure must fail if the remediation task fails to run or fails to properly remediate the issue defined by the event. A simple way to force a procedure to fail is shown below – reading a file that is known not to exist and setting the "Halt on Fail" option.

```
getFile("C:\DoesNotExist.xyz", "junk.xyz", "Overwrite existing file without alert", "All Operating Systems", "Halt on Fail")
```

- The process invoked by the procedure should complete within the allowed remediation time. If the remediation procedure has not completed within the allowed remediation time, the ITP will assume failure and generate a PSA ticket. Priority alerts are not supported on workstation class systems and will always be reduced to Priority 3 events.
    - Priority 1 or 2 have a 10 minute remediation window.
    - Priority 3 and lower have a 20 minute remediation window.

BLACKLIST Section

The blacklist section allows the MSP to define specific agents or machine-groups where full ITP processing is permitted, but alerts are not sent to the PSA. This might include development systems or entire lab locations. All processes are logged, and the account specified by the VsaAdminEmail will be notified of the cancellation (if cancellation notices are enabled) just like any other dropped event.

There are two formats used by the blacklist section:

- Agent.machine.group=Boolean
  The exact agent machine-group name is specified for a precise match to a specific agent. When a match is made, the event will be blacklisted.
- Machine.group=Boolean
  The specified machine.group is compared to the agent.machine.group name. If the specified value is matched in the agent.machine.group name, the event will be blacklisted.

The "Boolean" value must evaluate to "True" for the match to be made. This allows the blacklist to be defined but disabled until ready. A "True" value is any non-zero numeric value, "T", "On", or "Y".

PSA REWRITE Section

This is a powerful capability of ITP that allows direct parsing by the PSA while still providing a meaningful text message in the subject line. The original subject line contains all the detail needed to identify an event and is consistent between all events and agents. It never contains agent-specific information. This allows the original subject to be matched in a table that re-maps the computer-parsable subject into a code and a human-readable subject.

Creating this table and the codes is up to the MSP. We do recommend the use of a 7 to 9-character alpha-numeric code followed by a semicolon (;) and then the text message. The codes should be documented to ensure that each source event subject is mapped to a unique code. Here is an example of the Disk Capacity Smart Monitor's Event 162 (low free space) alert:

```
EVT|RMM-SMARTMON|162|Application|Error|MB-SM-A.EVT.S.P3.Alm=SMD162;Low Disk Alert!;classification
```

In the RMM Suite, most alert events are unique and thus easily mapped to a code. In this example, "SMD" represents "Smart Monitor-Disk", and 162 is the Event ID. The PSA parsing should take one more detail into account – when the event is sent to the PSA, the new subject will be:

```
RMM|<Agent_Name>|N|S|SMA162;Low Disk Alert!
```

The parsing rule should match on several parameters in the subject line:

- The "`RMM|<Agent_Name>|`" should be used to delimit and identify the Agent Name.
- The "`|N|S|SMA162;`" should be used to delimit and identify the specific event. You may need multiple rules (4) to identify these as New or Completed events and distinguish between Server

and Workstation sources. The "SMA162" is the specific value used to map to classification data within the alert.

- The `Classification` should be a comma-delimited list of 3 Type,Subtype,Item event classification values. When API Integration is enabled to the PSA, these are used to automatically classify the event. For platforms that utilize just 2 levels of classification, they are submitted as "type" and "subtype:item".

Of course, this is just an example – the MSP is free to develop any mapping method that they choose. MSP Builder provides an Excel Spreadsheet to assist in configuring these mappings that references each of our standard monitors.

TICKET NOTES & INTERNAL NOTES Sections

This is a set of mapping parameters that allows the MSP to append specific text into the ticket body that is submitted to the PSA, based on the alert parameters. When an API integration module is used to communicate to the PSA, an INTERNAL NOTES section is also supported, which will work in the same way to update the Internal Notes instead of the Ticket Body/Public Notes.

Two, three, or four parameters can be used for matching to determine when to insert the text value. Many PSAs will parse the incoming ticket body for "tags", which can be used to define specific fields or even add time to the ticket.

The general format of this section is **`Name.CLS.D1.R=Text String`**.

- Name The name of the Monitor Set ID (Field 1)
- CLS The class of the Monitor Set ID (Field 2)
- D1 The Data1 Field contents. This is usually the key event identifier value.
- R The remediation result (None/Pass/Fail/Skip).

The parameters are scanned in a specific order:

1. `Name.cls.D1.R` Fully Qualified comparison. All values must match. This is usually used to provide specific event and result notes.
2. `Name.cls.D1` Only the D1 optional parameter is provided. The R value is ignored. This can provide specific event notes, but not based on results of remediation procedures.
3. `Name.cls..R` Only the R optional parameter is provided. The D1 value is ignored. Note that the empty D1 field must be identified with the two dots as shown here.
4. `Name.cls` Generic qualification based on the MonSetID data only.

The message may contain any of the supported macros that contain information about the alert, as well as any of the Agent's custom field data. See the Macros table in the ITP Special Features section later in this manual for more information.

"Text String" can contain text and Macros in the format "<name>". The available macros are listed later in this document in the Special Features section. Macros include any available agent custom field data.

A special form of "Text String" is "FILE:name.ext". This should represent a text format file located in the ITP\TXNOTES folder. This entire file will be loaded, any macro replacement will be performed, and then the content will be appended to the ticket body (or Internal Notes, where supported by the PSA API module.

Custom Modules

Custom Modules are small "applets" that load and run on demand to customize the ITP Event Processing Engine. This permits a great deal of flexibility in the capabilities of the EPE. Modules are written by MSP Builder to either add functionality or in response to specific customer requests. Since most modules are less than a dozen or so lines of code, they can be developed, tested, and delivered to a customer in 1-2 business days. Modular code is also used to interface to various PSA platforms, although these are slightly larger and more complex.

PSA Modules

Technically not a "custom" module, these do load dynamically based on the settings in the configuration file. These can provide a "generic" integration where tickets are created by email and managed by parsing rules in the PSA, or a "full" integration that creates new or updates existing tickets via the API. The full integration will "fall back" to email delivery if the API becomes unavailable for any reason. The individual PSA modules are documented in the Appendix.

Notification Modules

These modules are used to send specifically formatted messages to "smart paging" and other event notification services. These are typically used to alert the after-hours team to priority events, and automatically escalate the notices when the primary person is not available or doesn't respond within a specific time period. MSP Builder currently supports Ops-Genie and On-Page services, although most any service can be easily accommodated.

Process (PRC) Modules

Process Modules perform specific processing of the event data to evaluate the conditions surrounding the event. They can examine the subject, body content, and several other data values provided by the various alert types supported by Kaseya. A process module will not directly perform any remediation – its job is solely to evaluate the data and deliver an event subject that can be parsed further by the EPE. This, therefore, can identify conditions where remediation is possible, and the EPE will use the custom header to identify the remediation procedure that will be initiated by further processing by the EPE. Process Modules can read all event data values, any configuration value, and even have its own custom configuration data in the standard configuration file.

Examples of the standard "PRC" type modules should provide an idea of what is possible:

- **Event Summarization**
  The PRC-PFS module performs summarization of multiple patch failure events over a specific period of time. The first event to arrive is "placed on hold" for a specific period, between 30 and 90 minutes (typically). Any additional patch failure events for the same system will have the patch ID extracted and stored, and the event will be cancelled. After the hold time expires, the original event will collect all the additional failure data and add it to the original event body, delivering a summary event to the PSA.
- **Status Check**
  The PRC-STS module evaluates Agent Offline Status events. This module determines if the event is for a server or workstation. If a server, it checks the Agent Offline status and cancels the event if the server has been online during the past 2 minutes. If the agent is a workstation, the event is cancelled, optionally sending a notification email if the offline period exceeds a specific threshold, such as 60 days. This might be used to remove the agent and terminate billing.

Incident Subject Rewrite (ISR) Modules

These are a simplified form of a Process Module and are invoked earlier in the EPE evaluation process. They evaluate the existing subject line, body, and other event data and deliver a properly parsable subject line that the EPE can then evaluate. These modules are often just 1-3 lines of code and reformat the event subject data. They do not perform any advanced processing the way Process Modules do.

An example of this module is the ISR-Webroot module, which simply extracts the agent name from the subject, discards the rest of the text, and inserts the agent name into the proper data field in a parsable subject line.

Parsable Subject Lines

So just what is a "Parsable Subject Line"?

The EPE uses a "pipe-delimited" subject line consisting of six fields of data. A typical "parsable" subject line looks something like this:

```
EVT|DHCP Server|1033|SYSTEM|ERROR|MB-DHCP-S.EVT.S.P3.Alm
```

The first field – "EVT" – defines this as an Event Log class of event.

The second field identifies the "Event Source", which is usually in a single monitor set, although different but related event sources can be found in a single monitor set.

The next three fields contain additional data to describe the event. These fields are optional, and for an Event Log event, all 3 are populated with the Event ID, the Event Log where the event was found, and the type of event (notice, warning, error, or critical).

The last field is referred to as a Monitor Set ID – it explicitly defines the Kaseya Monitor Set, the type of system where it is applied (Server or Workstation), the priority, and whether the event is an Action (.act), a Request (.req), or a simple Alarm (.alm).

The EPE uses the five primary data fields and five MonSetID sub-fields to determine the type of event, criticality, and even what to do to try and remediate it. It may look "cryptic", but to the "silicon brain" in the EPE, its clear and concise.

The MonSetID also makes it easy to modify the priority or action of a monitor set. Simply change the part of the name of the monitor set that references the priority, and the new priority setting takes effect instantly. If you develop a remediation procedure for an alarm monitor, simply change the "alm" to "act" and define the procedure in the configuration file.

Maintenance Tasks

ITP does not currently require any manual maintenance tasks to be performed.

We do recommend that you check the server at least monthly, verify that the ITP*.log files have current dates, and remove any “XXX” files from the ITP\ProcessLogs folder that are more than a day old. This is uncommon but files could be present when alert processing is unexpectedly interrupted or encounters invalid data. If the logs are not current, the ITP_Supervisor service should be stopped, then restarted after waiting 30 seconds.

Special Features

ITP provides several special features that help customize ticket generation, event tracking, agent cleanup, and even custom website monitoring. These are leveraged by the built-in scheduling of the ITP service.

Data Logging

Released in Version 2.5, ITP now creates a monthly CSV-format log to record the disposition of every alarm event that it processes. This allows the MSP to easily view the statistics of events received, processed or dropped, acted upon, and tickets that were passed to the PSA.

ITP also works in conjunction with the Smart Monitors to log when a Smart Monitor has prevented an alarm by either Transient Suppression or Self Remediation. These use a specific .LOG Monitor Set action type that the ITP Service recognizes. When these alerts arrive for processing by ITP, the information is written to the local CSV file but no additional ITP processing is performed. In the past, these events were simply logged locally on the agent, but this special alarm allows these Smart Monitor features to be centrally logged and reported.

CSV Log File

The CSV log file is located in the KWorking\Logs folder along with the other logs. It uses the name format "ITP_Summary_<MON>.csv". "<MON>" is the three-letter month abbreviation. This allows up to 12 months of data to be collected and retained on the ITP server. If longer term retention is required, these files should be moved to a separate location with year-specific folders or renamed to incorporate the year into the file name. After 1 year, the files are overwritten at the start of the month. *Long-term archiving, where required, is the responsibility of the MSP to manage!*

Log File Format

The log employs a CSV format that can be directly imported into a SQL database or opened in an Excel spreadsheet. Depending on the type of event and the processing performed, some or all data fields may be defined in each record.

Telemetry

ITP writes each of the above records to our telemetry server. Customers can view telemetry data as well as information about configuring the monitors and even troubleshooting the issue. ITP also sends health status telemetry records at least daily to report operational status and configuration issues. These may result in tickets being generated in our system and your technical staff contacted to review the issue.

Agent Cleanup

One challenge many MSPs face is keeping up with assets that customers have taken offline without any notice. These agents affect both license consumption and properly determining agent counts for correct billing. The Agent Cleanup process can be enabled to run daily and remove any agent from the RMM that has not checked in for a specified number of days. The process runs at midnight local time.

Agent Cleanup utilizes a section within the ITPSvcCfg.ini file called AGENT_CLEANUP, supporting the following values:

Enabled A Boolean value that enables or disables the daily process.

Aging The value specified here represents the number of days that an agent must not check into the RMM before being removed. The action causes the agent to be removed from the RMM database – it does not uninstall the agent. If the agent is powered on at some future time, it will check in and re-register with the RMM.

Limit Limits the number of agents that can be deleted during a single nightly cycle. A negative number will effectively disable deletions! The default is 50 agent deletions per night.

Daily Task Processor

The ITP service can initiate a series of BMS (RMM Suite tool) apps to perform specialized processing each night. These tasks run as independent processes and are triggered at midnight, local time. The NightlyTasks parameter is located in the COMMON section of the ItpSvcCfg.ini file.

NightlyTasks A comma-delimited list of tasks that will be run nightly. Tasks are generally defined by MSP Builder support. All files must be in the KWorking\Bin folder.

NightlyTasks# Same as above except that it will only run on the specific day - 0=Sunday, 6=Saturday. Invokes the ATUpdate process every Sunday when the Autotask PSA is used.

ATUpdate Updates the Autotask lookup tables. Runs on Sunday.

Macros

The system supports the following macros for use in the subject and ticket message injection:

<agent>	The full agent.machine.group name.
<agentmg>	The agent machine.group without the agent name.
<agentid>	The agent GUID.
<estatus>	The event status – New, Closed, or Repeating.
<estatus1>	The first letter of the estatus values.
<rstatus>	The remediation status – None/Pass/Fail/Skip.
<rstatus1>	The first letter of the rstatus values.
<hostclass>	"W" for workstation, "S" for server, "X" for other/unknown. This value is based on membership in the "wkstns" or "servers" machine group.
<hostos>	The host operating system code – eg: "10", "2016", "OSX", or "linux".
<data#>	Data value 1, 2, or 3, from the standard MSP Builder event subject.
<eventtype>	The alarm event type, provided by VSA.
<priority>	The event priority, 1-5.
<subject>	The entire, original subject string.
<altsubject>	The alternate subject, from PSA REWRITE section.
<eventcode>	The alarm Event ID.
<eventguid>	The internal Event GUID. Can be used to correlate log events with tickets.
<CF:name>	Any agent custom field can be defined. Up to 10 custom fields are allowed, with additional custom fields specified by "<CF#:name>", where "#" is in the range of 2 through 9.

When the macro is encountered in the Source text, it is replaced with the value represented by the macro. Caution should be used when defining Custom Field macros – if the custom field is not defined for the agent or does not exist, the replacement is made with "???" to indicate no information is available.

These macros can be used in the ticket subject line, Ticket Notes, and Internal Notes.

Website Monitoring

The Website Monitoring component is an external module run by the ITP scheduler on 5-minute intervals. It performs two key tasks – Page Keep-Alive and Site Monitoring.

Page Keep-Alive processes a list of web URLs and simply makes a request to each page. The primary purpose is to keep pages in the server cache and to keep ASP.NET site in an active compiled state. This helps to keep websites responsive. Generally, every top-level menu URL for a given website will be listed. Each URL is queried, but no further checking is performed, and unresponsive pages will not generate an alarm.

Website Monitoring makes a URL query to the primary page of the website. Any response code other than 200 or 301 will trigger an error. If the response is 200, the content is examined and A) must contain a specific phrase and/or B) must *not* contain a specific phrase. If these conditions are not met, an alarm is triggered specifying the customer, site, and other specific details.

The Website Monitoring function must be enabled in the ITPSvcCfg.ini. file. In the **WEBCHECK** Section, the **Enabled** value must be set to a Boolean True value. Once enabled, all further configuration is maintained by the website monitoring module.

Configuration File

The configuration file – ITPWSM.INI – is stored in the KWorking folder and uses the following sections and values:

COMMON Section

Subject This entry defines the format of the alert that will be generated when a site failure is detected. The following macros are supported:

&ORG&	The Customer Org ID
&URL&	The URL being queried
&TYPE&	The failure type
&UID&	The specific test ID (section name)

The default subject format is:
WEB-OAM|Url Monitor|&D1&|&D2&|&D3&|WEB.OAM.S.P3.Alm
The MonSetID in the last field should not be changed. The subject MUST contain 6 fields as shown above.

Count The number of consecutive errors before generating an alarm.

URLS Section

This defines the URLs to query for Keep-Alive functionality. Each value is the URL and is defined with a value of 1 to enable the query. Setting the value to 0 disables that URL.

MONITOR Section

Each monitor section has a unique name representing the parameters controlling that monitor. The section name usually identifies the check.

CustomerID The Kaseya customer Organization ID and secondary ID, separated by a period (.).

Url The URL to test, including “http://” or “https://”

Contains A text string that the response must contain.

NotContains A text string that the response must not contain.

Example showing a check for an “offline for maintenance” message that would create an alarm.

```
[MSP Builder-Home]
CustomerID=mspbldr.main-website
url=https://www.mspbuilder.com
NotContains=Offline for Maintenance
```

Customization Examples

Service Configuration – ITPSvcCfg.ini

This is an example of a Service Configuration File for any VSA platform:

```
; Configuration for the ITP Service
[GLOBAL]
; Interval (in seconds) between checks - default is 60 - 60-300 is recommended (1-5 minutes)
; Interval should be smaller when many tickets are generated, or high-priority systems
; are monitored.
Interval=120

; List of Service Desk Names to allow - when Service Desk is active.
; Blank to permit ALL Service Desks.
ServiceDesks=MSPB

; SelfUpdate args
; Disable nightly self-update - USE WITH CAUTION
;SelfUpdate=N
; Where to get updated from – DO NOT CHANGE except under direction of MSP Builder support!
;UpdateURL=https://dist.mspbuilder.com/

; Define whether the LAUser account will be generated and deployed on all
; workstations and on what schedule
; 0=Disabled, 1=First of month, 2=First & Fifteenth of month, 3=Every Monday.
[LAUSER]
LAUserCycle=3

; Should web checks be performed?
[WEBCHECK]
Enable=Y

; Should old agents be removed from VSA?
[AGENT_CLEANUP]
Enable=Y
Aging=60
Limit=25
```

SAAS Platform Requirements

The ITP service is usually installed on the VSA server for on-prem platforms. When a SAAS implementation is used, the ITP service must be installed on a server under the MSP's control. This can be an on-prem or a cloud server. Resource requirements are minimal, and a workstation (Windows 10) operating system can be used.

The SAAS platform also *requires* that the user accounts be represented as email addresses. As of 2020, the API account for all MSP Builder customers uses the format MSPB-APP@*custid*.mspb. The "custid" is the MSP Builder customer's Account ID. This does not need to be a valid email address or domain.

Process Engine Configuration – ITPConfig.ini

This configuration file has several sections, some of which can be duplicated to support "tenancy" – the ability to define configuration settings for specific VSA organizations. Essential customizations are highlighted in yellow.

GLOBAL Section

The GLOBAL section defines system-wide parameters. These apply without regard to tenancy controls.

```
[GLOBAL]
; Period, in minutes, where repeat events are considered duplicates for regular
; and actionable remediation events.
DedupPeriod=1440
ActDedupPeriod=30
```

The ActDedupPeriod should be smaller to allow faster remediation attempts and should not exceed 60 minutes.

```
;
; Period, in minutes, to wait for patch-related alerts to combine into a single ticket
PFSummaryPeriod=60
;
; Email From Address of VSA Server (can be overridden by MSP-specific setting)
VsaEmail=kaseya@msp.com
;
; VSA Admin Email - where cancellation notices and process alerts get sent, if the
; email is defined
VsaAdminEmail=ITPNotify@msp.com
```

This should represent the VSA admin or NOC team. Notification can be disabled by removing this definition or setting the following value to zero.

```
;
; A BINARY value to control failure and cancellation messages.
; 0=none, 1=Send Fail messages, 2=Send Cancel messages
NotifyFailCancel=3
;
; Directly Pass unprocessable events instead of cancelling if True.
DirectPass=Y
```

We recommend that DirectPass be enabled during initial setup and testing so all alerts become PSA tickets. As you learn which events to drop and configure the filtering, you can turn this off and rely solely on logging and/or cancellation email notifications.

```
;
; If Service Desk is in-use, list the terms in Notes that should be ignored - comma-delimited
list
SDNoteIgnore=Auto Generated,MSP Builder,submitter email address,NOC INTAKE,MonSetName
Expansion,2B - Validate,3 - Complete submits
;
; Helps to classify the event platform source, particularly when generic monitors are used.
ServerOS=2008,2012,2016,2019,Linux
WkstnsOS=7,8,8.1,10,MAC OS X
;
; Show and log a summary of data if true
ShowSummary=N
```

This MUST be disabled in production service mode! This is usually enabled when manually invoking the service executable from a command line during testing or evaluation. When enabled, the data collected is displayed for 3 seconds as each event is processed.

RMM_SETTINGS Section

This section defines all of the MSP-specific settings. The [RMM_SETTINGS] section must exist to define the primary MSP configuration. The ITP Service supports "tenancy", which allows additional RMM_SETTINGS sections to be defined, one for each additional tenant. The additional sections include the Tenant ID as defined in the Organization Custom Field called "MSP".

```
[RMM_SETTINGS]
; Name is for descriptive purposes only
Name=Technology Services
;
; RMM Email From Address
RMM_EmailFrom=kaseya@msp.com
```

This sets the email "from" address when sending email. All email is sent through the VSA.

```
;
UseUniqueFrom=Y|N
UseUniqueSender=Y|N
```

Theses options change the From address to help identify the customer org associated with the alert. Only one can be enabled, do not enable for CW Manage or other PSAs that can identify the client via subject parsing rules.

```
;
; PSA Type and Communication Method
PSA_Type=GEN
```

This defines the PSA type, which defines the interface module to load. Currently, this can be "GEN" for a generic, email-only connection, "CWF" for an API interface to ConnectWise, ATKand "BMS", for an API-based connection to Kaseya BMS.

```
;
; Defines the PSA Subject Line Format
PSA_Subject=RMM|<agent>|<estatus1>|<hostclass>|<altsubject>
```

The subject delivered to the PSA can be defined here, using the macros defined in the User Guide.

```
;
; If COMM method is API, must get ticket # in PSA for notification
PSA_Email=psa_intake@msp.com
```

The email address used to send events to the PSA.

```
;
; PSA Hostname and credentials for API calls
PSA_URL=cw.msp.com
PSA_AV1=msp
PSA_AV2=skE4coR29Dgk6
PSA_AV3=1someDRn9skEK4j7wX
PSA_AV9=MSPB-APP
```

The URL and credentials to authenticate to the PSA via the API. The last item is the User ID assigned to alarm tickets. See the Appendix for PSA-specific configuration settings.

```
;
; Define the target "board" for alert events
PSA_CW_Board=Alerts
```

If the PSA supports various "boards", this defines which one to use for alerts.

```
;
; CleanBody will remove the "for more information..." from EVT Log events if true
CleanBody=Y
;
; Drop specific MonSetID ActionType Codes
; Informational and Status Checks don't generate tickets.
DropCodes=Inf,Chk
```

The last field of the Monitor Set ID contains an "action identifier". "Alm" results in a ticket with no additional processing, "Act" triggers a remediation action that always results in a ticket, and "Req" triggers a "request" procedure that never results in a ticket. Other action types, such as "Inf" (information) and "Chk" (check) exist, but usually do not require a PSA ticket to be created. This section can define any MonSetID's action identifier, and if the event's action identifier matches this list, the event is cancelled.

```
;
; WeekDay/Weekend/Holiday help-desk operating hours. Use 00:00 / 23:59 for 24-hour operation;
00:00 / 00:00 for no operation.
WD_Start=8:00
WD_End=16:59
WE_Start=00:00
WE_End=00:00
HO_Start=00:00
HO_End=00:00
;
; Use holiday set ALL & 1
HolidaySets=0,1
```

This organization uses HOLIDAYSET_1 and HOLIDAYSET_2 to define the active holidays. HOLIDAYSETs will be discussed next.

```
;
; Early AM Notify Times
WD_NotifyTime=06:00
WE_NotifyTime=08:00
HO_NotifyTime=08:00
```

When a priority event occurs outside of customer coverage hours, the notification will be queued, and a single notification will occur for all alert events at the time specified. The times can be defined for weekdays, weekends, and holidays.

```
;
;Notify Process name
NotifyProcess=OpsGenie
```

This identifies the third-party notification service. The name here MUST match the name of the corresponding notification module (.mod) file.

```
;
; If NotifyControl is 0, the after-hours notification(s) are sent immediately.
; If the value is set to 1, then they are queued and sent on 5-minute intervals by the service.
NotifyControl=1
```

A value of 1 is recommended to prevent flooding the notification service with requests.

```
;
; After-Hours Notification Process ID and Priority Level (1=highest/5=lowest)
NotifyPriority=2
;
; Distribution group for after-hours/on-call notification of priority events
NotifyOCEmail=AfterHours@msps.com
```

A detailed email is sent to this distribution group for each alert that meets or exceeds the NotifyPriority value.

```
;
; Boolean value to control if the On-Call is always notified by email of priority events or only
when the Help Desk is not operating.
NotifyOCAlways=N
;
; Notify parameters - email to, subject, and body.
NotifyEmail=alerts@mspstechnologies.opsgenie.net
```

When notifications are made by email, this parameter defines the “To” address. The subject and body text (below) are always required

```
;
; Subject and Body for notifications Quotes must be escaped (\"), \n = CR/LF
NotifySubject=A high-priority event has occurred.
NotifyBody=Please check the recent emails with \"Priority Alert Notification\" in the
subject.\nMultiple events may have occurred!
;
; AdvanceWarn defines an email to send notification to at the start of a high-priority
remediation process
;AdvanceWarn=user@domain.com
```

This will send an email immediately when a priority alert arrives. Generally not needed.

HOLIDAYSET_# Section

One or more sections can be defined to identify holidays. HOLIDAYSET_1 is usually used for holidays that never change, such as New Year's Day and Independence Day. HOLIDAYSET_2 and higher are used individually or in combination by different tenants to define a complete set of holidays. Using multiple definitions allows a tenant to use multiple sets without the need to duplicate dates.

```
[HOLIDAYSET_0]
0101=New Year's Day
0704=Independence Day (US)
1225=Christmas Day
```

These are holidays with dates that don't change year to year.

```
[HOLIDAYSET_1]
;2018
0528=Memorial Day
0903=Labor Day
1122=Thanksgiving Day
1123=Day After Thanksgiving
```

These are the standard holidays observed by the MSP.

```
[HOLIDAYSET_2]
;2018
0115=Martin Luther King Day
0219=Presidents Day
1112=Veterans Day
```

These are holidays observed in addition to the above by one of the tenants.

Note that in the above example, the holidays in sets 1 and 2 are noted as being for a specific year. These need to be updated each January.

RESTRICTED Section

This section maps alert categories to time-based restrictions. This section can be MSP tenant specific.

```
[RESTRICTED]
; When can Class 4 alerts occur on Weekdays?
Class4_WD_Start=08:00:00
Class4_WD_End=17:00:00
; When can Class 4 alerts occur on Weekends?
Class4_WE_Start=00:00:00
Class4_WE_End=00:00:00
; Define the restricted alerts and classes
; Class 0: Allow alerts (also if not defined)
; Class 1: Never alert - monitor only
; Class 2: During helpdesk operational hours/days only
; Class 3: During customer coverage  hours/days only
; Class 4: Daily during allowed times defined in this section
;
;These are both performance monitors - Those called PERF alert only when the help desk is
; active, while those called PMON never alert
;
; Define the alert Category (MonSetID Field 2) and Restriction Class.
PERF=2
PMON=1
STS=0
```

This allows precise control over the type of monitors that can become alerts, and even during what times they are permitted. Note that "STS" is defined but set to always allow alerts – this technically is not required, but allows changing this value during testing, upgrades, or other controlled events to prevent events from being turned into tickets.

ALERT REWRITE Section

This is perhaps the most powerful capability of the ITP service, as it allows events to be parsed and controlled with great flexibility. This is most often used when non-agent devices send an email to VSA and the subject is fairly generic. The VSA Admin has the ability to examine the subject line and cancel further processing, replace the generic subject with one that can be further parsed and processed, or even invoke a custom processing module that can generate a proper parsable subject based on an examination of the event subject, body, and even external conditions.

```
; Alert Events received by VSA (usually by email) can be rewritten into proper alert
; formats or CANCELLED. This processes INCOMING ALERT EVENTS.
; Subject Fragment=Action; [Category|Event Text|D1|D2|D3|MonSetID] | ProcessName
; Action is REWRITE, CANCEL, or PROCESS
[ALERT REWRITE]
; Cancel these
passed a self test=CANCEL;;
started a self test=CANCEL;;
```

These common messages from APC UPS devices can simply be dropped.

```
; Redefine the SourceType and Subject
failed a self test=REWRITE;PWR|Power|UPS|Fault|SelfTest|Power.UPS.X.P3.Alm
EMU|SEVERE=REWRITE;ENV|High Temp|Environment|Temperature|HIGH|Temp.ENV.X.P2.Alm
```

This example shows a fragment from an Environmental Monitor that is rewritten into a proper, 6-field parsable subject, which sets information including the priority.

```
; Run a Special Process
Active Threats on=PROCESS;ITP-ISR-Webroot
Install Failed on=PROCESS;ITP-ISR-Webroot
Uninstall Failed on=PROCESS;ITP-ISR-Webroot
```

These subject fragments invoke a modular process that extracts the proper agent ID and defines the proper subject line. The same process handles the conversion for all three event types.

REMEDIATION Section

This section maps specific events to specific Agent Procedures that perform remediation tasks. The first two fields of the MonSetID are used to map an actionable monitor (field 5 of the MonSetID is “Act”) to a specific procedure. Each line defines the procedure name, and up to four Value/Parameter pairs. The Agent Procedure name must be accurate and accessible by the MSPB-APP account.

```
; Remediation Table
; Maps an event to a remediation procedure with up to 4 argument pairs
; Each arg,value pair represents a script-prompt name and the value to provide it.
; The value can be fixed text or a macro representing available data within the app.
[REMEDIATION]
; ID=AgentProcedure;Arg1,val1;Arg2,val2;Arg3,val3;Arg4,val4
; New Agent Init
CheckIn.NAI=ALL-Agent Onboarding - 1 - Init;;;;
; Server - Suspend Alarms
MB-MNT-SUSP_ALM-A.EVT=ALL-Alarms - Suspend 240 Minutes;;;;
; Workstation - Run Maintenance Now
MB-MNT-RUNNOW-A.EVT =WIN-MB Maint - Workstation - Immediate (MV);;;;
; Service Remediation
*.SVC=WIN-Remediation-Service;EventId,<eventcode>;SvcName,<data1>;;
```

This example illustrates two key features – the ability to use a wildcard in the name match and passing of two parameters. The first – EventId – simply passes the ID of the event to correlate the event by writing it to the agent procedure log. The second argument uses the second event data value to identify the service to be remediated.

```
; TIME Remediation
;MB-SM-TIME-A.EVT =
; KNM Gateway Restart
MB-KNMG-A.EVT=WIN-W32Time - Set Configuration;;;;
```

BLACKLIST Section

Agents or groups of agents where automation and monitoring is desired but PSA tickets are not wanted can be identified here. All automation is performed, but any ticket created to be sent to the PSA will simply be discarded.

```
[BLACKLIST]
; a specific agent
agent.machine.group=Y|N
; all agents in a specific machine-group
machine.group=Y|N
```

TICKET NOTES / INTERNAL NOTES Sections

Text and variable information can be injected into the ticket body for every ticket as well as for specific alert types. Up to 10 Custom Fields can be inserted per definition line to insert agent-specific notes. When an API Integration is used, a second section called "INTERNAL NOTES" can be defined, allowing notes to be added to the internal as well as primary (body) sections of the ticket.

The form **`FILE:name.ext`** can be used in place of a message string. When found, the file (located in the ITP\TXNOTES folder) will be loaded and a macro-replacement performed, allowing large volumes of text to be inserted. This works for both TICKET NOTES and INTERNAL NOTES. Since both body and internal note files are in the same folder, we recommend using a "TN_" prefix for body notes and a "IN_" prefix for internal notes.

```
[TICKET NOTES]
; The SuppressLCLink will suppress adding the Live-Connect link to the ticket body if true
;SuppressLCLink=Y
;
; Write this message to every ticket - example shows a custom-field injection
;Common=<CF:TktMessage> <CF2:fieldname>
;
;MB-KNMG-A.EVT..Pass=Restarted the gateway service {T:5}
;MB-SM-A.EVT=A low-disk space condition is present {T:5}
```

This example illustrates the use of a PSA Macro that adds 5 minutes of technician time to the ticket. This uses a PSA-specific macro format - check your PSA manual for proper macro use.

The same format of parameters can be used in the INTERNAL NOTES section, except for the SuppressLCLink parameter, which is only supported in the Body section.

PSA REWRITE Section

ITP delivers emails to the PSA using a machine-parsable format. While this makes parsing and classifying events easy, it can prove challenging for the technician to make sense of the ticket subject. The PSA REWRITE section maps the machine-parsable subject into a simple code plus a plain text message to describe the event. The PSA REWRITE section supports a Tenant ID.

MSP Builder provides an Excel spreadsheet that allows you to define the monitor sets, which generates the machine-parsable subject. This is then mapped to a code and text message. The result can be copied from the spreadsheet directly into the PSA REWRITE section of the configuration file. The spreadsheet and the ITPConfig.ini file are already populated with the complete set of MSP Builder monitors.

The rewrite section consists of a lookup of four parts, and a rewrite value of three parts.

Lookup Format

The lookup section consists of the Event Name, Type, Host Class, and an Event ID for Event Log alarms. `MB-ADDC.EVT.S.1150` is an Active Directory server monitor. This is placed on the left-side of the equal sign. Lookup values for Service monitors have just 3 parts and do not include the Event ID.

Rewrite Format

The Rewrite section consists of three parts - an email parsing code that uniquely identifies the event, including host class and priority; the plain-text message; and three classification values. The corresponding rewrite for the Active Directory alert above would be:

```
EVT-H-S1001150;AD Svcs-CRITICAL: Schema modification attempt failed;Active
Directory,Infrastructure,Schema
```

When sending an email to the PSA, the code and message are included, and the classification data is stripped off. Your parsing rule should use those classification values. When ITP interfaces with the PSA via an API, only the text message is included. The API will automatically use the classification data when the ticket is created.

Configuring Service Desk

Service Desk is used by VSA and the RMM Suite to provide an intake of email-based alarm notifications. These generally come from non-agent devices such as networking gear (routers, firewalls, and switches) and power devices (UPS and environmental monitoring),

ITPSVCCFG.INI File Settings

There are two settings that must be defined in the COMMON section:

- SuppressSDTickets=N
  This allows Service Desk to be queried for alerts.
- ServiceDesks=MSPB
  This limits the query to the RMM Suite defined desk. If the MSP has defined other Service Desks that should be queried, their IDs should be added to this list, separated by commas.

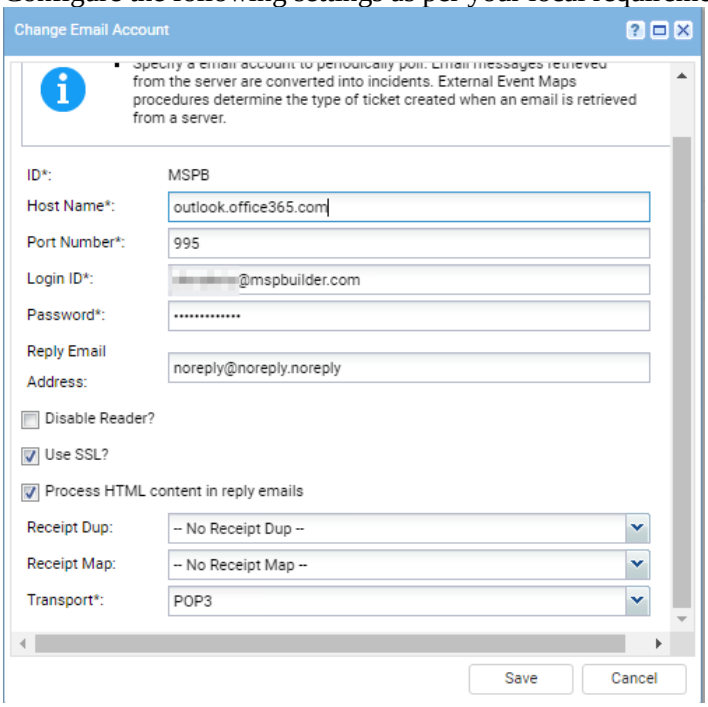
Import the MSPB Service Desk

The file '*MSPB_Service_Desk.xml*" should be imported. This is performed by MSP Builder support staff.

Configure the Service Desk Email Reader

Navigate to Service Desk > Common Configurations > Incoming Email/Alarm

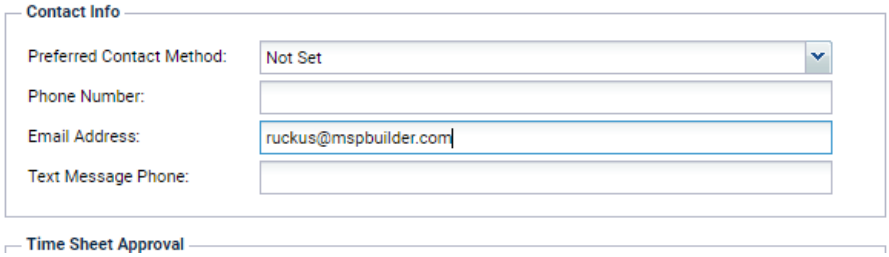
1. Move th the "Readers" tab and click "new"
2. Set the ID to "MSPB"
3. Configure the following settings as per your local requirements:



Mapping Devices to Client Orgs

Navigate to System > Org/Grps > Manage, select an organization and click the Staff tab.

Click “new” to create a “staff member”, which will map incoming email addresses to specific organizations. Add the required fields as shown below:



Repeat the above for each organization (and possibly device type). A generic entry can be used, such as “alarms*@customer.com*”, mapping all email alarms to a specific customer. All devices that send email alarms to VSA should be configured with the corresponding, customer-specific from-address.

ConnectWise Manage Integration

MSP Builder offers an API-based integration module for ITP that provides the following capabilities when using the ConnectWise Manage PSA platform:

- Uses Alert Rewrite to define Type, Subtype, and Item classifications.
- Provides an installation utility to create the T/S/I classifications and necessary associations.
- Identifies and updates existing tickets when the alert occurs again.
- Creates new tickets, associating the event with the proper customer and configuration item; sets the priority, targets the desired service board, and assigns the T/S/I classification values. The new ticket number is returned to ITP and included in any After-Hours notification emails.
- An email-based backup to API ticket delivery makes sure tickets get through even in the event of API or credential issues in Manage.

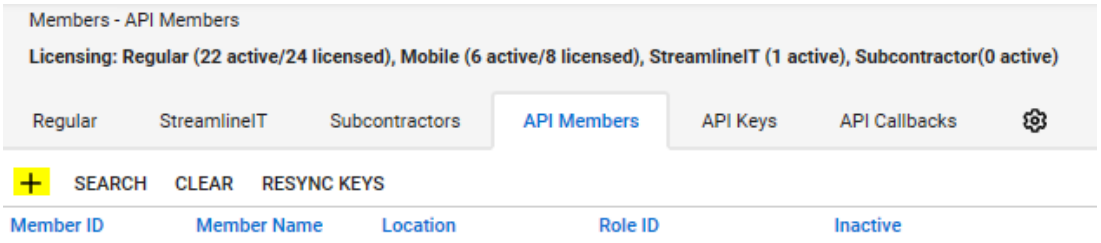
Installation

Installation requires only a simple configuration change in the ITPConfig.ini file to switch to the CWF module and define the account credentials. Security is protected through an MSP Builder token assigned by ConnectWise and embedded into our code. An API account needs to be created in ConnectWise Manage and the associated keys added to the configuration file.

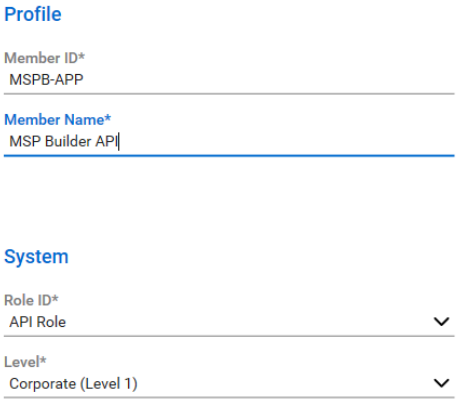
Create the ConnectWise Manage API Member

The following steps assume you have proper administrative rights on your ConnectWise Manage server!

1. Log into Manage as an Administrative User
2. Navigate to System - Security Roles and locate the API Role. If an API Role is not present, you will need to add one and assign the appropriate rights to the role before continuing. Provide the following access:
    a. Companies and Configurations - Inquire level
    b. Tickets - Create, Modify, and Inquire levels
    c. System Table Setup - Create, Modify, and Inquire levels for T/S/I creation and mapping only. This access is needed only by the setup tool to create the T/S/I values.
3. Navigate to System - Members, select the API Members tab. Click the "+" sign to add a new account.



4. In the profile page, create a Member ID of "MSPB-APP", a Member Name of "MSP Builder API", and assign the "API Role" to the Role ID.



Copy and save the Public and Private keys when displayed and save them!

Define the ITP to PSA Configuration

Log into the server where ITP is installed. Locate the ITPConfig.INI file in the KWorking directory and open it in a text editor such as Notepad or Notepad++.

Locate the "[RMM_SETTINGS]" section and define the following settings:

```
PSA_TYPE=CWF
PSA_URL=Your public CW Manage URL
PSA_AV1=Your Company ID
PSA_AV2=Your Public Key (from Step 4)
PSA_AV3=Your Private Key (from Step 4)
PSA_AV9=MSPB-APP or a CW user ID used when creating a ticket
PSA_CW_Board=The default CW Manage Board Name for Alert Tickets
PSA_CW_Source=The "Source Type" for alert events, such as "Monitor Alerts"
```

Do not modify the parameters in red - these are required to enable the CW Manage integration!

Review the Event Mapping & Classification Table

The EventMapping.xlsx spreadsheet was included in your CW Manage Integration Kit. This spreadsheet identifies every monitor that the RMM Suite provides, defines the alternate text message, Type, Subtype, & Item classifications, and the Lookup Table used in the ITPConfig.ini file.

Use extreme caution when modifying this spreadsheet as it can affect the operation of the integration module! Only edit the data in columns H, J, K, and L.

Column H - Alert Subject Text

This defines the plain-text description of the event that will be placed in the ticket title. This text MUST be no longer than 60 characters or data may be lost. Column N will display the length and will be shaded in red if it exceeds 60 characters. The existing messages accurately reflect the alert message and should be changed only when necessary.

Columns J, K, & L - Event Classification Data

These columns represent the Type, Subtype, and Item classifications assigned to the MSP Builder monitor sets. These can be changed to match your existing classifications if desired.

Suggested changes are matching your existing values, such as replacing "Active Directory" with "AD" or "Server" with "Servers". The installation utility will determine which values exist, which need to be created, and will create the proper associations.

Our standard rewrite configuration is present in current configuration files, so this step is optional. If you made any changes to the Alert Subject Text or Type/Subtype/Item values or have an older copy of the configuration data without the lookup table, copy all of the data values from the "Config Data-EMAIL+API" (column Z) and paste it into the "[PSA REWRITE]" section in the ITPConfig.ini file, *replacing any data that may already exist in that section*.

Create the T/S/I Data Mappings

The CWSetup.BMS application will index the required mappings defined in the ITPConfig.ini file, then evaluate the available mappings defined in ConnectWise Manage. Any data mapping that doesn't exist will be created and/or mapped as necessary.

1. Login to the ITP server, open a command prompt and CD to the KWorking folder
2. Run BIN\RMMKSE.EXE ITP\CWSetup.BMS and wait for it to complete. If this step fails, verify that the credentials you defined are correct and try again.

The integration is now ready for use.

Configuration Options

The following configuration items are specific to the CWF integration module and are located in the RMM_SETTINGS section(s) that target the CW Manage PSA.

Board

This defines the default board where alerts are delivered to. A typical configuration is to have alerts go to one board and requests go to another so that client requests can be prioritized. The value can be the board name or it's numeric ID.

```
; Define the target "board" for alert events
PSA_CW_Board=Alerts
```

Source Type

Manage allows each ticket to be classified by its source. In most cases, the source of "Monitor Alerts" will not need to be changed unless the MSP has defined an alternate name for this data source.

```
; Define the target "source type" for alert events
PSA_CW_Source=Monitor Alerts
```

New and Closed Statuses

By default, auto-remediated events look to assign a ticket to "Completed" status. New events which need to be reviewed by a technician assign a ticket to "New" status. If you are using different names for these statuses, configure the options below to ensure the correct mappings are applied.

```
PSA_TicketClosed=Completed – Auto Remediation
PSA_TicketNew=New – RMM Kaseya
```

Priority Level

Each of our events has a predefined priority from P1 (Critical) through P5 (information). By default, these map to Critical, High, Medium, Low, and Info in ConnectWise. Populate the relevant priority options for additional customization or to map to your customized Priority names. Levels P6-P9 are used for MSP-defined custom monitors and are not part of the standard RMM Suite monitor sets.

```
PSA_P1=Critical
PSA_P2=High
PSA_P3=Medium
PSA_P4=Low
PSA_P5=Information
PSA_P6=Software Licensing
```

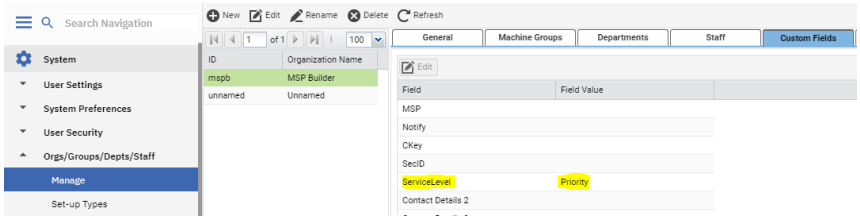
```
PSA_P7=No SLA
PSA_P8=Do Not Bill
PSA_P9=Approval Required
```

Service Level Board Routing

This is an optional configuration that allows the MSP to route alerts to multiple service boards based on a Service Class and Host Type assigned in VSA. Each customer can be assigned a Service Level ID. This is set in the Org Custom Fields and should be a single term (no spaces) that identifies a level of service (Bronze/Silver/Gold or similar). This term will map to a custom field in the ITPConfig.ini file to define the target service board in Manage.

```
; Define a mapping from Service Level Name and (optional) host type to CW Board Name
;PSA_CW_<SvcLevelName>.<HostType>=<CW Board Name>
;
;EXAMPLES:
; Define the service board for normal class client alerts, all host types
PSA_CW_Normal=Alerts
; Define the service board for priority class client alerts, workstations only
PSA_CW_Priority.W=Alerts
; Define the service board for priority class client alerts, servers only
PSA_CW_Priority.S=Priority Alerts
```

Consider the following example of Normal and Priority service levels. Most clients are assigned the Normal service level code, but a few belong to the Priority service level. The MSP creates two boards for alerts - “Alerts” and “Priority Alerts” and wishes only server alerts for priority class clients to be routed to the Priority Alerts service board. To accomplish this, the entries as shown above are created in the ITPConfig.ini file. Then, the Service Level is assigned to the client as shown below. Navigate to the **System - Orgs/Groups.. - Manage** menu, select the **Custom Fields** tab, then the **ServiceLevel** field and click **Edit**. Enter the name of the Service Level to use. This must match what hass been defined in the configuration file, as shown above. The value can specify either the board name or its numeric ID.



If the client’s ServiceLevel field is undefined, the alarm will route to the board defined by the **PSA_CW_Board** parameter. There is no distinction made between server and workstation alerts on the default board.

There are no limits on the number of Service Level types that can be defined. This can also be used to route alarm events to client-specific ticket boards without setting up a separate “MSP ID” section in the ITPConfig.ini file.

Note: The Host Type is based first on the machine-group where the agent is located. This will be either “S” for servers or “W” for workstations (“wkstns” group). If the agent is NOT located in or below one of these groups, the host type will be identified based on the device’s defined operating system. If ITP is unable to determine the operating system, the host type will be set to “X” (multiple types or non-computer). This host type will also be used for non-computer devices that send OOB alarms, such as firewalls, UPS/Power devices, and switches. This host type code should be accounted for when configuring Service Level processing, OR an entry without any Host Type value should be defined. *Incomplete configuration could cause alerts to be routed to the incorrect board.*

Datto Autotask Integration

MSP Builder offers an API-based integration module for ITP that provides the following capabilities when using the Datto Autotask PSA platform:

- Uses Alert Rewrite to define **Issue Type** and **Sub-Issue Type** classifications.
- As of the current release, there is no API function available for creating classifications. Once you have reviewed Event Mapping.xlsx for the classifications you will be using, you must create the records and associations in your PSA.
- Identifies and updates existing tickets when the alert occurs again.
- Creates new tickets, associating the event with the proper customer and configuration item; sets the priority, targets the desired service board, and assigns the ISIT classification values. The new ticket number is returned to ITP and included in any After-Hours notification emails.
- An email-based backup to API ticket delivery makes sure tickets get through even in the event of API or credential issues in Manage.

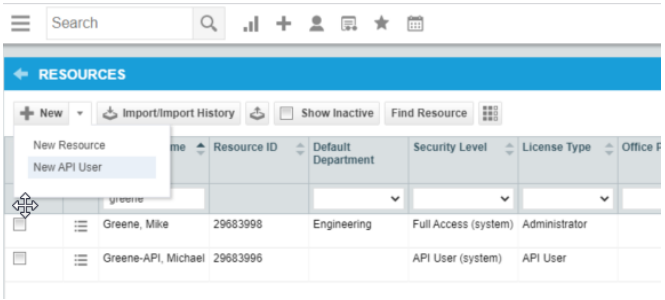
Installation

Installation requires one change to configuration option in ITPConfig.ini file to switch to the ATF module and define the account credentials. Security is protected through an MSP Builder token assigned by Datto and embedded into our code. An API account needs to be created in Datto Autotask and the associated keys added to the configuration file.

Create the Datto Autotask API User

The following steps assume you have proper administrative rights on your Datto Autotask instance!

1. **FirstName** = "MSP Builder"
2. **LastName** = "App Account"
3. **Email** = "engineers@mspbuilder.com"
4. **Security Level** = "API User (system)"
5. **Username** = "MSPB-APP@yourdomain.com"
6. **Password** can be anything but must be entered into our setup tool. Make sure you save this key or you will need to recreate the account.
7. **Integration Vendor** = "MSP Builder – RMM"



Configuration Setup

The ATSeup.BMS application will prompt you for authentication and ticket creation required fields, then, write them to ITPConfig.ini.

1. Login to the ITP server, open a command prompt and CD to the KWorking folder
    a. Dash dash init – REQUIRED to set initial configuration writes.
    b. Dash dash mkcontacts – OPTIONAL. Authenticates to your VSA and creates a list of all your client organization names in memory. This does a look up to the Autotask Rest API to see if the organization exists with an exact match name. If the organization exists, a contact is created in your PSA under this organization using the email address you provided in RMM_EmailFrom. String data before the "@" is replaced with the Kaseya VSA org ID and this is used to map email ticket submissions to the correct organization. Result to expect: clientID@yourdomain.com
        i. Organizations not found are logged. Check the log, correct any necessary mappings, and update on either platform to match the other. Script can be run again if necessary. Contacts are not created again if they already exist.
2. bin\RMMKSE.EXE itp\ATSetup.BMS –init (dash dash)
3. Enter the fully qualified API user we created above.
4. Enter the private key for the API user. Written using an encryption key and decoded before use.
5. Enter the default Queue name to route alert tickets to.
6. Enter the default Source name to route alert tickets to.
7. Off by default options are written to configuration file, i.e. PSA_TicketClosed=0, PSA_P1=0. These will not be used unless you configure customizations.

The integration is now ready for use.

Define the ITP to PSA Configuration

Log into the server where ITP is installed. Locate the ITPConfig.INI file in the KWorking directory and open it in a text editor such as Notepad or Notepad++.

Locate the "[RMM_SETTINGS]" section and define the following settings:

```
PSA_TYPE=ATF - The only option which must be set manually to enable to module
PSA_AV1=MSPB-APP@yourdomain.com (fully qualified account we created above)
PSA_AV2= Your Private Key (generated during account creation, now encrypted)
PSA_AT_Queue=The default Datto AT Queue Name to auto assign for Alert Tickets
PSA_AT_Source=The "Source Type" for alert events, such as "Monitoring Alert"
PSA_TicketClosed=Name of a status representing a remediated event. Default "Completed"
PSA_TicketNew=Name of a status representing a new event. Default "New"
PSA_P1=0
PSA_P2=0
PSA_P3=0
PSA_P4=0
PSA_P5=0
PSA_P6=0
PSA_P7=0
PSA_P8=0
PSA_P9=0
```

Do not modify the parameters in red - these are required to enable the CW Manage integration!

Review the Event Mapping & Classification Table

The Event Mapping.xlsx spreadsheet was included in your Datto Autotask Integration Kit. This spreadsheet identifies every monitor that the RMM Suite provides, defines the alternate text message, Type, Subtype, & Item classifications, and the Lookup Table used in the ITPConfig.ini file.

Use extreme caution when modifying this spreadsheet as it can affect the operation of the integration module! Only edit the data in columns H, J, K, and L.

Column H - Alert Subject Text

This defines the plain-text description of the event that will be placed in the ticket title. This text MUST be no longer than 90 characters or data may be lost. Column N will display the length and will be shaded in red if it exceeds 90 characters. The existing messages accurately reflect the alert message and should only be changed only when necessary.

Columns J, K, & L - Event Classification Data

These columns represent the Issue, Sub Issue (Part 1), and Sub Issue (Part 2, optional) classifications assigned to the MSP Builder monitor sets. These can be changed to match your existing classifications if desired. If column K and L are both populated, we combine these as "Part1: Part 2" in your Autotask instance.

Suggested changes are matching your existing values, such as replacing "Active Directory" with "AD" or "Server" with "Servers". The installation utility will determine which values exist, which need to be created, and will create the proper associations.

Our standard rewrite configuration is present in current configuration files, so this step is optional. If you made any changes to the Alert Subject Text or Issue/Sub Issue values or have an older copy of the configuration data without the lookup table, copy all of the data values from the "Config Data-EMAIL+API" (column Z) and paste it into the "[PSA REWRITE]" section in the ITPConfig.ini file, *replacing any data that may already exist in that section*.

Configuration Options

The following configuration items are specific to the ATF integration module and are located in the RMM_SETTINGS section(s) that target the Datto Autotask PSA.

Queue

This defines the default board where alerts are delivered to. A typical configuration is to have alerts go to one board and requests go to another so that client requests can be prioritized. The value can be the board name or it's numeric ID.

```
; Define the target "board" for alert events
PSA_AT_Queue=Monitoring Alert
```

Source Type

Manage allows each ticket to be classified by its source. In most cases, the source of "Monitor Alerts" will not need to be changed unless the MSP has defined an alternate name for this data source.

```
; Define the target "source type" for alert events
PSA_AT_Source=Monitoring Alert
```

New and Closed Statuses

By default, auto-remediated events look to assign a ticket to "Completed" status. New events which need to be reviewed by a technician assign a ticket to "New" status. If you are using different names for these statuses, configure the options below to ensure the correct mappings are applied.

```
PSA_TicketClosed=Completed – Auto Remediation
PSA_TicketNew=New – RMM Kaseya
```

Priority Level

Each of our events has a predefined priority from P1 (Critical) through P5 (information). By default, these map to Critical, High, Medium, Low, and Info in Autotask. Populate the relevant priority options for additional customization or to map to your customized Priority names.

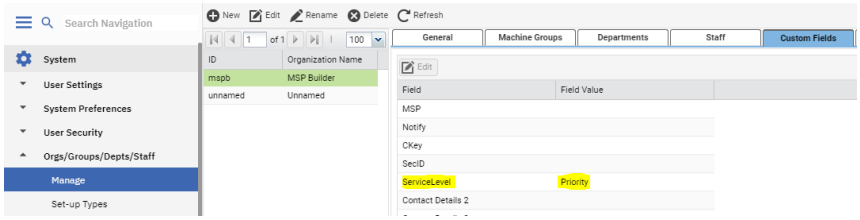
```
PSA_P1=Critical
PSA_P2=High
PSA_P3=Medium
PSA_P4=Low
PSA_P5=Information
PSA_P6=Software Licensing
PSA_P7=No SLA
PSA_P8=Do Not Bill
PSA_P9=Approval Required
```

Service Level Board Routing

This is an optional configuration that allows the MSP to route alerts to multiple service boards based on a Service Class and Host Type assigned in VSA. Each customer can be assigned a Service Level ID. This is set in the Org Custom Fields and should be a single term (no spaces) that identifies a level of service (Bronze/Silver/Gold or similar). This term will map to a custom field in the ITPConfig.ini file to define the target service board in Autotask.

```
; Define a mapping from Service Level Name and (optional) host type to AT Board Name
;PSA_AT_<SvcLevelName>.<HostType>=<AT Board Name>
;
;EXAMPLES:
; Define the service board for normal class client alerts, all host types
PSA_AT_Normal=Alerts
; Define the service board for priority class client alerts, workstations only
PSA_AT_Priority.W=Alerts
; Define the service board for priority class client alerts, servers only
PSA_AT_Priority.S=Priority Alerts
```

Consider the following example of Normal and Priority service levels. Most clients are assigned the Normal service level code, but a few belong to the Priority service level. The MSP creates two boards for alerts - “Alerts” and “Priority Alerts” and wishes only server alerts for priority class clients to be routed to the Priority Alerts service board. To accomplish this, the entries as shown above are created in the ITPConfig.ini file. Then, the Service Level is assigned to the client as shown below. Navigate to the **System - Orgs/Groups.. - Manage** menu, select the **Custom Fields** tab, then the **ServiceLevel** field and click **Edit**. Enter the name of the Service Level to use. This must match what has been defined in the configuration file, as shown above. The value can specify either the board name or its numeric ID.



If the client’s ServiceLevel field is undefined, the alarm will route to the board defined by the **PSA_AT_Board** parameter. There is no distinction made between server and workstation alerts on the default board.

There are no limits on the number of Service Level types that can be defined. This can also be used to route alarm events to client-specific ticket boards without setting up a separate "MSP ID" section in the ITPConfig.ini file.

Note: The Host Type is based first on the machine-group where the agent is located. This will be either "S" for servers or "W" for workstations ("wkstns" group). If the agent is NOT located in or below one of these groups, the host type will be identified based on the device's defined operating system. If ITP is unable to determine the operating system, the host type will be set to "X" (multiple types or non-computer). This host type will also be used for non-computer devices that send OOB alarms, such as firewalls, UPS/Power devices, and switches. This host type code should be accounted for when configuring Service Level processing, OR an entry without any Host Type value should be defined. *Incomplete configuration could cause alerts to be routed to the incorrect board.*

Kaseya BMS Integration

MSP Builder offers an API-based integration module for ITP that provides the following capabilities when using the Kaseya BMS PSA platform:

- Uses Alert Rewrite to define **Issue Type** and **Sub-Issue Type** classifications.
- As of the current release, there is no API function available for creating classifications. Once you have reviewed Event Mapping.xlsx for the classifications you will be using, you must create the records and associations in your PSA.
- Identifies and updates existing tickets when the alert occurs again.
- Creates new tickets, associating the event with the proper customer and configuration item; sets the priority, targets the desired service board, and assigns the ISIT classification values. The new ticket number is returned to ITP and included in any After-Hours notification emails.
- An email-based backup to API ticket delivery makes sure tickets get through even in the event of API or credential issues in Manage.

Installation

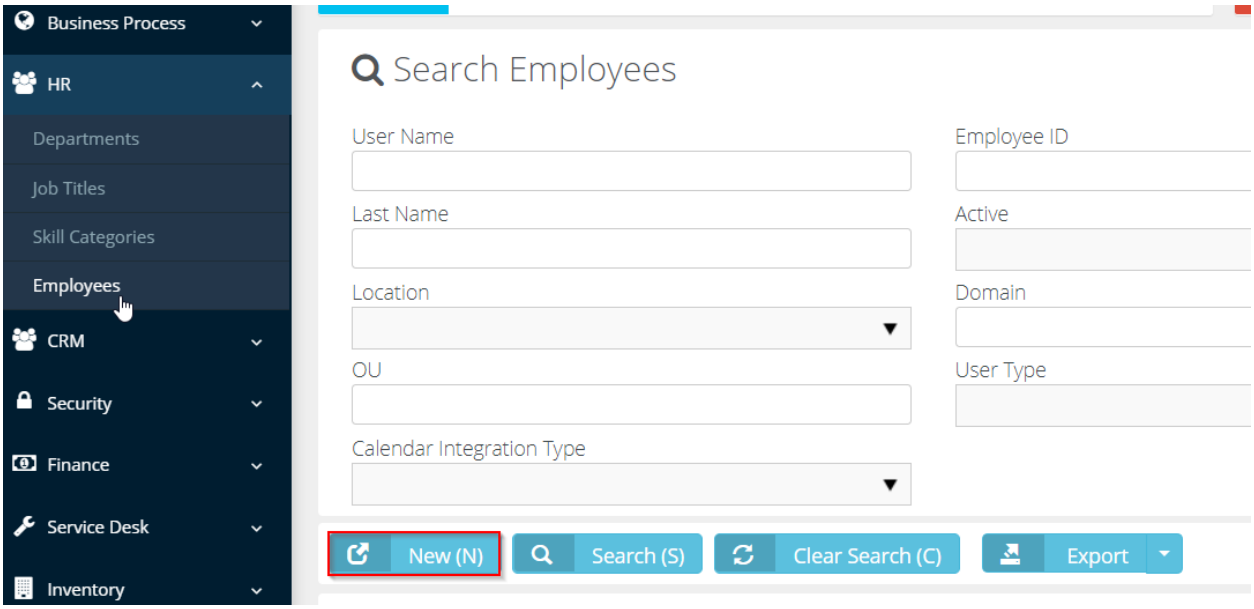
Installation requires one change to configuration option in ITPConfig.ini file to switch to the BMS module and define the account credentials. An API account needs to be created in Kaseya BMS and the associated keys added to the configuration file.

Create the Kaseya BMS API User

Navigate to your BMS instance (Admin > HR > Employees > New) to create the account:

1. User Name = “MSPB-APP”
2. First Name = “MSP Builder”
3. Last Name = “App Account”
4. Employee ID = Any unique number
5. Email Address = support@mspbuilder.com
6. Job Title = “Administrator”
7. Department = “Administration”
8. Manager = Any other administrator
9. Employment Type = “Full Time”
10. Employee Roles = “Administration”
11. Security Roles = “Administrator”
    This can be customized in Admin > Security > Roles.
    This must have **API Access=True** in the Admin section.
12. Location = Anything

13. User Type = "API Employee"
    a. The above selection disallows front end log in.



Prepare the Event Mapping Data

Review Event Mapping.xlsx from the MSP Builder website (Customer Login access required) on the Documentation page. Here you will find the events we monitor, the email subjects we create by default, and the ticket categories (Issue, Subissue 1, Subissue2). You can use our defaults or customize this to suit your needs or current environment. Without this step, the API will not auto-assign ticket classes when alert tickets are created.

Copy the customized classification data from the spreadsheet to your ITPConfig.ini file.

Notes:

1. Issue/Sub-issue creation is not available via REST API and must be created manually.
2. When creating MSPB Subissue, combine columns K and L with colon+space in between. [Excel formula: `=Concat(K, ": ", L)`]

Run the Init Command

Cd to the kworking folder on your ITP server and execute: `bin\rmmkse itp\BMSSetup.bms --init`

Consider using additional argument `--mkcontacts`
This will authenticate to your VSA and create a list of your organizations in memory. It will check BMS for a matching organization and create a contact (orgidentifier@yourdomain.com) in your PSA under the organization. This allows tickets generated by email to map to the correct org. If API ticket creation fails for any reason, the ticketing module falls back to email submission.

When the command runs, it:

1. Prompts for the BMS Server URL
2. Prompts for the API username
3. Prompts for the API account password, which is encrypted and written to ITPConfig
4. Prompts for the Server Tenant, the value other users enter at BMS log in
5. Prompts for the default Queue name to associate alarms with
6. Prompts for the default Source name to associate alarms with
7. Generates local file for BMS ID lookups

Customize the ITPConfig.INI File

Open ITPConfig in a text editor (notepad) and change any [RMM SETTINGS] sections PSA_TYPE option to BMS, globally, or for any custom organizations you've configured that you need to route to BMS. If you have created multiple routings for different clients, you may need to duplicate the [PSA REWRITE] section of ITPConfig for each ORGID, i.e [PSA REWRITE_ORGID] if these organizations utilize different classification parameters.

Configuration Options

The following configuration items are specific to the BMS integration module and are located in the RMM_SETTINGS section(s) that target the BMS PSA.

Queue (Included in BMSSetup)

This defines the default board where alerts are delivered to. A typical configuration is to have alerts go to one board and requests go to another so that client requests can be prioritized. The value can be the board name or it's numeric ID.

```
; Define the target "board" for alert events
PSA_BMS_Queue=Monitoring Alert
```

New and Closed Statuses

By default, auto-remediated events look to assign a ticket to "Completed" status. New events which need to be reviewed by a technician assign a ticket to "New" status. If you are using different names for these statuses, configure the options below to ensure the correct mappings are applied.

```
PSA_TicketClosed=Completed – Auto Remediation
PSA_TicketNew=New – RMM Kaseya
```

Priority Level

Each of our events has a predefined priority from P1 (Critical) through P5 (information). By default, these map to Critical, High, Medium, Low, and Info in BMS. Populate the relevant priority options for additional customization or to map to your customized Priority names.

```
PSA_P1=Critical
PSA_P2=High
PSA_P3=Medium
PSA_P4=Low
PSA_P5=Information
PSA_P6=Software Licensing
PSA_P7=No SLA
PSA_P8=Do Not Bill
PSA_P9=Approval Required
```