Nexpose Ticketing System Configuration Guide

Website: https://github.com/rapid7/nexpose_ticketing

Support: support@rapid7.com

Supported Ticketing Systems: JIRA, Remedy, ServiceNow, ServiceDesk

RAPID7

Last Revised: May 10, 17

Contents

Solution Summary

The goal of incident management is to restore normal service operation as quickly as possible following an incident, while minimizing impact to business operations and ensuring quality is maintained. The integration with Nexpose allows customers to create incident tickets based on vulnerabilities found across their systems. With this information in the chosen platform, said tickets can be assigned to work teams, prioritized and resolved.

Details

- Required Ruby Version: >= 2.1.5

The Ticketing System integration uses the Nexpose-Client Ruby gem, as well as the Nexpose reporting model, to generate information on the current state of the user chosen sites or tags in Nexpose, depending on the configuration. It then creates tickets either for each machine or vulnerability, as specified by the ticketing mode selected. On subsequent scans new tickets are created, existing tickets are updated (and potentially closed if resolved), based on any differences since the scan was performed. The configuration file also provides the option of using tags instead of sites, allowing the user to group assets across different sites together. This can be used to group targets that may need scanned more regularly, for example running daily scans on high risk targets across multiple sites without having to scan every site.

The prior scan information is tracked in a CSV file, with separate files for scan and tag runs. Deleting these files will cause the integration to act as if it is performing the initial scan upon its next execution i.e. new tickets will be created for each asset or vulnerability even if one already exists in the ticketing system.

Modes of Operation

Run Type

The Ticketing System can operate in two different running modes:

1. Site Run (Default): This mode will generate a report in Nexpose for any sites listed within the ticket_service.config file, as selected by the user. It will create tickets for any vulnerabilities found on any assets, based on the chosen ticket generation mode. If the current scan ID matches the scan ID stored in the historical file, the service will not check for new data. If no sites are listed, the ticket_service will check every site on the Nexpose server instance.

2. Tag Run: This mode will scan sites for assets marked with the tag specified in the ticket_service.config file. The service will then generate reports for the selected assets. The ticket_service will compare the scan ID of each tagged asset with the historical file before creating tickets or looking for updates for that asset. Tags have priority over sites and listing tags in the configuration file will prevent the integration from creating tickets for any listed sites. **Tag runs are not compatible with Vulnerability mode.**

Ticket Generation

There are three different modes for the ticketing system:

1. Default Mode: This mode will create one ticket per instance of a vulnerability i.e. a vulnerability present on three machines will have three tickets. This mode makes for smaller, more actionable incidents but has the potential to generate a large number of tickets. This mode can only create and close tickets. It does not update any information in existing tickets.

2. IP Mode: This mode creates a single ticket containing all vulnerabilities for each asset. This reduces the total number of incidents but can greatly increase the size of the work notes. On subsequent scans the ticket is updated to include new vulnerabilities and will mark fixed issues as "old".

3. Vulnerability Mode: This mode will create a ticket for each vulnerability, listing every scanned asset affected by this vulnerability for a site. This reduces the total number of incidents but can greatly increase the size of the work notes. On subsequent scans the ticket is updated to include any new information about the vulnerability and new assets with this vulnerability, as well as removing remediated assets. **Vulnerability mode is not compatible with tag runs.**

Differences between the Ticket Generation Modes

- NXID Generation - as explained below, each mode will generate a unique NXID per ticket in a different way to prevent ID collision.
- Updating Tickets - Only IP mode and Vulnerability mode support updating tickets.
- Queries - Vulnerability mode uses a different set of queries than those used for IP or Default mode.

NXID Generation

The NXID is a unique identifier used to identify tickets so they may be updated and closed. The format of the NXID depends on the ticketing mode selected.

- **nexpose_identifier_id** - site or tag group being scanned with Nexpose.
- **asset_id** - ID of the asset being scanned.
- **vulnerability_id** - ID of the discovered vulnerability from the database;
- **ip_address** - IP address of the asset being scanned.

The format of the NXID in each mode is as follows:

- **Default Mode:** <nexpose_identifier_id>**d**<asset_id>**d**<vulnerability_id>
- **IP Mode:** <nexpose_identifier_id>**i**<ip_address>
- **Vulnerability Mode:** <nexpose_identifier_id>**v**<vulnerability_id>

Installation & Configuration

Installing the Ticketing Gem

Ticketing Service integrations with Nexpose require the **nexpose_ticketing** Ruby gem that facilitates communication between Nexpose and various ticketing services.
Before installing the nexpose_ticketing Gem, a Ruby interpreter must be installed on the system running the Gem.

The following link shows the different options for installing Ruby in several platforms:

- https://www.ruby-lang.org/en/downloads/

RubyGems is the other prerequisite for using nexpose_ticketing. After successfully installing a Ruby interpreter, install RubyGems.

The following link shows the different options for installing RubyGems in several platforms:

- http://rubygems.org/

After installing Ruby and RubyGems, install **nexpose_ticketing** by opening a command prompt or terminal window with Ruby and RubyGems added to the PATH and run the following command:

- gem install nexpose_ticketing

This command will install the ticketing Gem and all necessary prerequisites.

Configuring the Ticketing Gem

Configuring a Service Integration with Nexpose requires modifying two configuration files from within the gem - the general ticket_service.config file and a ticketing service specific config file.

First, locate the directory of the installed nexpose_ticketing gem. This location varies from system to system. To configure, open the configuration files under the config folder found in the Gem installation:

- Windows: C:\Ruby<version>\lib\ruby\gems\<version>\gems\nexpose_ticketing\lib\nexpose_ticketing\config
- Linux: /var/lib/gems/<version>/gems/nexpose_ticketing/lib/nexpose_ticketing/config

The log-in details are specified within the <service_name>.config file. Service specific configuration information can be found in each individual Service Integration Guide.

The asset groups which are to be scanned, as well as the ticketing mode, are defined in ticket_service.config. Below is a configuration example for the ticket_service.config file.

ticket_service.config

Open ticket_service.config using any text editor and note the following configuration options:

Setting	Description	Sample Value
logging_enabled	Determines if the ticketing service logs information will be written to a text file. Log files are saved in the \log directory.	true
log_level	Sets the log level threshold for output.	info
log_console	Enables logging output to the console.	false
sites	Determines the sites which Nexpose will scan. Leave blank to scan all sites.	- '1' - '4'
severity	Minimum floor severity to report vulnerabilities to the ticket service.	8
site_file_name	Name of the report historical file for sites saved in disk.	last_scan_data.csv

Setting	Description	Sample Value
tag_file_name	Name of the report historical file for tags saved in disk.	tag_last_scan_data.csv
ticket_mode	Determines how tickets are generated in the target ticket service. Three options are currently available: - ‘IP’ creates one ticket per IP address with all vulnerabilities found for the IP address. - ‘Default’ creates one ticket per instance of each vulnerability. - 'Vulnerability' creates one ticket per vulnerability with all affected assets.	Default
max_ticket_length	The maximum character length of a ticket description (-1 is unbounded). It is important to set a max_ticket_length to ensure ticket information is not truncated by the third-party service e.g. JIRA has a maximum description length of ~32000 characters, beyond which the ticket is rejected.	-1
max_title_length	The maximum character length of a ticket title, including ellipsis (‘...’).	100
max_num_refs	The maximum number of references to be included for a single vulnerability. Each vulnerability can have many references.	3
timeout	The ticketing gem timeout in seconds. The number of seconds the gem will wait for a response from Nexpose before exiting.	10800
batch_size	This option breaks the returned vulnerability information (stored in CSV format) into batches of the chosen size for processing and sending to the chosen third party service. Batch size defines the number of rows included in each batch and is limited by both the third-party service and the mode chosen by the user.	5000

Setting	Description	Sample Value
batch_ticket_limit	This option breaks the returned vulnerability information into batches of tickets of the chosen size. Number of tickets per batch is limited by the third-party service	50
close_old_tickets_on_update	For 'I' and 'V' mode, this determines whether the system will close any resolved tickets automatically. Default mode tickets will be closed automatically regardless of this option.	Y
tags	Determines the tags which Nexpose will scan. Leave empty for no tags. Takes priority over sites.	- '1'
nxconsole	Address of the Nexpose server.	127.0.0.1
nxuser	Username of a user within Nexpose with report generation rights.	nxadmin
nxpasswd	Password of above username.	password

Notes:

As the config is a YAML file, indentation within the ticket_service.config file is meaningful. Please refer to the example ticket_service.config file included in the documentation folder - tags should be formatted in the same manner as sites, with each entry taking a new line.

The “batch_size” and “batch_ticket_limit” options work to limit the number of tickets processed and sent to the third-party system in a single batch, as several services will produce an error if too many tickets are in the queue concurrently. For modes producing large volumes of smaller tickets (Default and Vulnerability mode) the batch_size should be higher than the batch_ticket_limit, allowing more tickets to be created per batch. For IP mode, the batch_ticket_limit is to limit the number of tickets created – the batch_size should be much larger (e.g batch_ticket_limit of 50 -> batch_size 5000) as each ticket contains many vulnerabilities for each IP. These options may vary depending on the users Third Party deployment.

When using a mode with a large volume of tickets (Default mode and Vulnerability mode) on a site with a large asset to vulnerability ratio an error may be thrown – “Unable to resolve Hostname”. This issue is Linux specific and can be resolved by increasing the File descriptor limit.

To find the current file descriptor limit, run the command:

- ulimit –n

You can modify the limit for the current session using the command:

- ulimit –n x

where ‘x’ is the number you wish to increase the value to. For a permanent increase, you should modify the /etc/security/limits.conf file.

Please also refer to the specific Helper Integration guides for information on how to configure the Helpers and run the system after configuration.