jamk.fi

Integration of vulnerability scanner reporting

Petteri Sahari

Bachelor's thesis
February 2019
Technology, communication and transport
Degree programme in Information Technology Engineering

jamk.fi

Description

Author(s) Sahari, Petteri	Type of publication Bachelor's thesis	Date 5/2019
		Language of publication: English
	Number of pages 42	Permission for web publication: x

Title of publication
Integration of vulnerability scanner reporting

Degree programme
Data Network Technology

Supervisor(s)
Kotikoski, Sampo; Saarisilta, Juha

Assigned by
Kimmo Puranen / Qvantel Finland Oy

Abstract

The study was assigned by Qvantel Finland Oy. The company needed to automate InsightVM's reporting in form of Jira tickets for the company. This need was based on the requirement to remove workload from the company's employees who had previously created these issues manually.

The study started with hands on research of Jira system and InsightVM. The author also had previous experience with Jira.

Rapid7, the developer of InsightVM software has developed an interface for integrating InsightVM and Jira systems together, which was used to accomplish the wanted outcome in integrating Jira and InsightVM.

The work consisted of researching the methods and the needs to be fulfilled in order to accomplish the already known outcome between the two systems.

The study resulted in automatic creation of tickets based on the reports that InsightVM scans provided. The conclusion was that the need was fulfilled as the tickets were automatically created with the integration.

Keywords/tags (subjects)
InsightVM, Jira, Atlassian, Workflow, Scheme, Issue Type, Issue, Ticket, Report, Screen, Automation, Reporting, Report, Vulnerability Management, Rapid7, Scanner

Miscellaneous (Confidential information)

jamk.fi

Description

Tekijä(t) Sahari, Petteri	Julkaisun laji Opinnäytetyö, AMK	Päivämäärä 5/2019
		Julkaisun kieli: Englanti
	Sivumäärä 42	Verkkojulkaisulupa myönnetty: x

Työn nimi
Integration of vulnerability scanner reporting

Tutkinto-ohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
Sampo Kotikoski, Juha Saarisilta

Toimeksiantaja(t)
Qvantel Finland Oy / Kimmo Puranen

Tiivistelmä

Lähtökohtana työlle toimi tarve saada InsightVM-tikettien luonti automatisoitua Jira-järjestelmään, jotta Qvantel Finland Oy:n työntekijöiden ei tarvitsisi tehdä tikettejä manuaalisesti käsin.

Tarkoitus oli siis saada InsightVM:n tietoturvahaavoittuvuusskannauksien tulosten perusteella tehtävät raportit automaattisesti raportoitua tikettien muodossa Jira-järjestelmään.

Työn toteuttaminen aloitettiin tutkimalla Jira- ja InsightVM -järjestelmiä ja hyödyntämällä työn toteuttajalla olevaa ennalta kerättyä kokemusta Jira-järjestelmästä. Lopputulos järjestelmien yhteensopivuudesta ja toimivuudesta oli jo ennalta tiedossa eli työssä selvitettiin, miten ennalta tiedettyyn lopputulokseen päästäisiin.

Rapid7, InsightVM-ohjelmiston kehittäjä, on luonut rajapinnan järjestelmien integroimiseksi, jota hyödynnettiin työn toteutuksessa.

Tuloksena oli tikettien automaattinen luonti Jira-järjestelmään, kuten oli tarkoituskin. Toimeksiantaja hyötyi työstä siten, että manuaalista työtä tikettien luonnissa ei enää tarvita, ja täten integraatio tuo rahallista arvoa työnantajalle.

Avainsanat (asiasanat)
InsightVM, Jira, Atlassian, Workflow, Scheme, Issue Type, Issue, Ticket, Report, Screen, Automation, Reporting, Report, Vulnerability Management, Rapid7, Scanner

Muut tiedot

Contents

Figures

Tables

Terminology

Issue	A ticket that appears in Jira for users
Workflow	A way of work for solution management
Project	A way to control access to issues per user
Security Level	A way to restrict visibility of issues inside a project
Scheme	Property used to bind other configurations
Field Configuration	Used to make fields mandatory inside a project
Asset	A server environment inside a data center
Host	An individual server inside an asset
Condition	A Condition for transitioning in workflow
Trigger	A set action to occur when transitions take place in work-flow
Role	Used to specify user’s role inside a project
Group	Can be used to manage user’s permissions inside a project
Post function	Will occur in workflow after transition
Validator	Can be used to validate fields in workflow transition
Screen	A view shown to user when viewing issues

1 Introduction

This thesis aims to explain and discover whether it is possible to integrate JIRA Software with InsightVM vulnerability scanner.

Jira Software (JIRA) is an issue tracking product developed by Atlassian. JIRA can be used to assist agile teams to plan and develop their projects, e.g. develop a program, and start listing bugs found in the program, as issues in the JIRA, and then start fixing them visibly so anyone included in the project has visibility of the current status of the bugs and their fixes (JIRA in a nutshell demo video, n.d.).

The necessary configurations to Jira will be done before proceeding with the integration, as Jira is in a key role for making the vulnerabilities visible to the employees whose job is needed to fix them.

InsightVM Vulnerability management (InsightVM) is a tool that can be used to scan, i.e. to search for vulnerabilities in the computer server environments that are used to host webservices such as a company's web shop.

The integration should allow the InsightVM to provide vulnerability scan reports to be automatically created after the vulnerability scan has occurred.

Integrating InsightVM with JIRA will greatly reduce the workload of the company's employees who have to create tickets in JIRA manually based on the reports provided by InsightVM.

InsightVM will be integrated into self-hosted Jira Software to create automatic reports of found security related vulnerabilities in the systems at which the InsightVM security scanner is targeted.

2 Research methodology and goals

The research method chosen for this thesis was constructive research, as the wanted outcome was already known before the thesis was started. It was just a matter of researching how to accomplish the wanted outcome which was to automate issue creation in Jira based on InsightVM's vulnerability reports; in short, to make the vulnerabilities automatically appear as issues in Jira for patching (Pasian, 2015).

The goal for this thesis is that the integration would work successfully between the two systems and that this thesis could be used as an example when configuring new projects in either one of these two systems.

3 Jira

3.1 Jira in a nutshell

Jira Software is an issue management system or in other words, a ticketing system. To a user who has never used or heard about anything similar, it can be overly simplified as follows: a company has a list of issues company employees need to be able to track. In order to track those issues, the issues must be created into JIRA to which all sorts of different issue types can be configured. E.g. Matt wants to report a bug found in software. Matt picks an issue type "Bug" when creating a ticket to Jira. Then after creating a ticket with issue type "Bug", the issue is assigned to a developer responsible for fixing bugs. If the issue contains all necessary info to reproduce the issue, the person fixing the bugs can proceed fixing the bug. After the bug has been patched, the developer will comment that it is ok; the bug was fixed and perhaps it is also mentioned that reason x was the root cause for the bug to happen. If the developer does not have enough information to reproduce the issue, the issue can be assigned back to Matt who originally reported on the bug and created the issue.

Jira provides multiple ways to illustrate and to track the progress of ongoing issues, for example via email notifications and via dashboards configured by the users (How do you explain Jira?).

In fact, as far as customizability goes, JIRA can be used as a fulfledged company Service Desk contact point as well if JIRA Service Desk is installed on top of the Jira Software (What is Jira Service Desk?).

If Jira is configured properly, almost all work requests inside a medium-sized to large IT organization should and can be done via a ticketing system such as Jira in order to keep track on what is being currently developed or worked on. Jira also has a large community backing up the userbase and easing up the problems that might occur or might have occurred in the past.

Carrying out all necessary changes listed in this thesis requires that the user has JIRA administrator privileges.

3.2 Projects

Projects are used in JIRA to separate the issues from each other, for example Project "Customer X" does not mix its issues with project "Customer Y". It also opens a possibility to customize the following to be individually configured per project (Project screens, schemes and fields):

- Workflow schemes
- Issue type schemes
- Issue type screen schemes
- Issue security schemes
- Field configuration schemes
- Screen schemes
- Notification schemes
- Permission schemes

When creating these configurations to JIRA, it is recommended for easier usage to name the configurations to match the project naming, for example: A field configuration scheme in Project X could be named Project X **Field Configuration Scheme**. Using this naming policy will greatly reduce the risk of associating wrong schemes to wrong projects (Defining a project).

3.3 Schemes

Each feature in JIRA has a scheme that is used to add certain features into a certain issue type, for example an issue type “Epic” can have its own screen scheme, which would contain a screen to be associated with the issue type.

All schemes and features can be shared between projects; however, it is not advised to do so, since if there becomes a need later to modify one project’s schemes, the changes will then take effect in all other projects where the same schemes are used as well (Atlassian JIRA Configuration Tutorial: JIRA Schemes (Part 1)).

Just like all every other project-specific configuration, this should be named logically to avoid confusion onwards, when making changes to the projects.

3.4 Workflows

In Jira, there are many different features to customize according to the project’s needs, which will provide the most fluid way of working with JIRA inside the project. Workflows are used to manage phases of the ongoing issue, so that when a ticket is created in a certain create issue screen, it will transition to a second status.

Based on the user interaction in the issue, Jira will move to another step until the issue has been resolved. In some cases, the workflows can also include a step to re-open the issues, depending yet again on the project’s needs (Building an awesome Jira workflow: concepts and examples).

Screens can also be added into transitions, in case e.g. a new field value should be added to the issue in the middle of the workflow when an issue has already been created (Make every transition count; Assignees).

Workflows can also contain post functions, validators, conditions, properties, triggers and global transitions to create certain configurations to be possible.

3.4.1 Post functions

Post functions can be added to any step of the workflow to make certain specific tasks occur when the workflow transition is done. An example of a post function could be adding a value to an issue after the next transition has occurred.

There are five essential post functions that need to be included in each workflow step to make the workflows function. The list is as follows (Advanced workflow configuration):

- Set issue status to the one that the next step in workflow has (example: change status from Open to In-progress)
- Add a comment to the issue if one is provided during the transition (requires a separate screen to be added if the comment adding is wanted to be done in transition itself)
- Update the issue history to keep the issue history log up-to-date (Cannot be modified, so if issue contains confidential info by accident, the issue must be deleted completely, and a new one created without the info)
- Re-index the issue to keep the database in sync with the issues
- Start an event that can be then be processed by listeners

3.4.2 Indexing of Jira

It is worth noting that if JIRA faces errors, it supports per-project re-indexing as well as whole JIRA re-indexing. The whole JIRA re-index run can take a while depending on the amount and size of the database.

3.4.3 Validators

Validators can be used to check that specified fields have a value added or that it has been changed from the original value. This works much like *Field configurations*, however, this instead can be used in each step of the workflow individually, so for example: A field can be empty when the issue is created; however, not after a certain step in the workflow, it can be made mandatory in the middle of an ongoing issue.

3.4.4 Conditions

Conditions are much like *Validators* they can be used to make some configurations in the middle of a workflow. Conditions can be used to control who should be able to execute things at certain step of the workflow, for example:

- only allow issue reporter to transition the issue to next status
- only allow users with a certain role or group membership to execute transition
- only allow code execution if this code has or has not been committed against the issue

If condition is not met, in other words if condition fails, the user cannot transition issue to another status or step of the workflow. Failing the condition would make the next step of the workflow invisible in the View issue screen and hence make it not possible to transition further until the condition is met (Building an awesome Jira workflow: concepts and examples).

3.4.5 Properties

Properties are settings that can be configured inside each step of the workflow. An example of a property could be that a resolution status could be cleared from the workflow, when the issue is transitioned from Closed-status to Open-status; this sort of transition is usually named “Reopen”, as it is used to re-open the issue (Building an awesome Jira workflow: concepts and examples).

3.4.6 Triggers

Triggers are a tool to be used to sync Jira between development tools, for example Bitbucket or GitHub, just like writing this thesis is about integrating InsightVM to work with Jira to reduce the workload when creating scan reports.

Triggers can be used to automatic updating of the issues, so developers do not have to manually update e.g. the statuses of each code commitment. Triggers can be used to automatically transition issues when development related events occur on the development tool.

3.4.7 Global Transitions

Global Transitions are transitions in workflow that can be accessed in the issue from any step of the workflow. This means that no matter what the current status is in the issue, it can be transitioned to this status, an example status could be “On-hold”. Global transitions cause a problem: when going to this step on the workflow, it is possible to skip steps, which might not be something that the project wants if there are mandatory steps in the workflow to go through.

3.5 Screens

Screens are used in JIRA to define what fields are showing up in the issue when it is opened. There are three types of screens:

- Create Issue Screen
- View Issue Screen
- Edit Issue Screen

The first screen is only used when creating the issue, so when clicking on the “Create” button in JIRA, it will open a new window called a *screen*. After the fields are filled in, and the user has created the issue, the user will transition to the *view issue screen*.

If the issue or its contents needs to be edited afterwards, the user can click on the “Edit” option to open the *Edit Issue Screen.* The editing can also be limited via field configurations, so that e.g. some values cannot be edited afterwards.

It is also possible to add any screen to appear during workflow transition, if for example a new field would need to be filled in at a certain step of the workflow.

3.6 Field configurations

Field configurations can be used to define fields to be mandatory, so that when a user creates issues, it is possible to force the user to fill in the fields that are mandatory. Defining fields as mandatory can be useful in situations, where it is important that a field called “Environment” is not left empty. If the user makes the field mandatory, the “Create issue screen” will not let the user to proceed creating the issue until the field has a value.

It will ease up understanding and shorten the resolve time in the issues as the people working with the issue have at the very least some base level of information where to start when working to resolve issues.

3.7 Issue Types

Issue types are used to describe different issues. Each issue type can have its own configurations inside each project as follows: screens, workflows, field configurations. Example issue types inside a project could be: Bug, Task, Sub-task, Epic, Test Case.

Issue types will be associated with the projects Issue Type scheme, granting them visibility in that specific project. Each issue type can have and should have its own icon for easier separation from other issues types inside the project.

3.8 Security levels

Security levels are a feature used inside projects to limit the visibility of issues inside the project, so if it is necessary to limit the visibility on Task issues inside Project Y, the issues can be then shown to users inside the project with necessary privileges only. Security levels privileges can be mapped to either project roles, groups, or to individual users.

3.9 Project Roles

Project roles work similarly to groups but are only project specific and can be only granted via the project configuration settings. Project Role names are global in Jira, but the roles that are available in each project can be associated individually. Project roles will become available on the project once at least one user or group has been assigned to a project role.

There are a few default roles in Jira, these roles are: *Users*, *Developers* and *Administrators*. Granting a project leadership to a user does not grant any roles, so if for example the project lead needs project admin privileges, they should be added to have Administrators project role (Managing project roles)

3.10 Project permissions

Project permissions are policies that define what the user can do inside a Jira project. In each permission should be carefully evaluated who should have the access to each permission, so that accidents in the issues can be minimized.

These permissions can be mapped to:

- users
- groups
- project roles
- issue roles
- anyone “anonymous”
- multi user picker custom field
- multi group picker custom field

There are quite many different permissions in Jira to be granted, so it is possible to limit the functionalities of each user in each project in a good manner (Managing project permissions, n.d.). The following project permissions exist in JIRA (See Table 1):

Table 1 Project permissions

Administer project
Browse project
Manage Sprints
View development tools
View workflow
Assign issues
Assignable user
Close issues
Create issues
Delete issues
Edit issues
Link issues
Modify reporter
Move issues
Resolve issues
Schedule issues
Set issue security (security level)
Transition issues (change status to another)
Manage watcher list
View voters and watchers
Add comments
Delete all comments
Delete own comments
Edit all comments
Edit own comments
Create attachments
Delete all attachments
Delete own attachments
Work on issues
Delete all worklogs
Delete own worklogs
Edit all worklogs
Edit own worklogs

These permissions should be carefully mapped to groups or project roles accordingly. If for example the project lead ability to add components inside the project is to be granted, there should be at least the following project permissions: Administer projects, Browse projects. In practice, those two roles are not nearly enough to start working with the project but give a basic idea what can be granted to make certain functionality available for that specific user (Managing project permissions).

4 Rapid7 InsightVM

Security issues are known to be a common problem in the whole IT industry; keeping the services secured is a huge task to deal with so it is problematic to get reports done and then start the system patching. The way InsightVM works in practice is explained as follows (How it works; Streamline Vulnerability Remediation Workflows.):

1. InsightVM evaluates the risks of each system through a scan or agent-based assessment.
2. The collected vulnerability data is handled to each individual host included in the Insight platform.
3. InsightVM remediation project has configurations to define what scope of issues need to be remedied, by whom and by when. (For example, Windows based systems should be patched by the corresponding system owner.)
4. When configured with Jira, the tickets are created automatically and will be assigned to the corresponding system owners to be handled.
5. Once the issue has been handled, the remediation project in InsightVM shows that the vulnerabilities are marked as fixed.

When the next scan occurs, InsightVM confirms that the fix has been applied, and the case on that vulnerability has been patched

4.1 InsightVM Vulnerability Management Use Case

Vulnerabilities can be found manually; however, is very cumbersome. Using automated scanning software will help a great deal, as it has the most common vulnerabilities listed and its database is constantly updated by Rapid7 (Vulnerability Database). Creating reports about the found vulnerabilities will make it much easier to the company's IT personnel to fix the known vulnerabilities as they are listed in JIRA, which will also help company personnel to check the current status of ongoing vulnerability fixes.

Keeping services and servers secured from prying eyes is mandatory for any business or service that hopes to have any wish to not be broken into. In the worst case, if the servers are not secure enough, a person from around the globe could potentially attack the website and break it, which in turn would result in the company not being able to function as the channel that is used to sell products and maintaining the business would no longer work (Playstation Network and Qriocity Outage FAQ).

Another horror scenario would be that because of the vulnerable environments, a so called "Black Hat hacker" would break in to the systems containing a company's customers´ personal information such as home addresses, private phone numbers, location data, social security numbers which should never be allowed to be handed to wrong hands. Black Hat hackers could then proceed selling or publishing the information to the highest bidder – or even free because of personal political reasons (What is a Black-Hat Hacker; Hacking is a business).

4.2 Asset groups & filtering

InsightVM contains information about the hosts where it has the agents applied to, so the scans can be run on these hosts. These hosts can then be filtered by asset groups in InsightVM platform. This makes it possible to scan these assets on group basis individually depending on the configurations in either the filter itself and or by listing certain hosts in certain asset groups (Working with asset groups).

5 Requirements for integration

Requirements for integration InsightVM with JIRA has the following requirements:

- Working connectivity between InsightVM and Jira servers
- A service account in Jira that is used to create the tickets in Jira
- A project with necessary configurations to create the tickets in Jira
- A remediation project inside InsightVM

This list does not include the needs of users need to have access to both systems in order to make the required configurations and only takes into account when the user has assumed required accesses to both systems with enough privileges.

6 Configurations

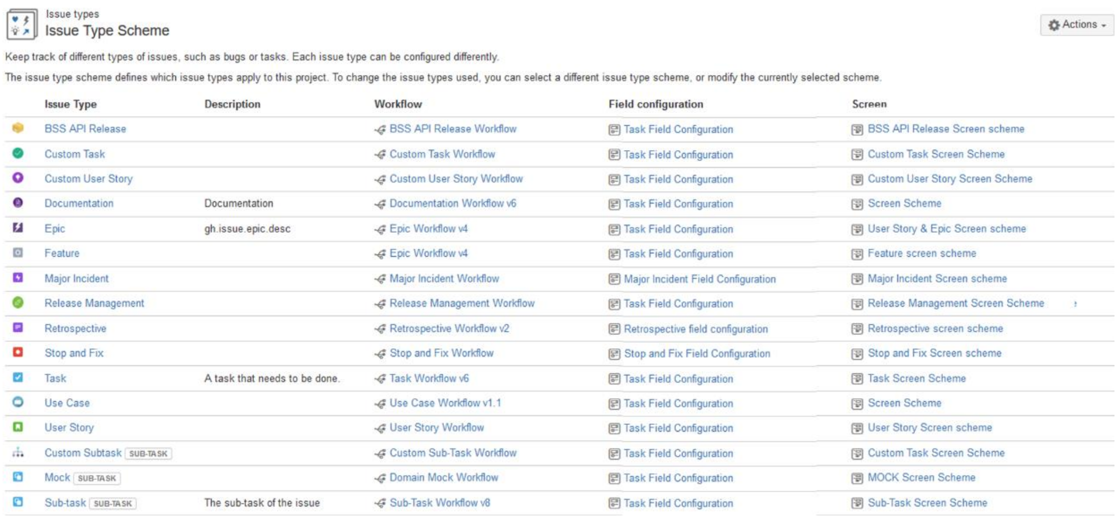
The necessary Jira and InsightVM configurations are provided in this chapter, so that the reader gets an idea what sort of steps are needed to accomplish the wanted outcome.

6.1 Jira configurations

In this solution, a JIRA Software version 7.3.3 has already been installed to Oracle Linux servers and the work includes configurations to be done in order to get the integration working in JIRA.

6.1.1 Issue type scheme

An issue type scheme will be configured according to the project's needs; in this case the scheme has 16 different issue types, one of which is named *Sub-task* as seen in Figure 1.



Issue Type Scheme

Keep track of different types of issues, such as bugs or tasks. Each issue type can be configured differently.

The issue type scheme defines which issue types apply to this project. To change the issue types used, you can select a different issue type scheme, or modify the currently selected scheme.

Issue Type	Description	Workflow	Field configuration	Screen
BSS API Release		BSS API Release Workflow	Task Field Configuration	BSS API Release Screen scheme
Custom Task		Custom Task Workflow	Task Field Configuration	Custom Task Screen Scheme
Custom User Story		Custom User Story Workflow	Task Field Configuration	Custom User Story Screen Scheme
Documentation	Documentation	Documentation Workflow v6	Task Field Configuration	Screen Scheme
Epic	gh.issue.epic.desc	Epic Workflow v4	Task Field Configuration	User Story & Epic Screen scheme
Feature		Epic Workflow v4	Task Field Configuration	Feature screen scheme
Major Incident		Major Incident Workflow	Major Incident Field Configuration	Major Incident Screen scheme
Release Management		Release Management Workflow	Task Field Configuration	Release Management Screen Scheme
Retrospective		Retrospective Workflow v2	Retrospective field configuration	Retrospective screen scheme
Stop and Fix		Stop and Fix Workflow	Stop and Fix Field Configuration	Stop and Fix Screen scheme
Task	A task that needs to be done.	Task Workflow v6	Task Field Configuration	Task Screen Scheme
Use Case		Use Case Workflow v1.1	Task Field Configuration	Screen Scheme
User Story		User Story Workflow	Task Field Configuration	User Story Screen scheme
Custom Subtask SUB-TASK		Custom Sub-Task Workflow	Task Field Configuration	Custom Task Screen Scheme
Mock SUB-TASK		Domain Mock Workflow	Task Field Configuration	MOCK Screen Scheme
Sub-task SUB-TASK	The sub-task of the issue	Sub-Task Workflow v8	Task Field Configuration	Sub-Task Screen Scheme

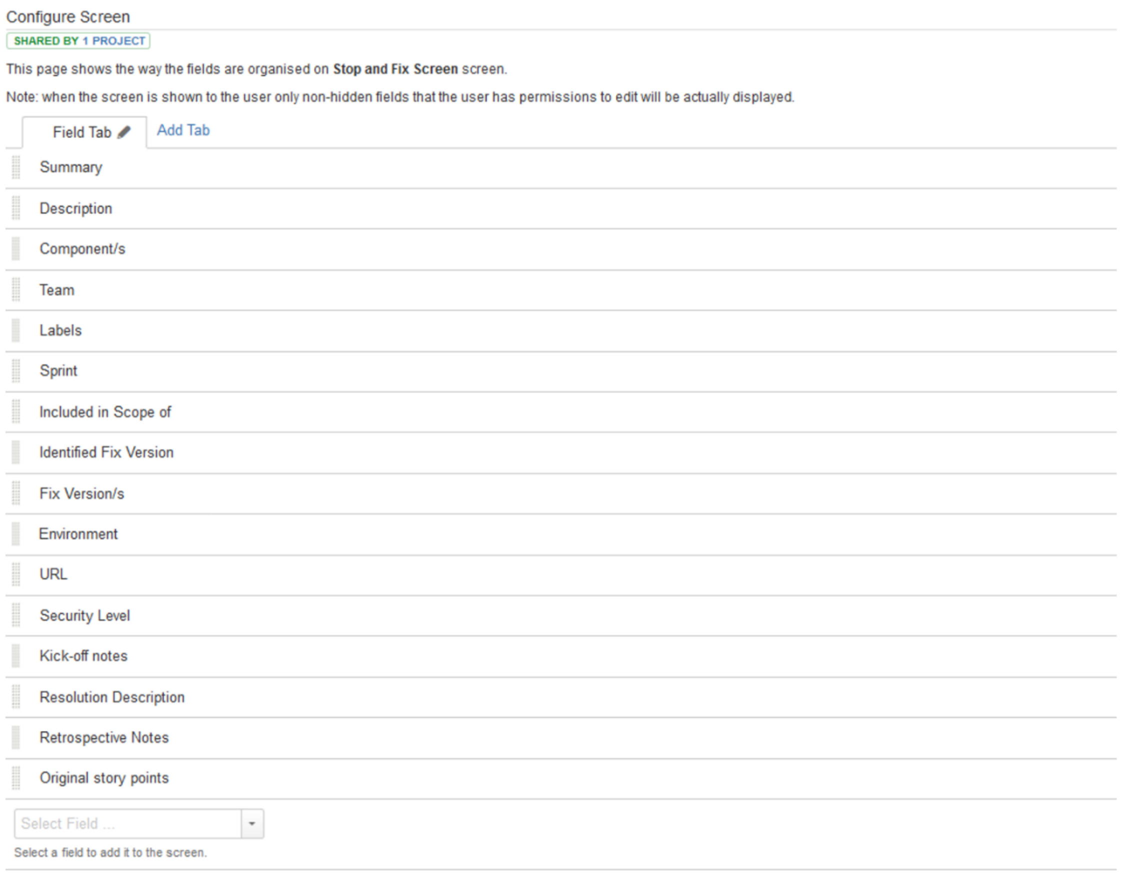
Figure 1. Issue type scheme

Sub-tasks are used for any issue type, so that if there is one big main issue with many smaller issues, the smaller ones need to be fixed in order to get the big picture working, since they are useful steps to be included under the main issue.

6.1.2 Screen configuration

Screen configuration consists of all fields needed for the issues to show up correctly. Usually the screens contain at least the following fields: *Summary, Description, Assignee, Reporter* and *Priority.* Other fields can be added as well according to the needs of a project as illustrated in

Figure 2. Stop and Fix screen



Configure Screen

SHARED BY 1 PROJECT

This page shows the way the fields are organised on **Stop and Fix Screen** screen.

Note: when the screen is shown to the user only non-hidden fields that the user has permissions to edit will be actually displayed.

- Summary
- Description
- Component/s
- Team
- Labels
- Sprint
- Included in Scope of
- Identified Fix Version
- Fix Version/s
- Environment
- URL
- Security Level
- Kick-off notes
- Resolution Description
- Retrospective Notes
- Original story points

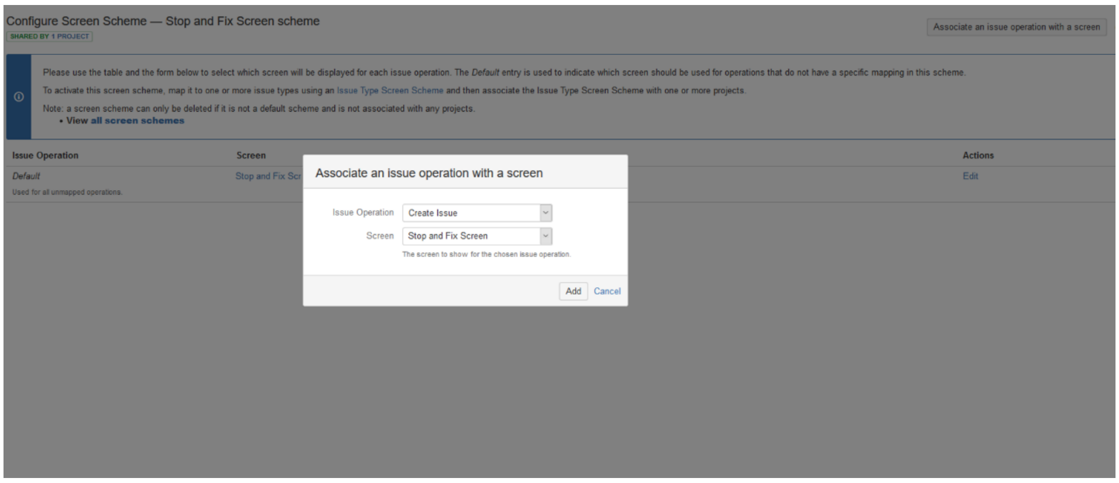
Select a field to add it to the screen.

Figure 2. Stop and Fix screen

Each field can be differently configured; some fields, for example, have pre-filled values from which the user can pick the needed value(s) depending on if the field is a multi select, or single select list. Fields *reporter* and *assignee* are *“user picker”*type fields. *User picker* field allows the end user to pick another user that has access to the system, to be added as a value to the field. For example, User Red is responsible for fixing issues that are related to Issue Red, then User Green can assign the issue directly to User Red to be solved.

6.1.3 Screen scheme configuration

Screen schemes are used to specify which Issue operations should have each screen, so say for example that when creating the issue *Create issue* -screen pops up for the end user as can be seen in Figure 3.



Configure Screen Scheme — Stop and Fix Screen scheme

SHARED BY 1 PROJECT

Associate an issue operation with a screen

Please use the table and the form below to select which screen will be displayed for each issue operation. The *Default* entry is used to indicate which screen should be used for operations that do not have a specific mapping in this scheme.

To activate this screen scheme, map it to one or more issue types using an Issue Type Screen Scheme and then associate the Issue Type Screen Scheme with one or more projects.

Note: a screen scheme can only be deleted if it is not a default scheme and is not associated with any projects.
- View **all screen schemes**

Issue Operation	Screen	Actions
Default Used for all unmapped operations.	Stop and Fix Scr	Edit

Associate an issue operation with a screen

Issue Operation: Create Issue

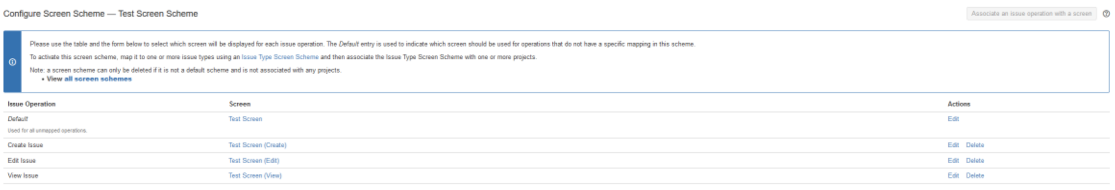
Screen: Stop and Fix Screen

The screen to show for the chosen issue operation.

Add Cancel

Figure 3. Screen Scheme configuration

When creating certain issues, some fields should be available, but if we want that for example the value of field *notes* should not be possible to be edited after the issue has been created, then we want to add the screen with corresponding field only in the *Create Issue* screen but not in *View Issue* and *Edit Issue* screens as shown in Figure 4.



Configure Screen Scheme — Test Screen Scheme

Associate an issue operation with a screen

Please use the table and the form below to select which screen will be displayed for each issue operation. The *Default* entry is used to indicate which screen should be used for operations that do not have a specific mapping in this scheme.

To activate this screen scheme, map it to one or more issue types using an Issue Type Screen Scheme and then associate the Issue Type Screen Scheme with one or more projects.

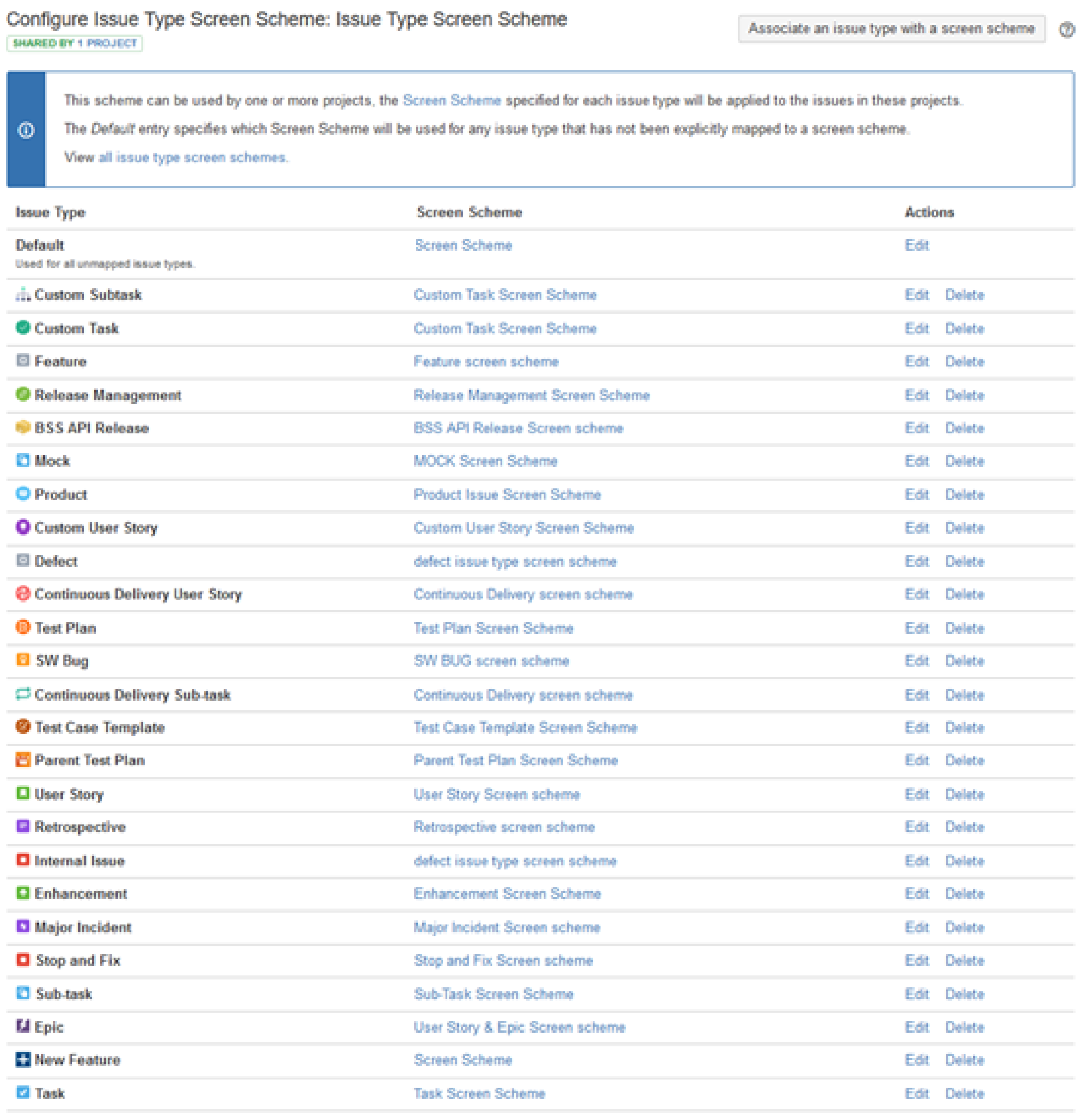
Note: a screen scheme can only be deleted if it is not a default scheme and is not associated with any projects.
- View **all screen schemes**

Issue Operation	Screen	Actions
Default Used for all unmapped operations.	Test Screen	Edit
Create Issue	Test Screen (Create)	Edit Delete
Edit Issue	Test Screen (Edit)	Edit Delete
View Issue	Test Screen (View)	Edit Delete

Figure 4. Screen screen scheme

6.1.4 Issue type screen scheme

Issue type screen scheme is used inside the project to specify which issue types use which individual screen scheme. These can be shared between projects; however, for the purposes of this thesis an issue type screen scheme is used that is separate from all other projects. This is illustrated in Figure 5.



Configure Issue Type Screen Scheme: Issue Type Screen Scheme

SHARED BY 1 PROJECT

Associate an issue type with a screen scheme

This scheme can be used by one or more projects, the Screen Scheme specified for each issue type will be applied to the issues in these projects.
The *Default* entry specifies which Screen Scheme will be used for any issue type that has not been explicitly mapped to a screen scheme.
View all issue type screen schemes.

Issue Type	Screen Scheme	Actions
Default Used for all unmapped issue types.	Screen Scheme	Edit
Custom Subtask	Custom Task Screen Scheme	Edit Delete
Custom Task	Custom Task Screen Scheme	Edit Delete
Feature	Feature screen scheme	Edit Delete
Release Management	Release Management Screen Scheme	Edit Delete
BSS API Release	BSS API Release Screen scheme	Edit Delete
Mock	MOCK Screen Scheme	Edit Delete
Product	Product Issue Screen Scheme	Edit Delete
Custom User Story	Custom User Story Screen Scheme	Edit Delete
Defect	defect issue type screen scheme	Edit Delete
Continuous Delivery User Story	Continuous Delivery screen scheme	Edit Delete
Test Plan	Test Plan Screen Scheme	Edit Delete
SW Bug	SW BUG screen scheme	Edit Delete
Continuous Delivery Sub-task	Continuous Delivery screen scheme	Edit Delete
Test Case Template	Test Case Template Screen Scheme	Edit Delete
Parent Test Plan	Parent Test Plan Screen Scheme	Edit Delete
User Story	User Story Screen scheme	Edit Delete
Retrospective	Retrospective screen scheme	Edit Delete
Internal Issue	defect issue type screen scheme	Edit Delete
Enhancement	Enhancement Screen Scheme	Edit Delete
Major Incident	Major Incident Screen scheme	Edit Delete
Stop and Fix	Stop and Fix Screen scheme	Edit Delete
Sub-task	Sub-Task Screen Scheme	Edit Delete
Epic	User Story & Epic Screen scheme	Edit Delete
New Feature	Screen Scheme	Edit Delete
Task	Task Screen Scheme	Edit Delete

Figure 5. Issue type screen scheme

6.1.5 Workflow Configuration

Workflows are considered carefully before they are implemented to the issue types, as this affects how the end user handles the issues. An example would be that when user is creating an issue, after setting the values to *Create Issue Screen* then the issue will transition to the next status of the workflow *Waiting for kick-off*. Next, the user proceeds on clicking "Kick-off" button on the *View issue screen*, then the issue transitions to the status "In Progress" as demonstrated in Figure 6.

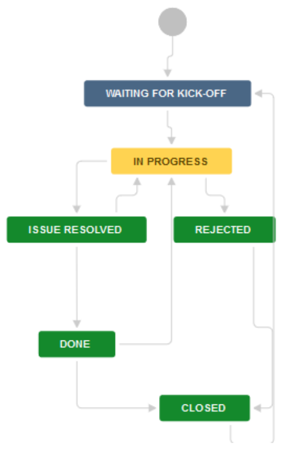


Figure 6. Stop and Fix workflow

After the issue has been set to "In progress", based on user interaction the workflow it is transitioned to the next status. Each box with a color represents a status in the workflow, and the labels between the statuses are *transitions*. Transitions show up to the end user as buttons when viewing the issues. Edit workflow view in Figure 7 shows all the properties that can be configured in the workflows.

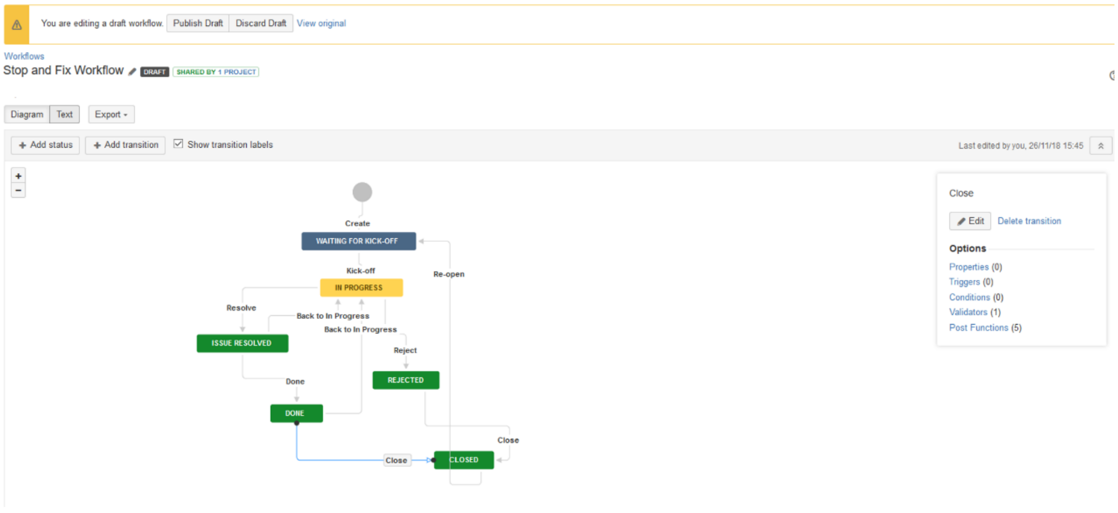
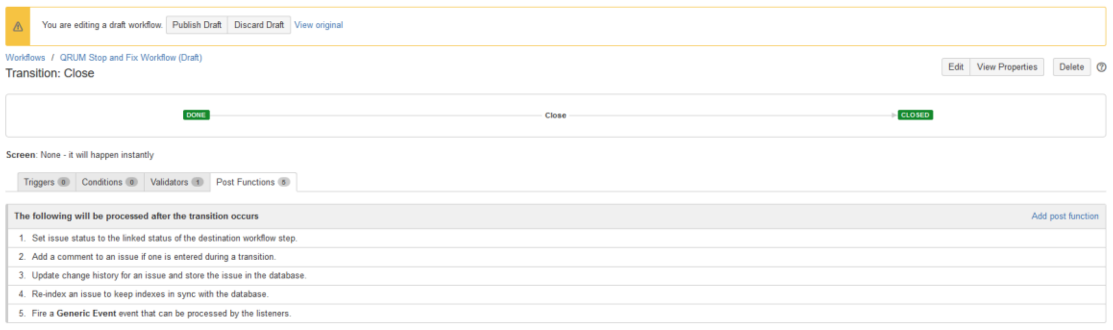


Figure 7. Stop and Fix workflow - Edit view

After the user has created the basic workflow diagram, the workflow must be published to become active. Post functions as many other properties can be configured individually to each step of the workflow, for example if the status of the workflow needs to be changed with the transition from *Done* to *Closed*, the transition needs a post function “Set issue status to the linked status of destination workflow step” as shown in Figure 8.



Transition: Close

Screen: None - it will happen instantly

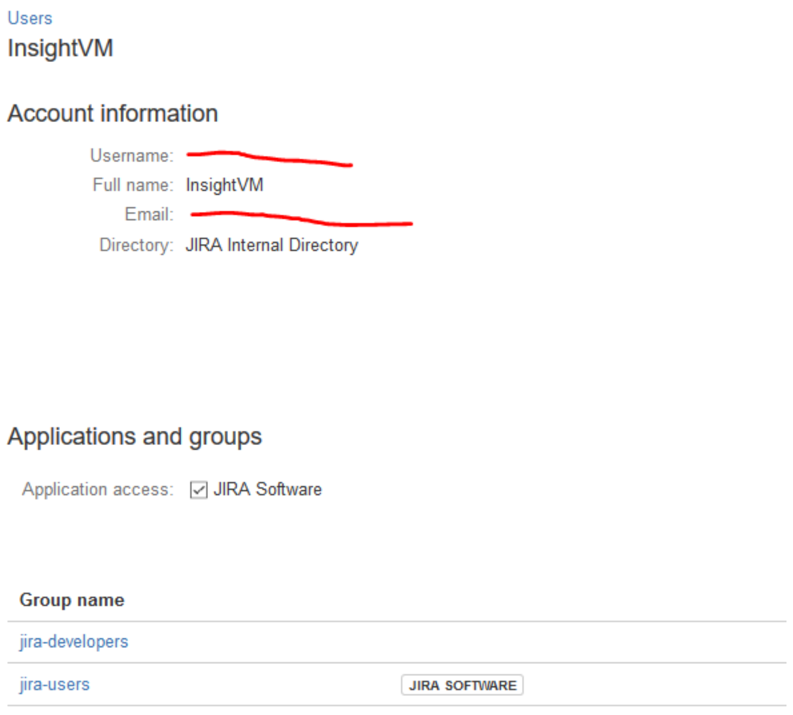
The following will be processed after the transition occurs

1. Set issue status to the linked status of the destination workflow step.
2. Add a comment to an issue if one is entered during a transition.
3. Update change history for an issue and store the issue in the database.
4. Re-index an issue to keep indexes in sync with the database.
5. Fire a **Generic Event** event that can be processed by the listeners.

Figure 8. Post Functions view

6.1.6 Service account for InsightVM

A service account needs to be created for InsightVM service in Jira for it to create issues. Depending on the project where the issues need to be created, the service account needs to have necessary roles as illustrated in Figure 9. The username and email address are hidden from the figure for security reasons.


Users
InsightVM

Account information

Username:	[redacted]
Full name:	InsightVM
Email:	[redacted]
Directory:	JIRA Internal Directory

Applications and groups

Application access: ☑ JIRA Software

Group name	
jira-developers	
jira-users	JIRA SOFTWARE

Figure 9. InsightVM Service account

6.2 InsightVM Configurations

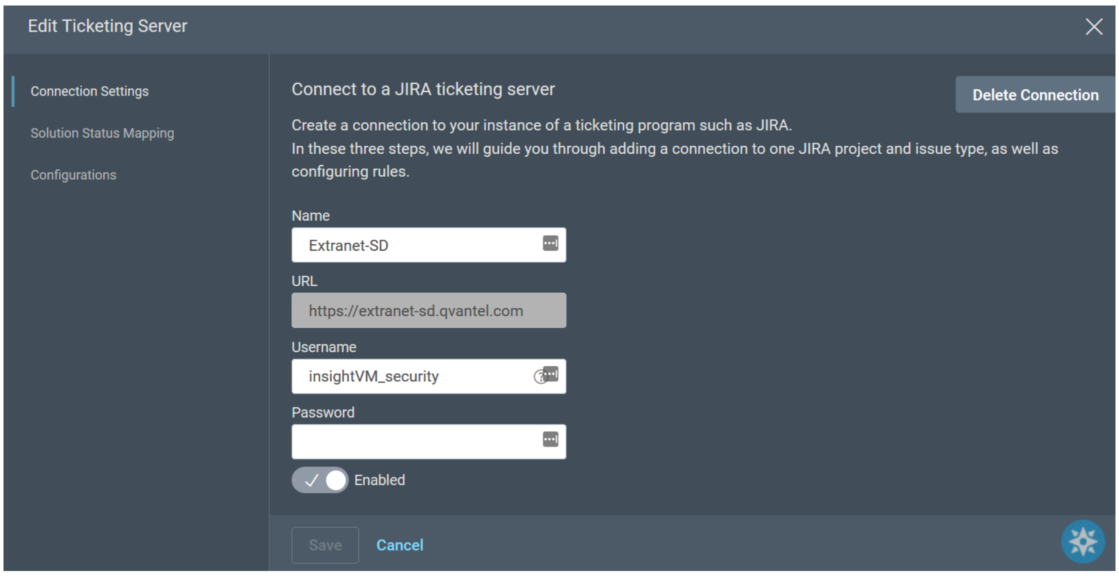
InsightVM requires a few variables to be configured in order to make the Jira connectivity to work. These configurations consist of the following configurations:

- Adding the Service account to InsightVM's configuration
- A Remediation project to be configured with needed asset groups used as filtering factor

6.2.1 Configuring connection to JIRA in InsightVM

Integration to InsightVM starts with creating a ticketing server connection. For the connection, the following environment variables need to be configured as shown in Figure 10:

- Username to be used in Jira
- Password for the same username
- URL to access the environment
- Name for the Ticketing server service to distinguish it from others if multiple different ones are used



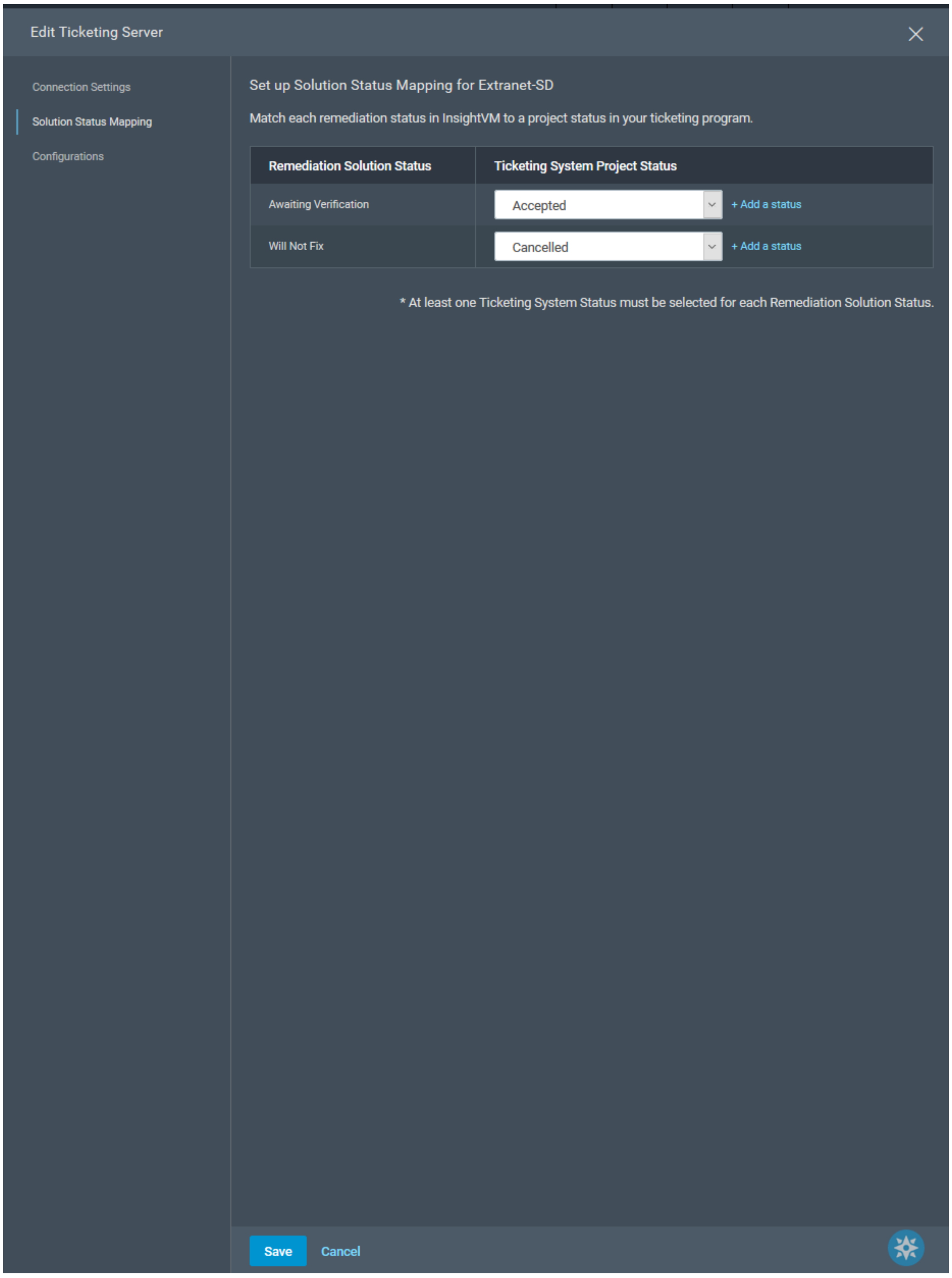
Connect to a JIRA ticketing server

Create a connection to your instance of a ticketing program such as JIRA.
In these three steps, we will guide you through adding a connection to one JIRA project and issue type, as well as configuring rules.

Name: Extranet-SD
URL: https://extranet-sd.qvantel.com
Username: insightVM_security
Password:

Figure 10. Connection to JIRA ticketing server

After the initial connection settings have been created, InsightVM will prompt for solution status mapping, which means that if an issue is transitioned to a mapped status, then InsightVM will decide on actions based on the statuses as illustrated in Figure 11.



Edit Ticketing Server

Set up Solution Status Mapping for Extranet-SD

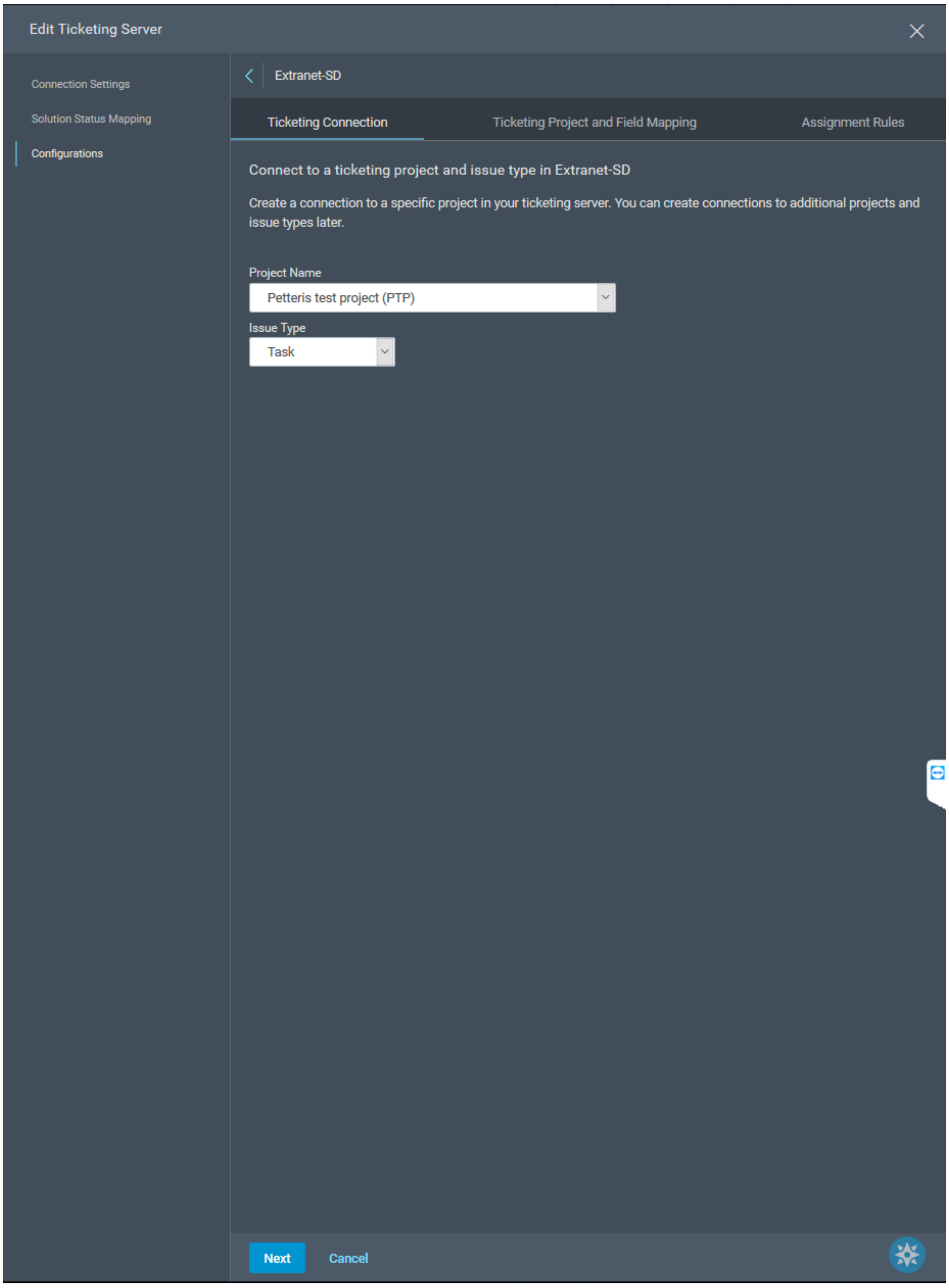
Match each remediation status in InsightVM to a project status in your ticketing program.

Remediation Solution Status	Ticketing System Project Status
Awaiting Verification	Accepted
Will Not Fix	Cancelled

* At least one Ticketing System Status must be selected for each Remediation Solution Status.

Figure 11. Solution Status mapping

The next part would be to configure the project from Jira to the InsightVM to be used, which is illustrated in Figure 12.


Edit Ticketing Server

Connection Settings
Solution Status Mapping
Configurations

Extranet-SD

Ticketing Connection | Ticketing Project and Field Mapping | Assignment Rules

Connect to a ticketing project and issue type in Extranet-SD

Create a connection to a specific project in your ticketing server. You can create connections to additional projects and issue types later.

Project Name: Petteris test project (PTP)

Issue Type: Task

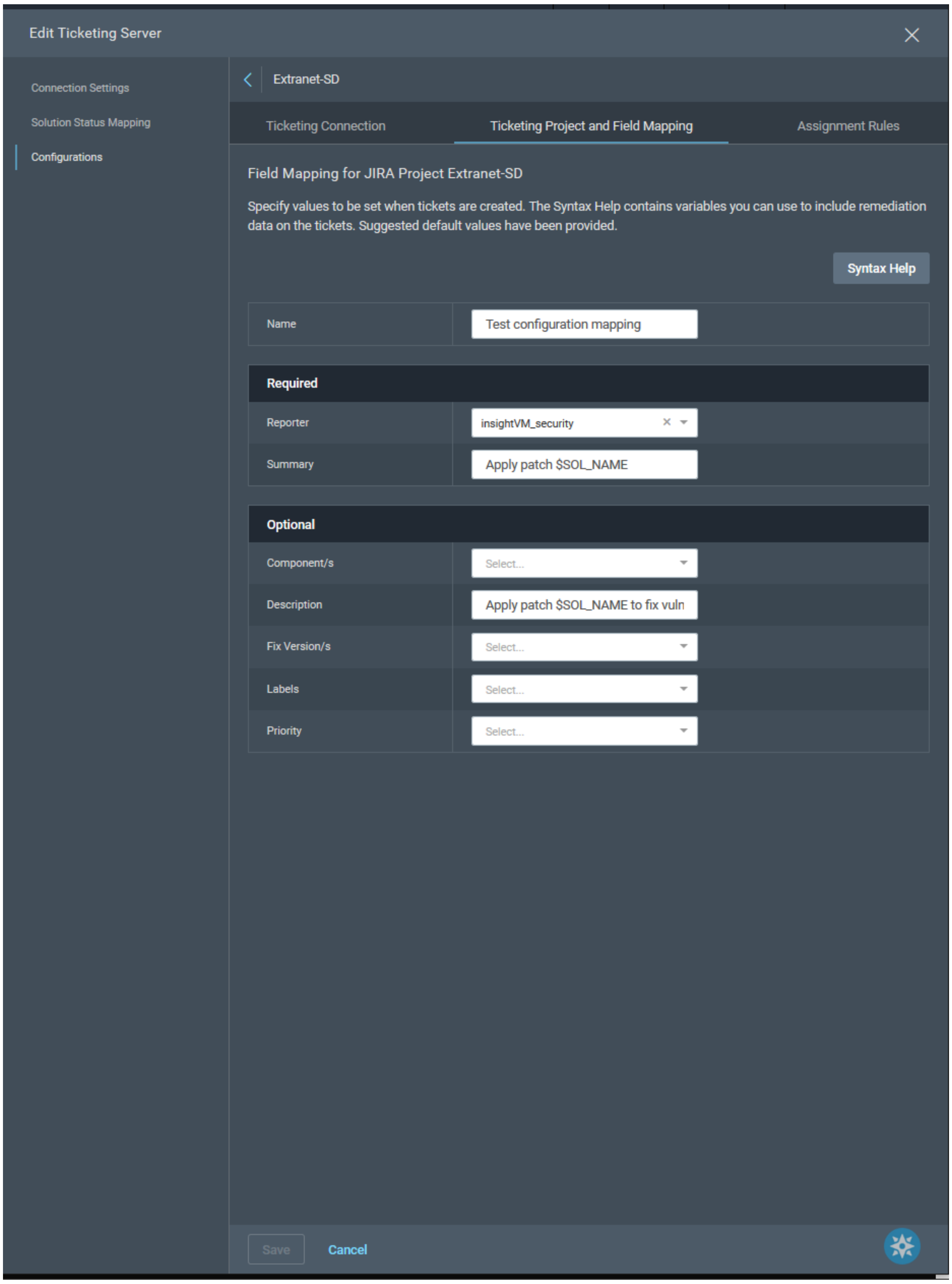
Next Cancel

Figure 12. Project connectivity

After that the admin configuring the system needs to specify what templates should be used when the automation has created the tickets. The fields that should be set would be the following:

- Security level
- Description
- Summary

That way the visibility of issues related to this project can be limited and there is enough information so that the specialists of the company can start fixing the issues as shown in Figure 13.



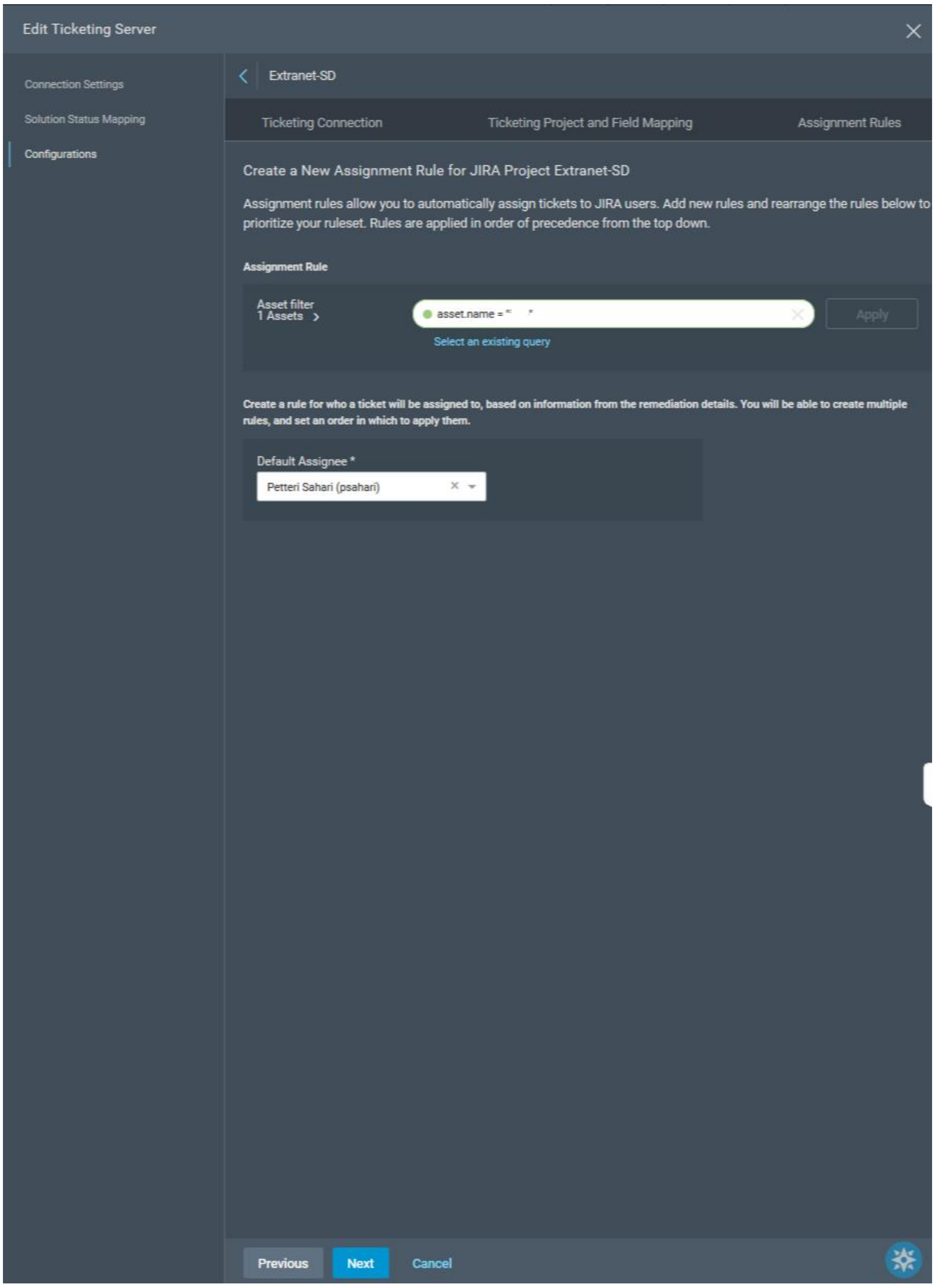
Field Mapping for JIRA Project Extranet-SD

Specify values to be set when tickets are created. The Syntax Help contains variables you can use to include remediation data on the tickets. Suggested default values have been provided.

Name	Test configuration mapping
Required	
Reporter	insightVM_security
Summary	Apply patch $SOL_NAME
Optional	
Component/s	Select...
Description	Apply patch $SOL_NAME to fix vuln
Fix Version/s	Select...
Labels	Select...
Priority	Select...

Figure 13. Ticketing Project and Field mapping

The user should next notice that there is an asset configured between the “marks”; however, it is hidden for security reasons as seen in Figure 14. For reference, the asset name could be the same as the hostname of that asset. The asset name could be for example “**test_asset_1**” or anything else. The default assignee for issues related to this asset has been assigned to the author of this thesis.



Edit Ticketing Server

Connection Settings
Solution Status Mapping
Configurations

Extranet-SD

Ticketing Connection | Ticketing Project and Field Mapping | Assignment Rules

Create a New Assignment Rule for JIRA Project Extranet-SD

Assignment rules allow you to automatically assign tickets to JIRA users. Add new rules and rearrange the rules below to prioritize your ruleset. Rules are applied in order of precedence from the top down.

Assignment Rule

Asset filter
1 Assets

asset.name = "   "

Select an existing query

Apply

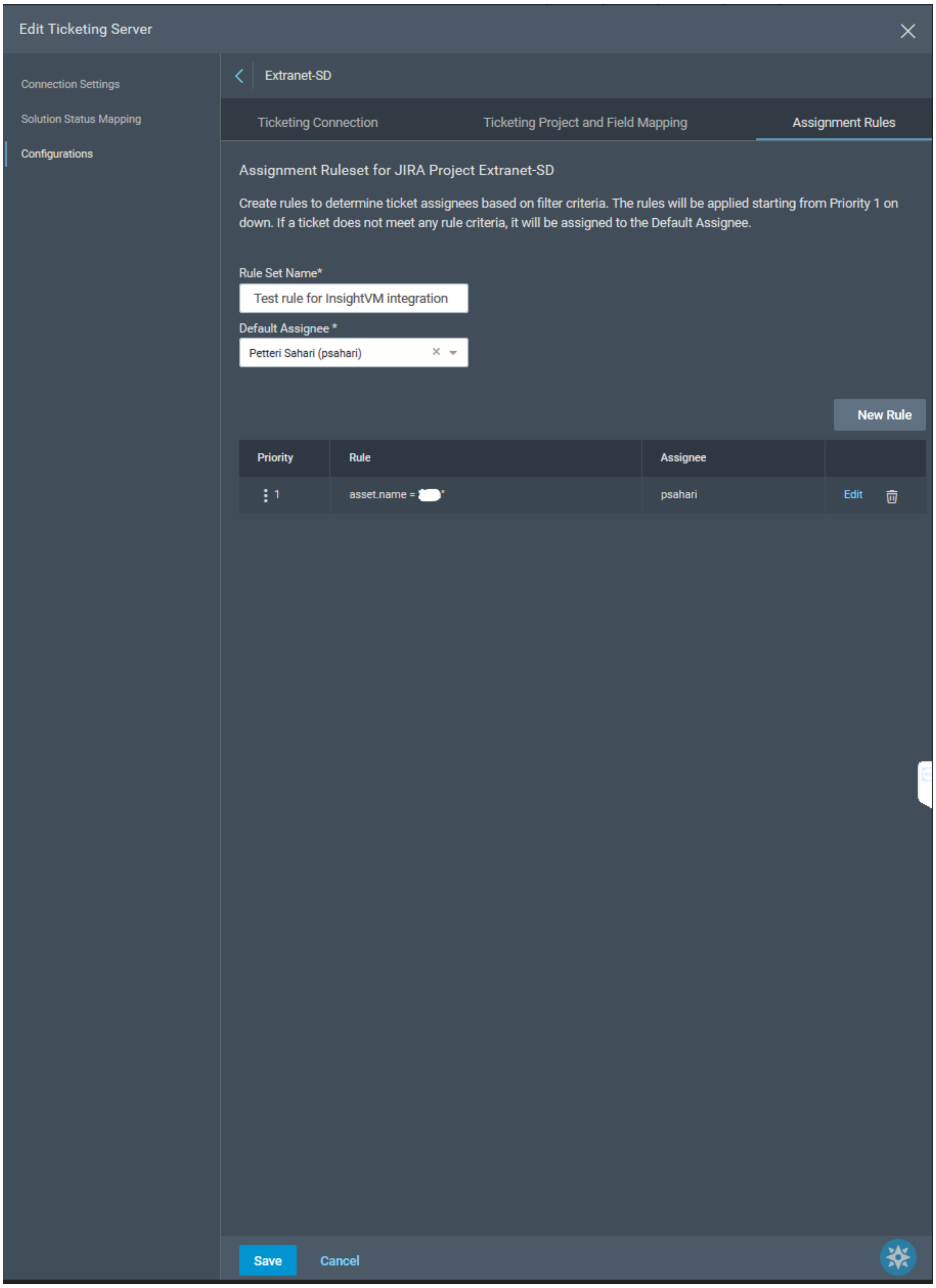
Create a rule for who a ticket will be assigned to, based on information from the remediation details. You will be able to create multiple rules, and set an order in which to apply them.

Default Assignee *
Petteri Sahari (psahari)

Previous Next Cancel

Figure 14. Configuring assignment rules

After the assignment rules have been configured, the newly created assignment appears below the “**Default assignee**” field as demonstrated in Figure 15.



Assignment Ruleset for JIRA Project Extranet-SD

Create rules to determine ticket assignees based on filter criteria. The rules will be applied starting from Priority 1 on down. If a ticket does not meet any rule criteria, it will be assigned to the Default Assignee.

Rule Set Name*: Test rule for InsightVM integration

Default Assignee *: Petteri Sahari (psahari)

Priority	Rule	Assignee	
1	asset.name = [redacted]	psahari	Edit

Figure 15. Assignment rules created successfully

After the connection is working and has been verified, InsightVM shows the ticketing server connection as enabled, which is shown in Figure 16.

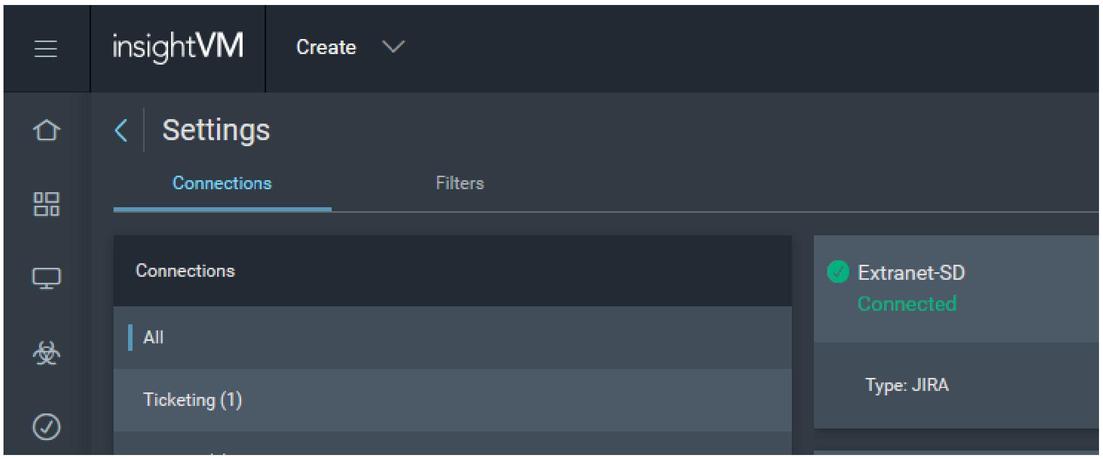
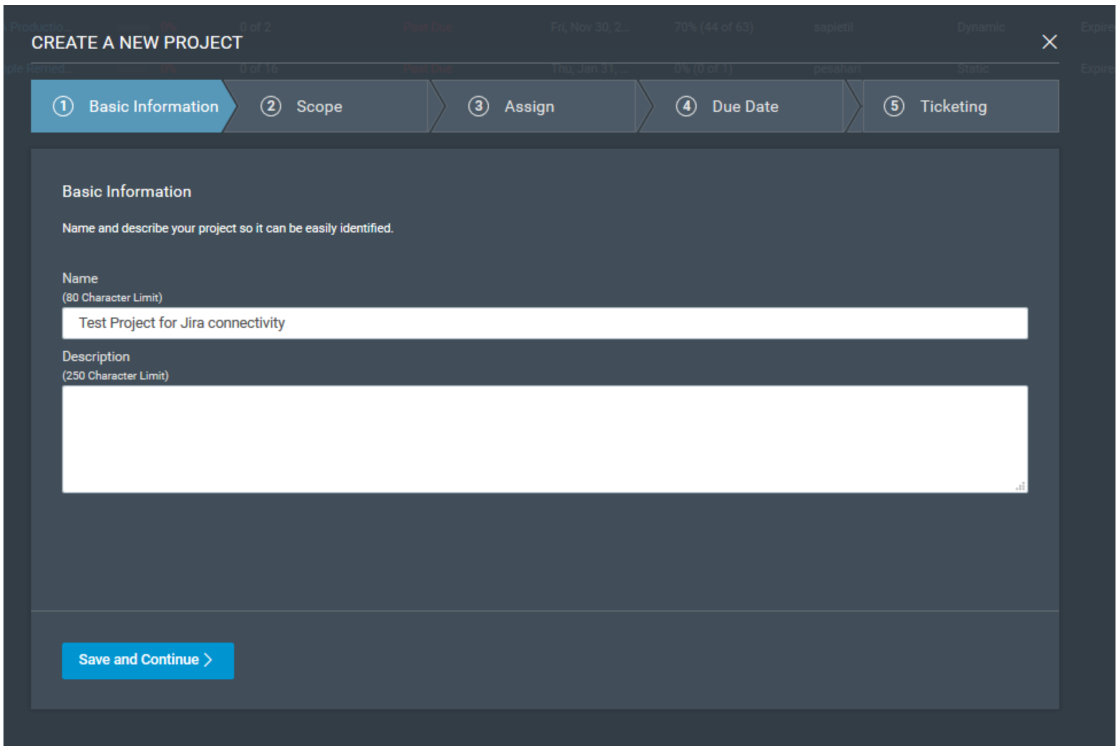


Figure 16. Working connectivity

6.2.2 Remediation project configuration

Connectivity has been confirmed to work; next, a remediation project needs to be created in order to make the ticketing integration to start working as illustrated in Figure 17.


CREATE A NEW PROJECT

① Basic Information ② Scope ③ Assign ④ Due Date ⑤ Ticketing

Basic Information

Name and describe your project so it can be easily identified.

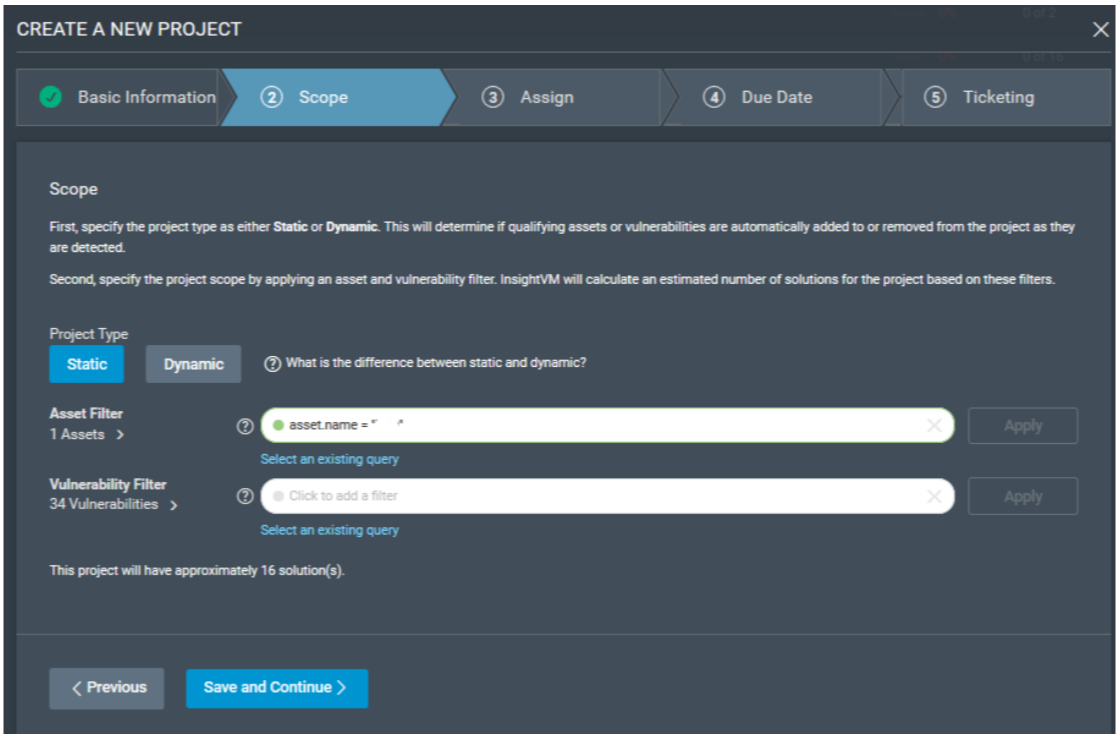
Name
(80 Character Limit)
Test Project for Jira connectivity

Description
(250 Character Limit)

Save and Continue >

Figure 17. Creating a new Remediation project

In the remediation project, the scope of assets was to be included from which the tickets are to be created as seen in Figure 18. One should notice that the asset name has also been hidden; yet again, the value could be anything (such as **test_asset_1**).



CREATE A NEW PROJECT

Basic Information | ② Scope | ③ Assign | ④ Due Date | ⑤ Ticketing

Scope

First, specify the project type as either **Static** or **Dynamic**. This will determine if qualifying assets or vulnerabilities are automatically added to or removed from the project as they are detected.

Second, specify the project scope by applying an asset and vulnerability filter. InsightVM will calculate an estimated number of solutions for the project based on these filters.

Project Type
Static | Dynamic | ⓘ What is the difference between static and dynamic?

Asset Filter
1 Assets >
● asset.name = "[redacted]" — Apply
Select an existing query

Vulnerability Filter
34 Vulnerabilities >
Click to add a filter — Apply
Select an existing query

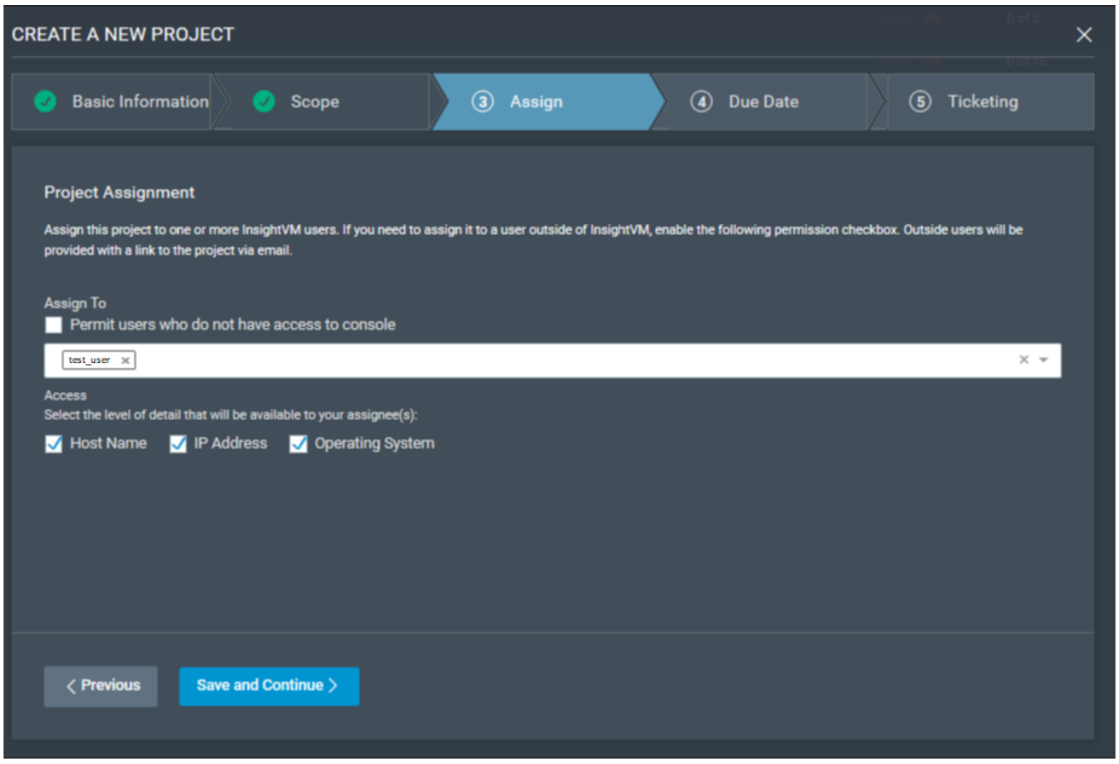
This project will have approximately 16 solution(s).

< Previous | Save and Continue >

Figure 18. Scope of assets

One should notice that 34 Vulnerabilities were found already in the test asset. It is important to keep that value in mind, and it is discussed later in this thesis in more detail.

Then, if one wants to allow users inside InsightVM to have access to the remediation project, the users need to be specified for that as demonstrated in Figure 19.



CREATE A NEW PROJECT

Project Assignment

Assign this project to one or more InsightVM users. If you need to assign it to a user outside of InsightVM, enable the following permission checkbox. Outside users will be provided with a link to the project via email.

Assign To

☐ Permit users who do not have access to console

test_user

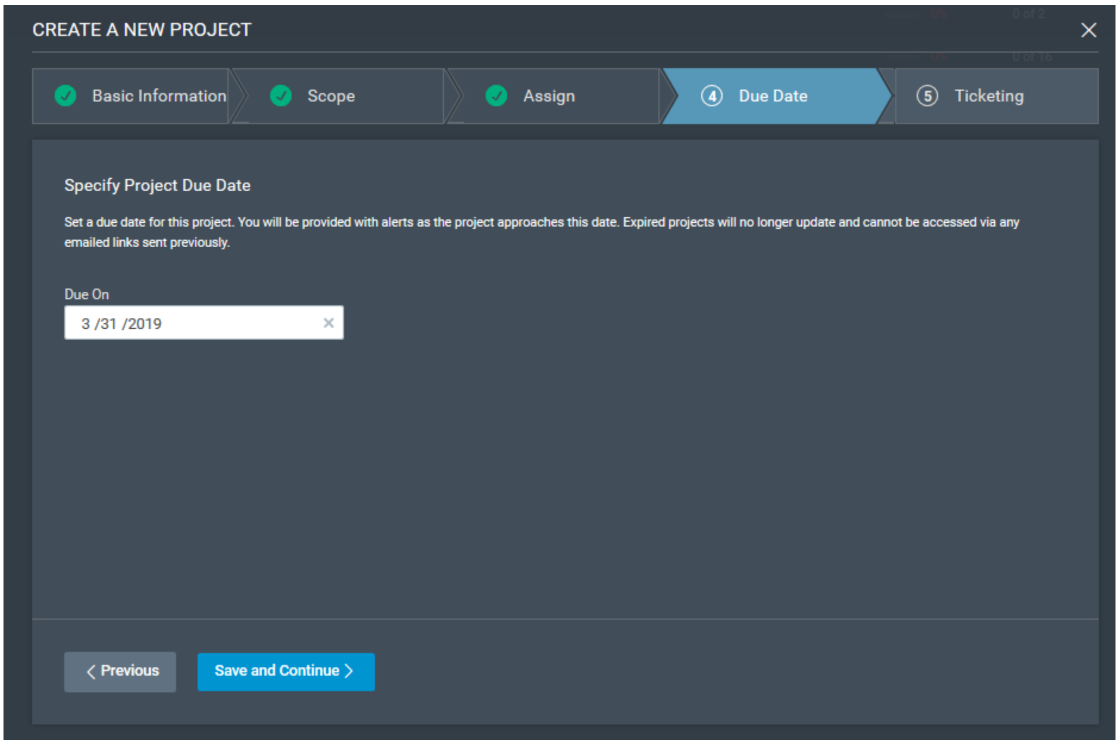
Access

Select the level of detail that will be available to your assignee(s):

☑ Host Name ☑ IP Address ☑ Operating System

Figure 19. Remediation project permissions

The next screen allows the user to change the due date to the upcoming issues reported by InsightVM as shown in Figure 20.



CREATE A NEW PROJECT

Specify Project Due Date

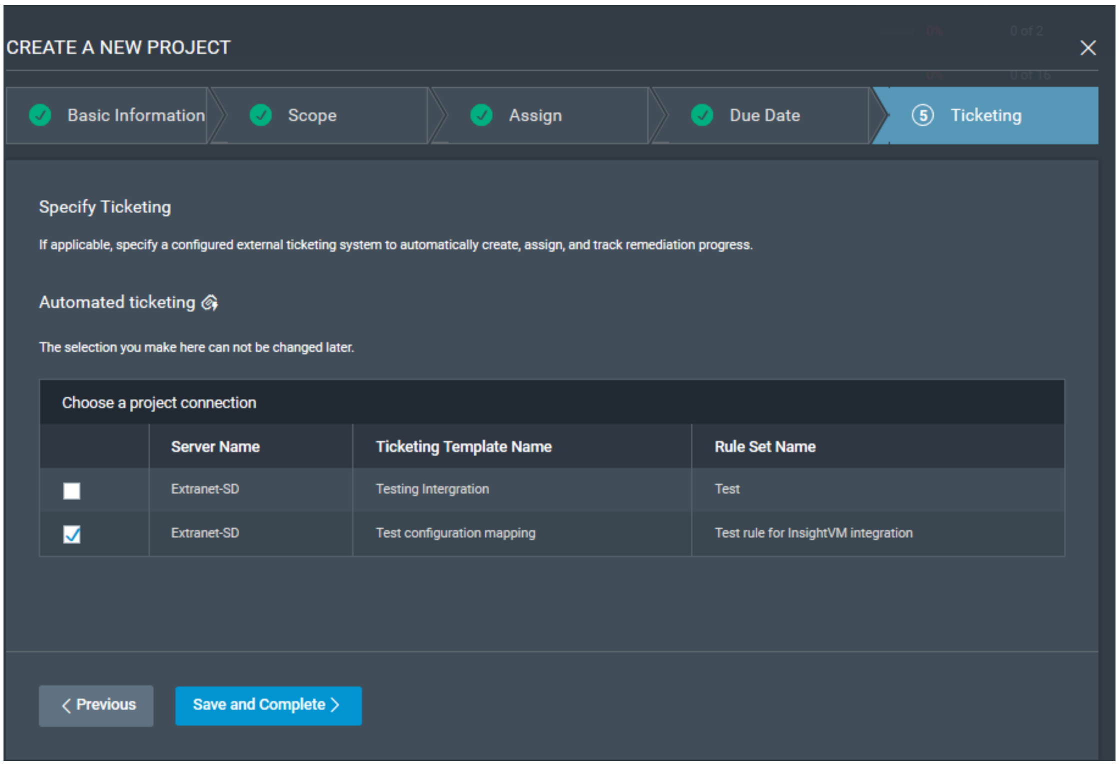
Set a due date for this project. You will be provided with alerts as the project approaches this date. Expired projects will no longer update and cannot be accessed via any emailed links sent previously.

Due On

3 /31 /2019

Figure 20. Due date configuration

The next step would be to specify, which ticketing system should be used for this remediation project. In this case, the connectivity created earlier is being used as seen in Figure 21.



Specify Ticketing

If applicable, specify a configured external ticketing system to automatically create, assign, and track remediation progress.

Automated ticketing

The selection you make here can not be changed later.

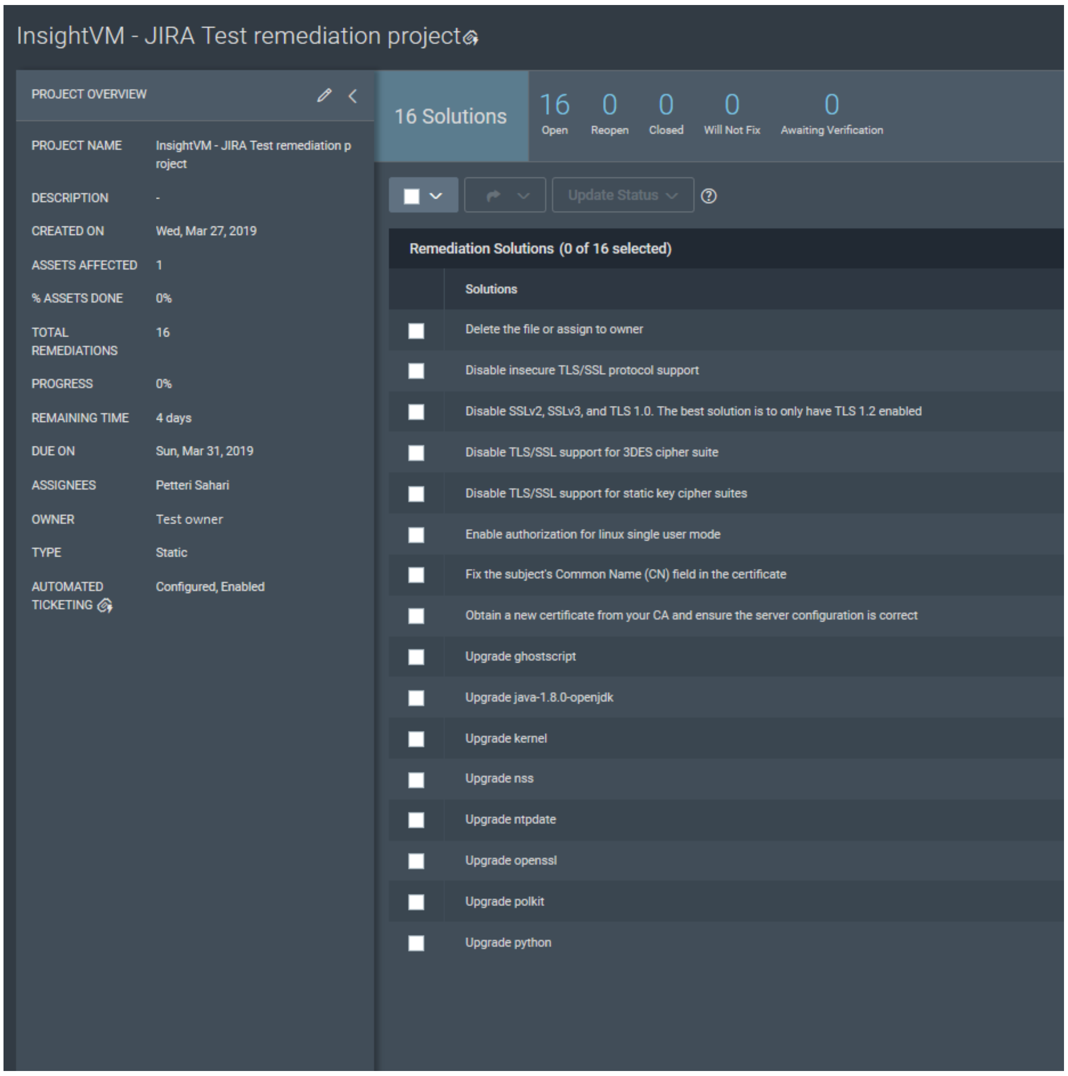
Choose a project connection

	Server Name	Ticketing Template Name	Rule Set Name
☐	Extranet-SD	Testing Intergration	Test
☑	Extranet-SD	Test configuration mapping	Test rule for InsightVM integration

Figure 21. Ticketing system options

7 Results

It was mentioned earlier that 34 vulnerabilities were found in the test asset; however, the remediation project only shows 16 solutions. The reason for the amount of solutions found being lower than the found vulnerabilities is that the asset group contains multiple hosts which have the same vulnerabilities; hence, the same remediation can be used for more than one of the hosts in the asset group, which is seen in Figure 22.



InsightVM - JIRA Test remediation project

PROJECT OVERVIEW

PROJECT NAME	InsightVM - JIRA Test remediation p roject
DESCRIPTION	-
CREATED ON	Wed, Mar 27, 2019
ASSETS AFFECTED	1
% ASSETS DONE	0%
TOTAL REMEDIATIONS	16
PROGRESS	0%
REMAINING TIME	4 days
DUE ON	Sun, Mar 31, 2019
ASSIGNEES	Petteri Sahari
OWNER	Test owner
TYPE	Static
AUTOMATED TICKETING	Configured, Enabled

16 Solutions

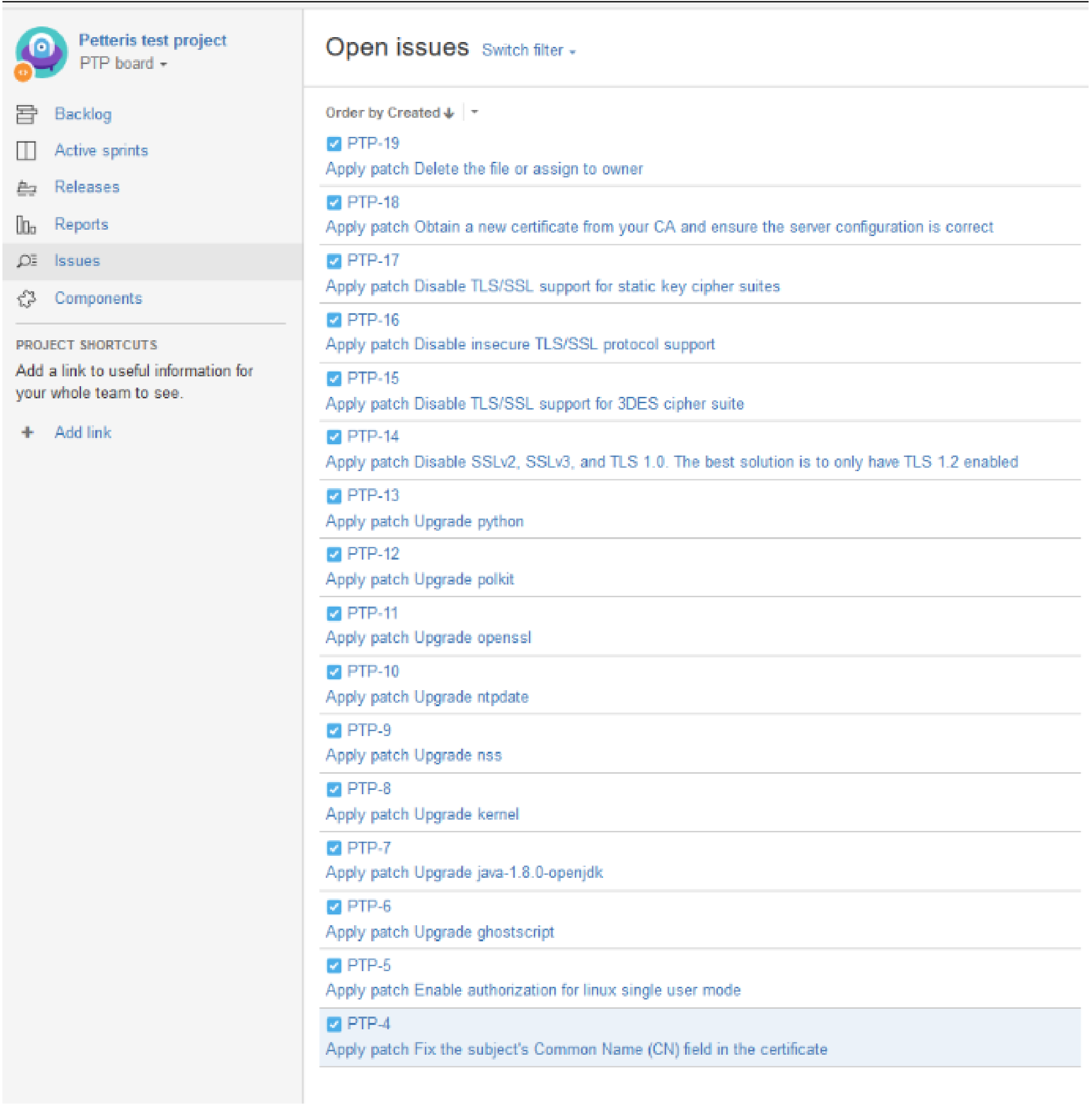
16	0	0	0	0
Open	Reopen	Closed	Will Not Fix	Awaiting Verification

Remediation Solutions (0 of 16 selected)

Solutions
Delete the file or assign to owner
Disable insecure TLS/SSL protocol support
Disable SSLv2, SSLv3, and TLS 1.0. The best solution is to only have TLS 1.2 enabled
Disable TLS/SSL support for 3DES cipher suite
Disable TLS/SSL support for static key cipher suites
Enable authorization for linux single user mode
Fix the subject's Common Name (CN) field in the certificate
Obtain a new certificate from your CA and ensure the server configuration is correct
Upgrade ghostscript
Upgrade java-1.8.0-openjdk
Upgrade kernel
Upgrade nss
Upgrade ntpdate
Upgrade openssl
Upgrade polkit
Upgrade python

Figure 22. Remediation project

From the writer’s perspective, it took a few minutes for the automation to create the issues; however, eventually they found their way into JIRA as shown in Figure 23.



Petteris test project
PTP board

Backlog
Active sprints
Releases
Reports
Issues
Components

PROJECT SHORTCUTS
Add a link to useful information for your whole team to see.

Add link

Open issues Switch filter

Order by Created

PTP-19
Apply patch Delete the file or assign to owner

PTP-18
Apply patch Obtain a new certificate from your CA and ensure the server configuration is correct

PTP-17
Apply patch Disable TLS/SSL support for static key cipher suites

PTP-16
Apply patch Disable insecure TLS/SSL protocol support

PTP-15
Apply patch Disable TLS/SSL support for 3DES cipher suite

PTP-14
Apply patch Disable SSLv2, SSLv3, and TLS 1.0. The best solution is to only have TLS 1.2 enabled

PTP-13
Apply patch Upgrade python

PTP-12
Apply patch Upgrade polkit

PTP-11
Apply patch Upgrade openssl

PTP-10
Apply patch Upgrade ntpdate

PTP-9
Apply patch Upgrade nss

PTP-8
Apply patch Upgrade kernel

PTP-7
Apply patch Upgrade java-1.8.0-openjdk

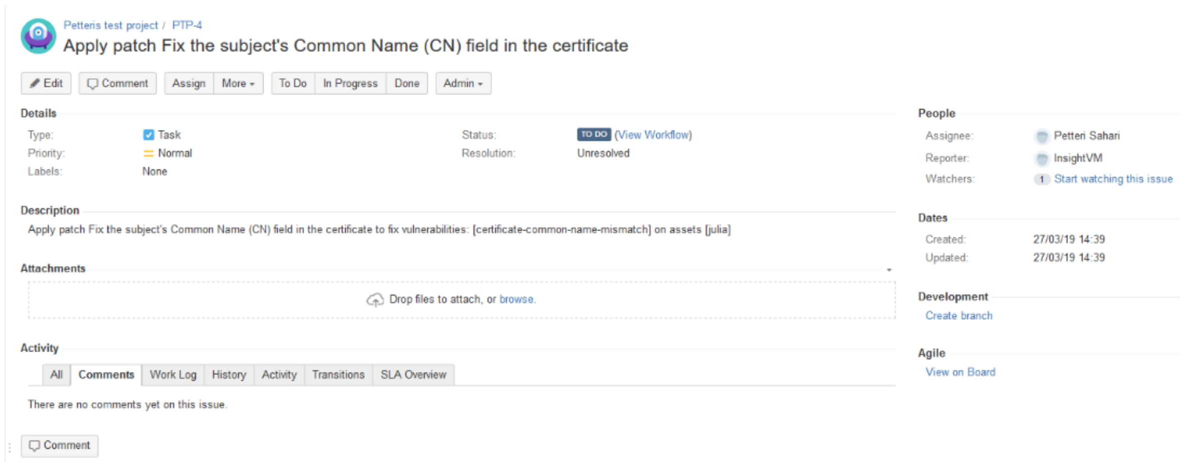
PTP-6
Apply patch Upgrade ghostscript

PTP-5
Apply patch Enable authorization for linux single user mode

PTP-4
Apply patch Fix the subject's Common Name (CN) field in the certificate

Figure 23. Issues created by the automation

To prove that it was indeed the automation that created the issues, it is possible to check that the issue was reported by the user “InsightVM” created earlier. The automation filled in the fields “Summary”, the title of the ticket and description with more detailed information about the found vulnerability and a solution to fix is shown in Figure 24.


Petteris test project / PTP-4

Apply patch Fix the subject's Common Name (CN) field in the certificate

Details

Type:	Task	Status:	TO DO (View Workflow)
Priority:	Normal	Resolution:	Unresolved
Labels:	None		

Description

Apply patch Fix the subject's Common Name (CN) field in the certificate to fix vulnerabilities: [certificate-common-name-mismatch] on assets [julia]

Attachments

Drop files to attach, or browse.

Activity

There are no comments yet on this issue.

People

Assignee:	Petteri Sahari
Reporter:	InsightVM
Watchers:	1 Start watching this issue

Dates

Created:	27/03/19 14:39
Updated:	27/03/19 14:39

Development

Create branch

Agile

View on Board

Figure 24. Example ticket from InsightVM

The ticket was even automatically assigned correctly as seen in the “Assignee” field.

8 Conclusions

The goal from the beginning was to make InsightVM and Jira to communicate together in order to reduce the workload caused by creating reports to Jira manually. Critical vulnerabilities found in the systems would need to be patched regardless of this automation as a requirement from company’s customers. Making the reporting automated will make the process of patching the systems easier once it was placed in the system.

Constructive research was chosen as the research method for this thesis, as the author of this document already knew the end results, i.e. what would happen when two systems were to be integrated together, so the work was actually just to understand and reasearch a way to accomplish the known outcome. The whole thesis required a great deal of experience and digging into Jira before understanding completely how the process would work in the end to integrate the systems together.

While writing this thesis, the author took a extensive look into Atlassian Jira and Rapid7 InsightVM product documentations to fully understand and explain to the reader how the systems work and to explain how much knowledge is needed in order to make these systems work together.

Through integrating two systems together, the process automatisation produces clear value to the company by moving resources that were previously used for routine work to now being used for work that is harder and requires more brainwork.

9 Discussion

Had the company chosen another set of products for accomplishing these tasks, it would have probably been a slightly more difficult task to integrate these systems as the author had experience of Jira before starting this thesis.

With previous experience in the systems it was much easier to get hold of what steps are actually needed in order to make the wanted outcome to work. After accomplishing the set goals, the writer is confident that similar integrations will be easier to handle in the future regarding other systems as well.

References

(2018) Webinar: The new Jira begins now. Wong, J. and Breton, J. Video Published to YouTube 26 November 2018. Accessed on 24 December 2018. Retrieved from https://www.atlassian.com/software/jira/demo

Advanced workflow configuration. n.d. Atlassian confluence site. Published 16 August 2018. Accessed 25 December 2018. Retrieved from https://confluence.atlassian.com/adminjiracloud/advanced-workflow-configuration-776636620.html

Atlassian JIRA Configuration Tutorial: JIRA Schemes. Crisostomo, M. The Grey Blog website. Published 11 November 2010*.* Accessed 25 December 2018. Retrieved from http://thegreyblog.blogspot.com/2010/11/atlassian-jira-configuration-tutorial.html

Building an awesome Jira workflow: concepts and examples. Radigan, D. Atlassian website. Published 16 October 2013. Accessed 25 December 2018. Retrieved from https://www.atlassian.com/blog/jira-software/building-workflow-awesome

Configuring workflow triggers. n.d. Atlassian confluence site. Published 6 August 2018. Accessed 25 December 2018. Retrieved from https://confluence.atlassian.com/adminjiraserver071/configuring-workflow-triggers-802592811.html

Defining a project, n.d. Atlassian confluence site. Published 27 July 2018.-Accessed on 23 December 2018. Retrieved from https://confluence.atlassian.com/admin-jiraserver/defining-a-project-938847066.html

Defining a screen. n.d. Atlassian confluence site. Published 29 June 2018. Accessed on 24 December 2018. Retrieved from https://confluence.atlassian.com/adminjiraserver/defining-a-screen-938847288.html

How does JIRA indexing work? n.d. Atlassian confluence site. Published 19 November 2018. Accessed on 24 December 2018. Retrieved from https://confluence.atlassian.com/jirakb/jira-indexing-faq-776654790.html

How do you explain Jira? Allen, K. Isostech. Published 10 February 2015. Accessed 26 January 2019. Retrieved from https://www.isostech.com/blogs/atlassian/explain-jira/

JIRA in a Nutshell demo video. Video Published to YouTube 12 November 2012. Accessed 22 April 2019. Retrieved from https://www.youtube.com/watch?v=xrCJv0fTyR8

Managing project permissions. n.d. Atlassian confluence site. Published 6 December 2018. Accessed 23 December 2018. Retrieved from https://confluence.atlassian.com/adminjiracloud/managing-project-permissions-776636362.html#Managingprojectpermissions-permission_schemes

Managing project roles. n.d. Atlassian confluence site. Published 6 December 2018. Referenced 26 December 2019.

https://confluence.atlassian.com/adminjiracloud/managing-project-roles-776636382.html

Pasian, Beverly. Published May 2015. Designs, Methods and Practices for Research of Project Management. Gower Publishing.

PlayStation Network and Qriocity Outage FAQ. Caplin, N. Playstation Blog. Published 28 April 2011. Accessed 26 January 2019. Retrieved from https://blog.eu.playstation.com/2011/04/28/playstation-network-and-qriocity-outage-faq/

Project screens, schemes and fields. Atlassian confluence site. Published 19 April 2018. Accessed 22 April 2019 Retrieved from https://confluence.atlassian.com/adminjiraserver071/project-screens-schemes-and-fields-802592517.html

Streamline Vulnerability Remediation Workflows. N.d. Rapid7 site. Published 12 April 2018. Accessed 25 December 2018. Retrieved from https://www.rapid7.com/globalassets/_pdfs/product-and-service-briefs/rapid7-insightvm-jira-integration-brief.pdf

What is a Black-Hat hacker? N.d. Kaspersky site. Published N/A. Accessed 7 February 2019. Retrieved from https://www.kaspersky.com/resource-center/threats/black-hat-hacker

What is Jira Service Desk? n.d. Atlassian confluence site. Published 12 September 2018. Accessed 26 January 2019. Retrieved from https://confluence.atlassian.com/get-started-with-jira-service-desk/what-is-jira-service-desk-917968347.html

Working with asset groups Rapid7 site, Published N/A. Accessed 7 April 2019 Retrieved from https://insightvm.help.rapid7.com/docs/working-with-asset-groups