Multi-Tenancy Support

FortiSOAR 7.6.2

FORTINET DOCUMENT LIBRARY
https://docs.fortinet.com

FORTINET VIDEO LIBRARY
https://video.fortinet.com

FORTINET BLOG
https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT
https://support.fortinet.com

FORTINET TRAINING & CERTIFICATION PROGRAM
https://www.fortinet.com/training-certification

FORTINET TRAINING INSTITUTE
https://training.fortinet.com

FORTIGUARD LABS
https://www.fortiguard.com

END USER LICENSE AGREEMENT
https://www.fortinet.com/doc/legal/EULA.pdf

FEEDBACK
Email: techdoc@fortinet.com

April, 2025
FortiSOAR 7.6.2 Multi-Tenancy Support
00-400-000000-20210107

TABLE OF CONTENTS

Change Log

Date	Change Description
2025-10-14	Updated the Configuring High Availability topic for releases 7.6.2 onwards in the Distributed Tenancy Support chapter.
2025-04-29	Initial release of 7.6.2

Overview of Multi-tenancy support in FortiSOAR

FortiSOAR provides additional features and out-of-the box configuration for a multi-tenant environment, enhancing its native support for multi-tenancy for managed security services providers (MSSPs). The FortiSOAR platform supports multi-tenancy in either of the following forms as well as a hybrid of both forms:

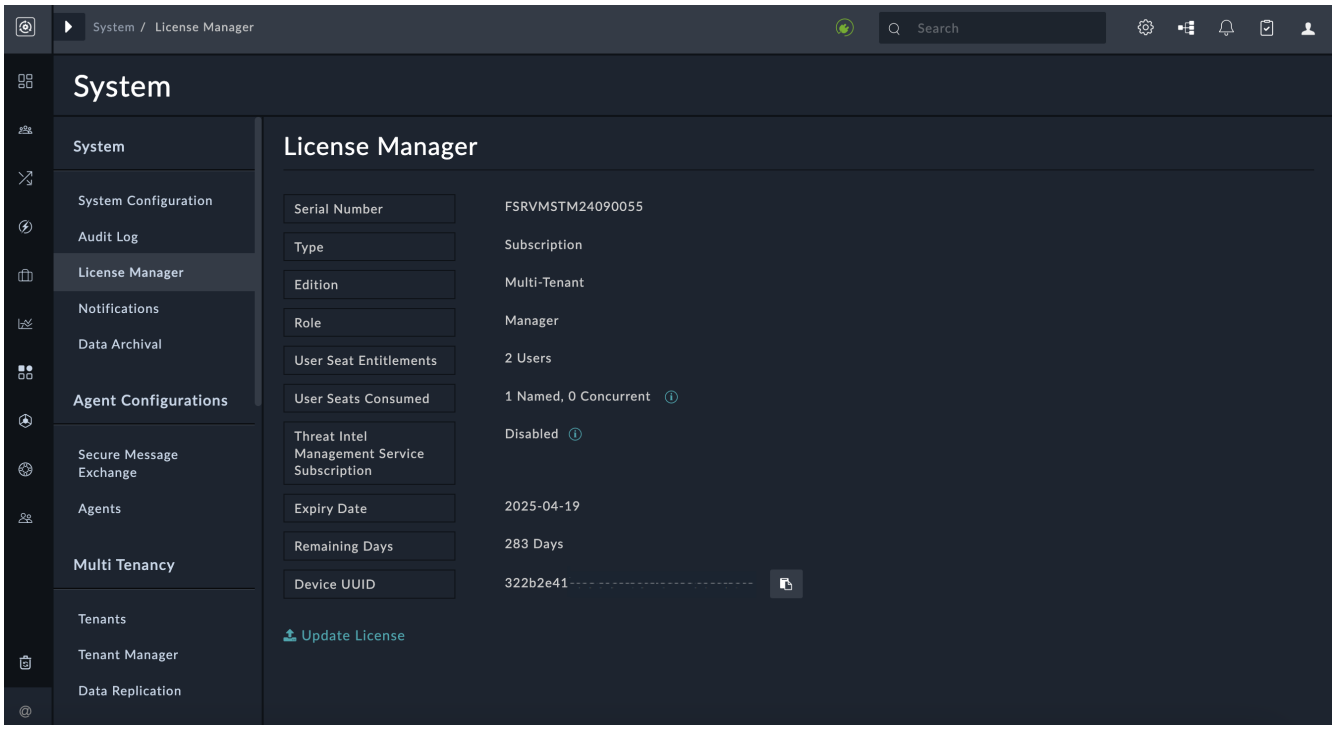
- Distributed Tenancy support: Used by MSSPs to provide services to remote tenants. In this case, each tenant will have its own FortiSOAR instance that will also automatically communicate to a master FortiSOAR instance having visibility across all the tenants. For more information, see the Distributed Tenancy Support chapter.
- Shared Tenancy support: Used by master (teams like the SOC team) to provide services to local tenants. In this case, you have only a single FortiSOAR instance, wherein multiple tenants will access a single FortiSOAR instance. For more information, see the Shared Tenancy Support chapter.

When multi-tenancy is enabled for the FortiSOAR platform, all alerts, incidents, and other forms of records created are associated with a tenant. RBAC and ownership rules applied to each tenant govern who has visibility to which records, thereby restricting visibility of one tenant's records to the other, but at the same time, providing the service provider a single consolidated view across tenants.

Tenants act like a wrapper that can contain multiple agents, which can connect to various disparate networks and remotely execute action. In the case of dedicated tenants, a default agent is automatically created and added to the dedicated tenant as part of the tenant creation process. In the case of shared tenancy, you do not require to create an agent for that tenant, since everything in case of a shared tenant runs on the master instance only using the master agent. If you require to run some actions on a separate network, then you can add and configure another agent, associate that with the shared tenant. The RBAC of this agent will be controlled by correct team assignment. Further, you can also add multiple agents to both shared and dedicated tenants. For more information on agents and how to run remote actions using agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide."

Licensing for Multi-Tenancy

For multi-tenancy support, you require a FortiSOAR license that has been enabled for multi-tenancy. Therefore, when you are generating a license in FortiCare, ensure that you generate a license that has multi-tenancy support, i.e., a license with "Edition" set to 'Multi-Tenant' and "Role" set as 'Manager' for a master node (as shown in the following image), 'Tenant (Single user locked)' for a dedicated tenant node and 'Tenant (Multiple user capable)' for organizations having a distributed SOC:



For more information about licensing, see the *Licensing FortiSOAR* chapter in the "Deployment Guide."

You can deploy the FortiSOAR license using the FortiSOAR Admin CLI (`csadm`) as follows:

SSH to your FortiSOAR VM and login as a *root* user and run the `# csadm license --deploy-license <License File Path>` command. For more information on `csadm`, see the *FortiSOAR Admin CLI* chapter in the "Administration Guide."

You will see a **Multi Tenancy** section in your `System` page by clicking the **Settings** icon on the top-right corner of FortiSOAR, only if your FortiSOAR license has been enabled for multi-tenancy.

After your license is deployed, FortiSOAR runs a **Publish** on the instance. If during this time, you try to open the FortiSOAR UI, you might see a "Publish In Progress" message. Also, you cannot revert the environment to a non-multitenant environment once you have deployed the license.

Notes for upgraded multi-tenant configurations

- For the procedure of upgrading multi-tenant configurations, see the "Upgrade Guide."
- In case of a distributed deployment, both the master and the tenant nodes must be upgraded.
- From version 6.4.1 onwards, each dedicated tenant automatically creates an agent that can be used to remotely execute actions. You can add multiple agents to a tenant, therefore, tenants become a wrapper that can contain various agents that can connect to various disparate networks. For more information on agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide."

Recommended Resource Requirements for Virtual Machines (VM)

For Master and Tenant

Recommended Specifications

- 12 available vCPUs
- 48 GB available RAM
- 1 TB available disk space: Recommended to have high-performance storage, preferably SSDs.
- 1 vNIC

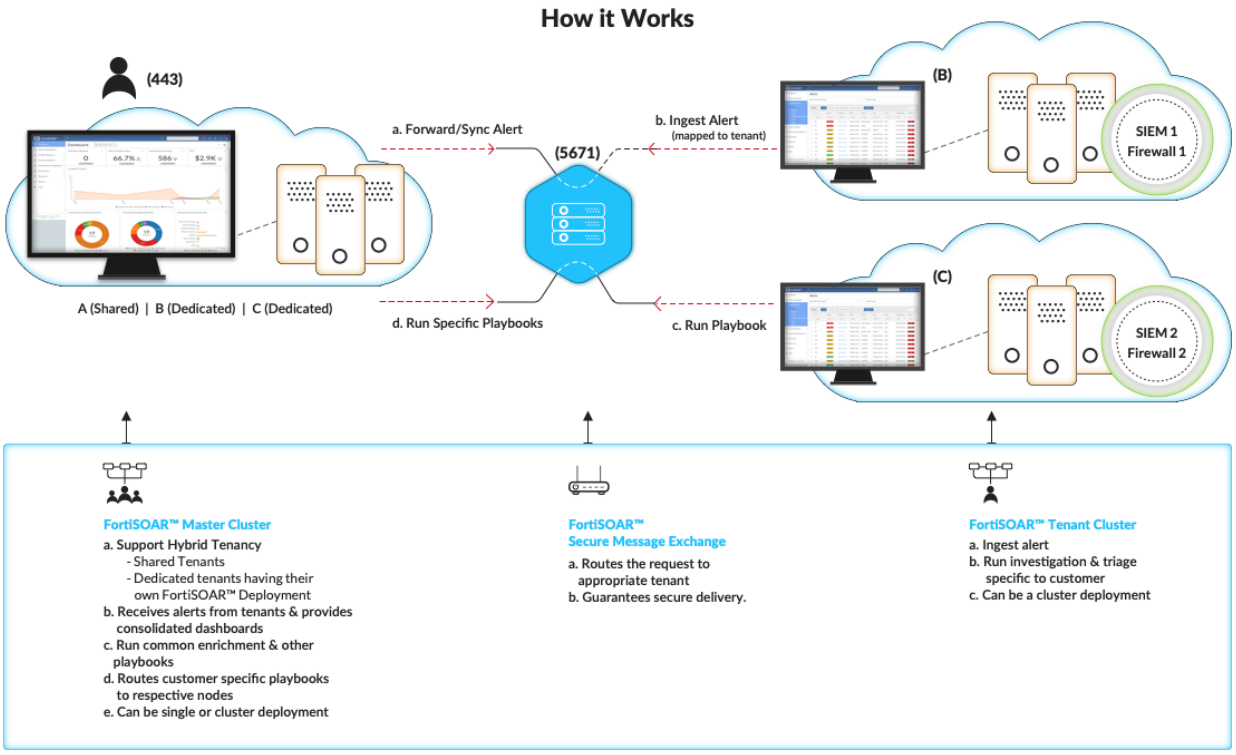
For Secure Message Exchange

Recommended Specifications

- 8 available vCPUs
- 16 GB available RAM
- 100 GB available disk space: Recommended to have high-performance storage, preferably SSDs.
- 1 vNIC

Architecture

The overall architecture of the FortiSOAR Distributed Multi-Tenancy Model:



A brief description of how FortiSOAR supports the Distributed Multi-Tenancy model follows:

- A master FortiSOAR node is installed and configured at the MSSP (Service Provider) location.
- A FortiSOAR tenant node is installed and configured for each tenant (customer) location, same as configured for the master node.
  Note: Each of these FortiSOAR installations (both master as well as tenants) are fully functional standalone instances. They are additionally enabled for automatic replication of data between them, and also for remotely executing workflows at the tenant node from the master node itself.
  - Communication between the master and tenant nodes are done over a secure channel using the Secure Message Exchange.
    Each tenant node communicates with the secure message exchange on a dedicated space that is access controlled with credentials that are unique to each tenant.
    The secure message exchange ensures a guaranteed message delivery irrespective of whether the target node is up or not at the time the message is sent. The messages are queued and delivered to the target node once it comes online. This prevents message loss during network disruptions.

An example: The master FortiSOAR node is monitoring the tenants' FortiSOAR instance, and if for example, the investigation requires to run a playbook to get the reputation of a particular IP address from the tenants' end, then this facility allows the master node to run the playbook remotely at the tenant's end. The playbook will be run at the tenant node, using the tenant's credentials, for the 3rd-party connector for checking the IP reputation, and the remediation actions (if any), specified in the playbook, will also be done on the tenants' end.

Shared Tenancy Support

FortiSOAR supports a shared tenancy model for multi-tenancy and MSSP support. A managed security service provider (MSSP) is an IT service provider that provides organizations with some amount of cybersecurity monitoring and management, which may include virus and spam blocking, intrusion detection, and firewalls and virtual private network (VPN) management.

In the case of shared tenancy, tenants share the same system as that of the master, i.e., tenants are local, but with restricted access on the system. The master (teams like the SOC team) provides cybersecurity monitoring and management to various tenants (customers/teams) in a single FortiSOAR instance.

You can choose between the Distributed and Shared tenancy models depending on your requirements based on the following points:

- In case of distributed tenancy support, tenants have greater control over the data that they share with the master.
- In case of distributed tenancy, you can avoid additional infrastructure overheads such as, directly connecting the tenant's SIEM to the MSSPs SOAR platform etc.
- Scaling considerations: If you have a large number of tenants then it is easier to scale in case of the distributed tenancy support model, as data primarily resides at the tenant nodes, and the computations also happen at the tenant, a single console at the master node can easily handle multiple customers.

The shared tenancy model ensures that data of different tenants are segregated, and data access is controlled using RBAC (role-based access control). Therefore, a tenant can view only their own data or record and not the data for other tenants.

Each tenant can be given their own login, using which they can view their dashboards, report, check the actions taken on their records, check their SLA management, etc.

The master can extend the `Tenants` module to include fields that they require for various tenant, such as address of the tenant, tenant SLA, etc. For more information, see the *Extending the Tenants module* section in the Distributed Tenancy Support chapter.

Steps required for deploying and configuring shared tenancy support

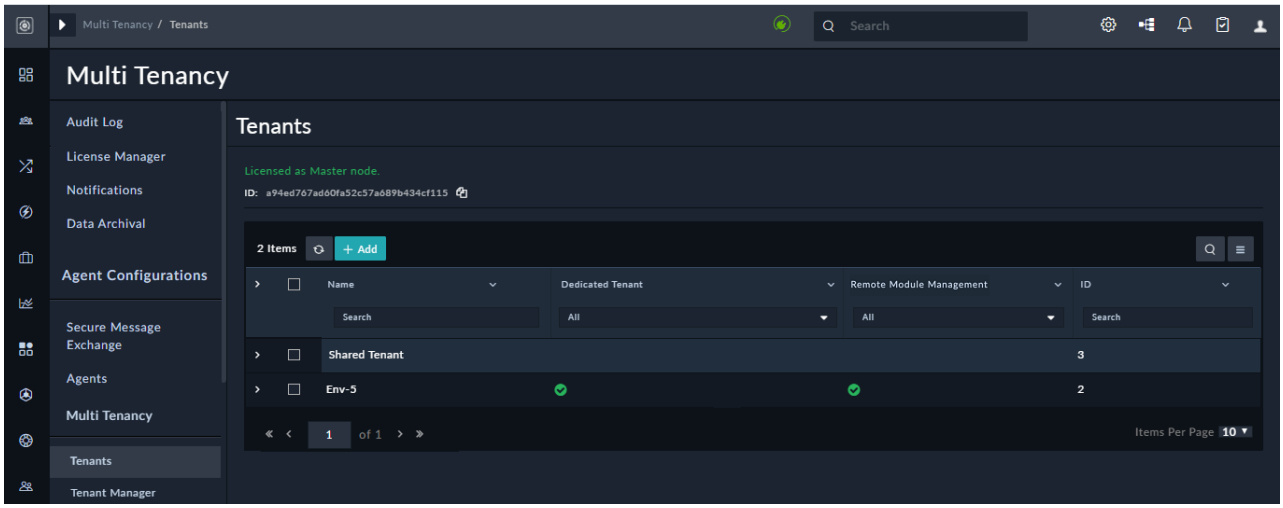
1. **Licensing FortiSOAR**
   For shared tenancy support, you require a FortiSOAR license that has been enabled for multi-tenancy. Therefore, when you are generating a license in FortiCare, ensure that you generate a license for multi-tenancy support, i.e., a license with edition set to "MT". For more information, see *Licensing for Multi-Tenancy* section in the Overview chapter.
2. **Deploying FortiSOAR**
   See the *Deploying FortiSOAR* chapter in the "Deployment Guide" for detailed information on how to deploy the FortiSOAR Virtual Appliance. For more information, see the *Deploying the Master Node* section in the Distributed Tenancy Support chapter.
3. **Configuring Shared Tenancy**
   See the Configuring Shared Tenancy section for configuring and onboarding local tenants.

Configuring Shared Tenancy

Only if your FortiSOAR license has been enabled for multi-tenancy you will see a **Multi Tenancy** section on your `System` page.

To see whether your system has been enabled for shared tenancy, you can do the following:

1. Log on to FortiSOAR as an administrator.
2. Click the **Settings** ( ) icon to open the `System` page.
3. On the `System` page, you will see the `Multi Tenancy` section. Click the **Tenants** item in the left menu, to view details of the master.
   On the top of the `Tenants` page, you will see the tenant ID of the master node, and also see `Licensed as Master node` text.



Onboarding tenants

Adding tenants

1. Log on to FortiSOAR as an administrator and click the **Settings** icon to open the `System` page.
2. To add tenants, click **Tenants** in the left menu and on the `Tenants` page, click **Add**.
   To edit the configuration of an existing tenant, click the tenant whose configuration you want to update, which opens the agent record in the detail view. Update the configuration parameters as required, and click **Save**.
   If you no longer require an existing tenant you can deboard that tenant. Deboarding a tenant is an irreversible operation which also deletes all data related to that tenant from the master node. For more information see Deboarding Tenants.
   If you want to deactivate a shared tenant, then you can deactivate the user login for that shared tenant.
3. In the `Add Tenant` dialog, configure the following parameters:
   a. From the **Choose Tenant Type** field choose the type of tenant you want to add.
      In this case, select **Shared** to add this tenant as a local tenant of the master, i.e., in the same FortiSOAR

instance for shared tenancy. Local or Shared tenants, based on their permissions, share the current FortiSOAR instance with the Master.

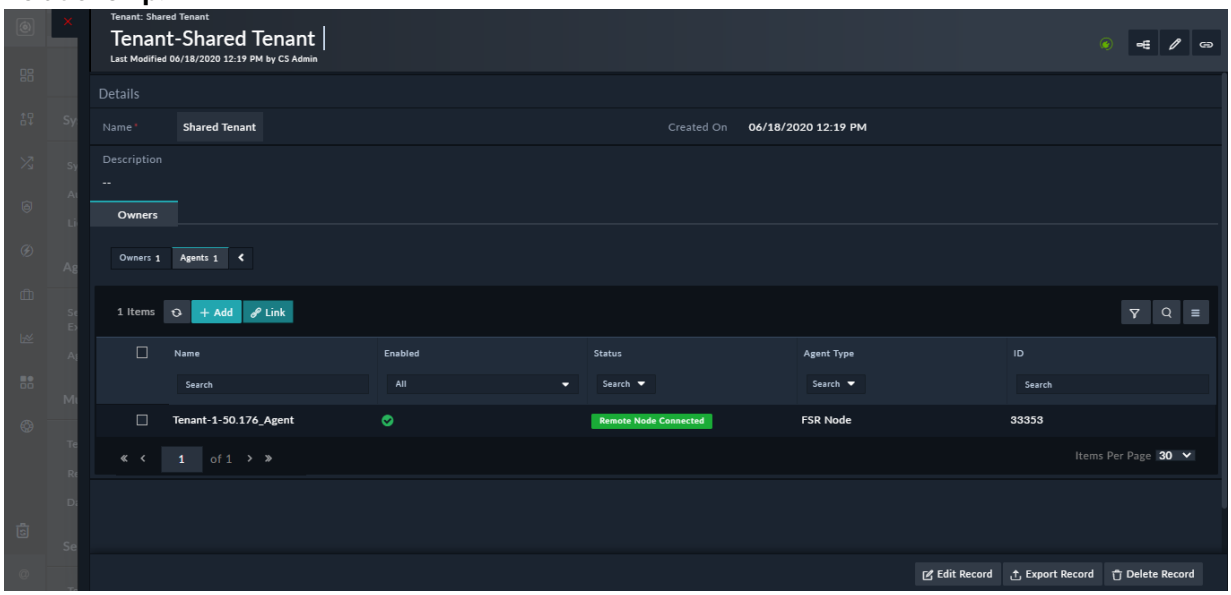
b. In the **Name** field, enter the name of the tenant.

c. (Optional) In the **Description** field, enter the description of the tenant.

d. From the **Owners** drop-down list, select the teams that you want to add as an owner to the records coming from this tenant and click **Link Team**.
The teams that you have selected and the SOC Admin team (considering that the tenant has been created using the SOC Admin team) will be associated as owners of any record created for that tenant. If no team is selected, then the teams that the user who adds the tenant belongs to will be associated as owners of any record created for that tenant.

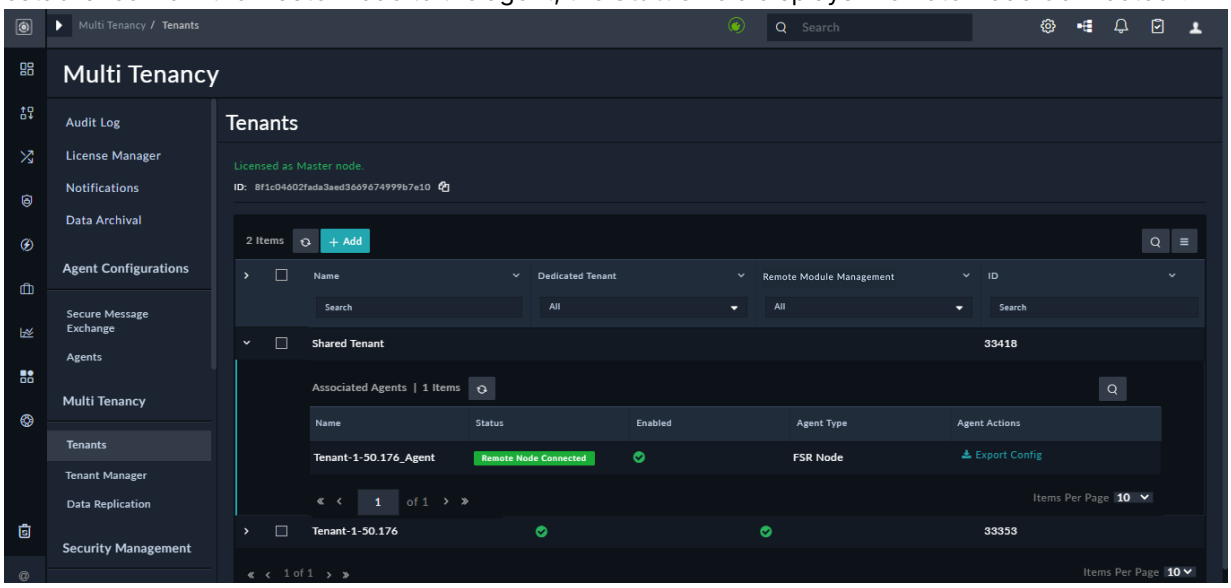
e. To complete adding a new tenant associated with the master node, click **Save**.
As tenants are wrappers, no configuration state is associated with a tenant and you also do not need to create an agent, since everything in case of a shared tenant runs only on the master instance using the master agent. However, if you require to run some actions on a separate network, then you can add and configure another agent, and then associate that with the shared tenant.

f. To link an agent with a shared tenant, open the record of the shared tenant and in the `Owners` section, click the **Agents** tab, and click **Link** to open the `Link Agents` dialog that contains a list of configured agents. Select the agent that you want to associate with the shared tenant and click **Save Relationship**:



Once you have linked an agent you can expand the row of the shared tenant and check the connection between the agent and the master node using secure message exchange. Once the connection is

established from the master node to the agent, the **Status** field displays "Remote Node Connected".



To assist with debugging issues on the Agent, you can obtain agent logs from the master node starting with release 7.4.2, as explained in the 'Downloading Agent Logs' topic in the Distributed Tenancy Support chapter.

List of Statuses

Following is the list of statuses that can be displayed:

- **Configuration In Progress**: Process of configuring the agent has begun.
- **Awaiting Remote Node Connection**: Connection between the FortiSOAR node and secure message exchange is established and awaiting the connection to the agent.
- **Remote Note Connected**: Agent has been connected to the FortiSOAR node using secure message exchange.
- **Configuration Failed**: Agent failed to be added on the secure message exchange.
- **Message Exchange Unreachable**: Secure message exchange is unreachable.
- **Remote Node Unreachable**: Agent is unreachable from the FortiSOAR node.

Associating the tenant with teams and users

If you have not associated the tenant with teams at the time of adding tenants (step 3e of the previous procedure), then once you have created a tenant, you must associate the tenant to team(s). Tenants must be associated with teams so that all records created for a given tenant are automatically owned by the same teams who are associated with the tenant. For example, if alerts are created whose tenant is set as T01, then only the teams associated with team T01 and their ancestors in the team hierarchy will be able to view the alert record.

Steps mentioned in the following sections describe how to add a team and then associate the tenant with that team.

Once you associate the tenant with teams you must also assign users to those teams, as mentioned in the following *Adding a user to a team* section.

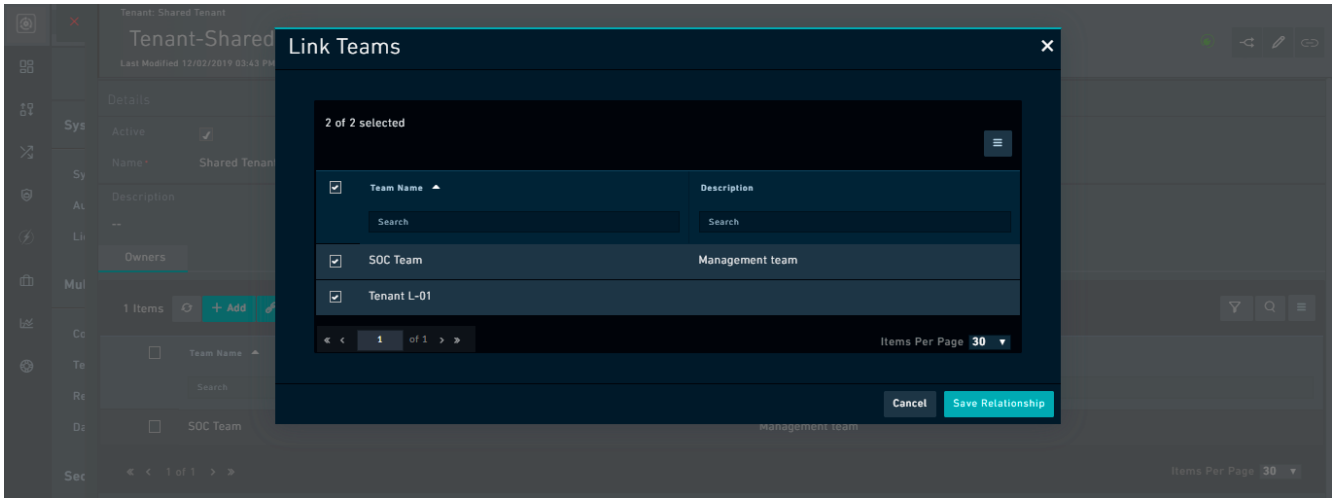
Adding a Team

To add a team, click the **Settings** icon and in the `Security Management` section click **Teams**. Click **Add Team** to display the `Add Team` dialog and enter the name of the team that you want to create, for example, `Tenant-L01` and optionally add a description for the team and click **Create** to add the new team.

Associating the tenant with the teams

You should now associate the tenant with this newly created team, so that records created in this tenant will be owned by the team. You must also link a management or master team, such as the `SOC Team` with the tenant, or ensure that the master team is at the top of the team hierarchy, so that record will be visible to the master.

To assign the tenant to a team, click the **Settings** icon and in the `Multi Tenancy` section click **Tenants**. Click the tenant that you want to assign to the team, for example, `Tenant-L01`, which opens the tenant record. In the tenant record, click **Link**, and in the `Link Teams` dialog, select the team that you want to associate with the tenant. In our example, select the `Tenant-L01` and `SOC Team` team and click **Save Relationship**:

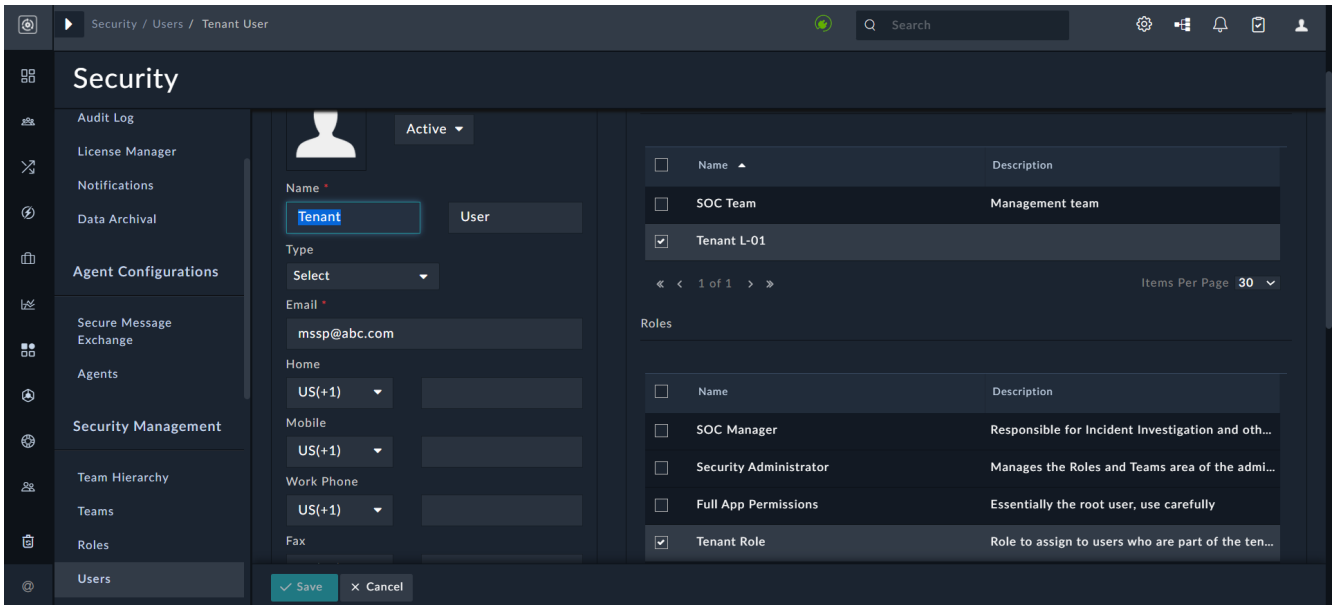


Adding a user to a team

The users in the team that you have created should be assigned to a role with a minimum of `Read` permission on the `Application`, `Tenants`, and `People` modules, apart from required permissions on the case management modules, like `Alerts`, `Incidents`, `Tasks`, etc. You must not assign these users any `Security` permissions. It is also recommended that you should not assign any `Playbook` or `Connector` permissions to these users, if you do not want them to run any playbooks.

You can create a new role (for example `Tenant` Role) with the recommended permissions and assign them to the users that you want to assign to the Tenant team.

For example, a user named **Tenant User**, who you want to create as a tenant user, should be added or linked to the team selected as `Tenant-L01` and role assigned as `Tenant Role`:



Invoking playbooks using an alias

Release 7.3.1 adds support for invoking playbooks using an alias in a shared tenancy model, which enables you to add playbook aliases and create agnostic playbooks even in the case of shared tenants. For more information about using aliases and managing playbook mappings, see the Managing Playbook Mappings topic in the Distributed Tenancy Support chapter.

Deboarding tenants

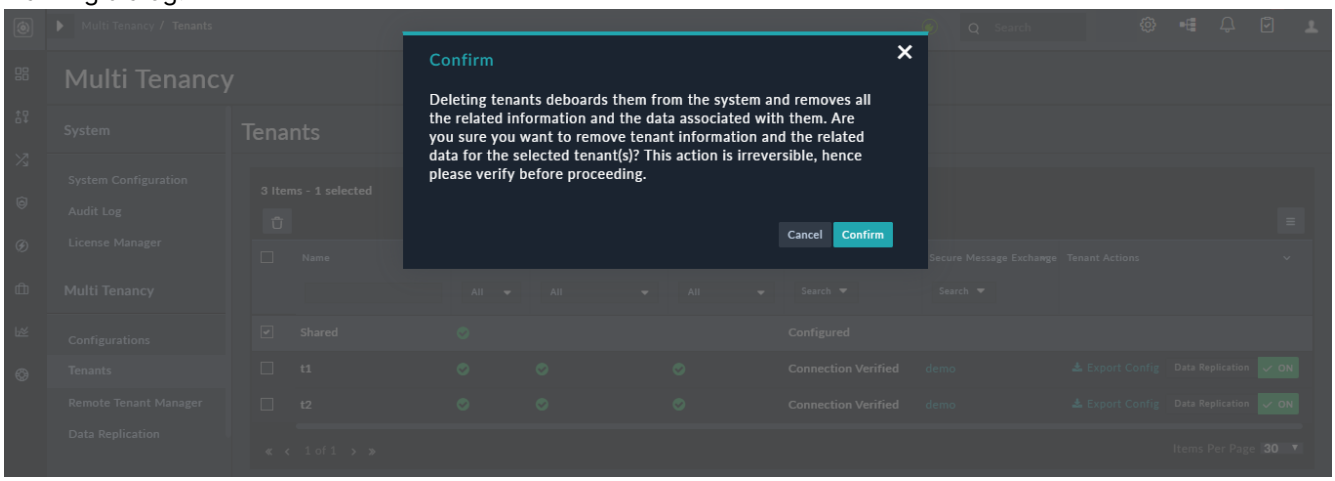
You might want to deboard existing tenants due to the following reasons:

- Tenant has not been configured correctly and you are required to remove the tenant and then add the tenant back.
- You (service provider) no longer manages the tenant.

To deboard existing tenants, you require to have `Read` and `Delete` permissions on the `Tenants` module. Deboarding tenants not only deletes the tenant from the master node, but also removes all information and data associated with that tenant from the master node. Once you delete a tenant, you cannot retrieve any information related to that tenant, therefore you must be careful while performing this operation.

To deboard a tenant, log on to FortiSOAR as an administrator and click the **Settings** icon to open the `System` page. Click **Tenants** in the left menu and on the `Tenants` page, click **Delete**. FortiSOAR will display the following

warning dialog:



To deboard the tenant, click **Confirm** on the warning dialog.

Once you deboard a dedicated tenant, the associated dedicated agent is also removed and other agents unlink themselves from the tenant.

Example of editing the templates to add the Tenant field

In the ingestion playbooks, you must ensure that the tenant field is correctly set on the created records. While creating records manually, you must take care that the tenant field is defined. If no tenant is defined, the 'self' tenant gets associated with the record.

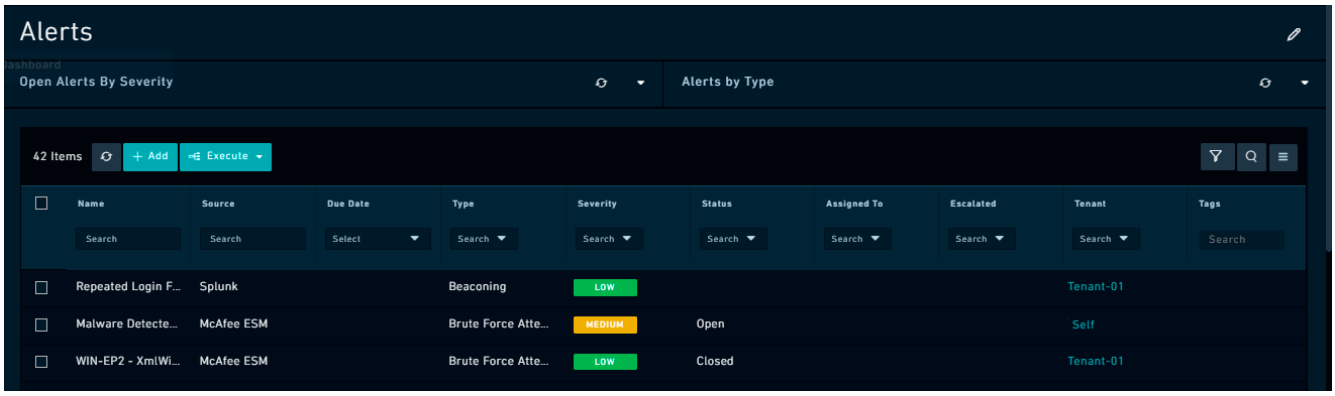
You (administrators of the master team) should edit the templates (SVT), for the modules in which tenants will create records, for example, **Alerts**, to include the **Tenant** field when a user is creating a record.

To edit the SVT, navigate to the module for which you want to update the SVT, for example, `Alerts`, open a record and click **Edit Template**. On the `Template Editing Mode Enabled` page, in the appropriate widget, such as `Form Group Details` click **Edit**, add the `Tenant` field, and then click **Add**. Click **Save** to save your changes to the SVT.

When an alert from a tenant is added, for example **Repeated Login Failures**, then the **Tenant** field the name of tenant associated with this record gets populated, thereby enabling the master to easily identify which records belong to which tenant. If no tenant is assigned while adding the alert, i.e., if the alert is created on the master, then that record is created as a `Self` record, i.e., **Self** will appear in the Tenant column.

Once a record is created, from version 7.0.2 onwards, if the value of the tenant associated with a record is set to "Self", then you are allowed a one-time edit of the tenant field. This is because at the time of record (alert) creation, the tenant to be mapped to the alert is not uniquely identifiable. Only after extraction and enrichment of the alert can the tenant be correctly determined.

The following image displays a view on the master with alert records from different tenants:



Now the master can perform all required actions such as running investigative playbooks, on this alert record and provide cybersecurity management to the tenant.

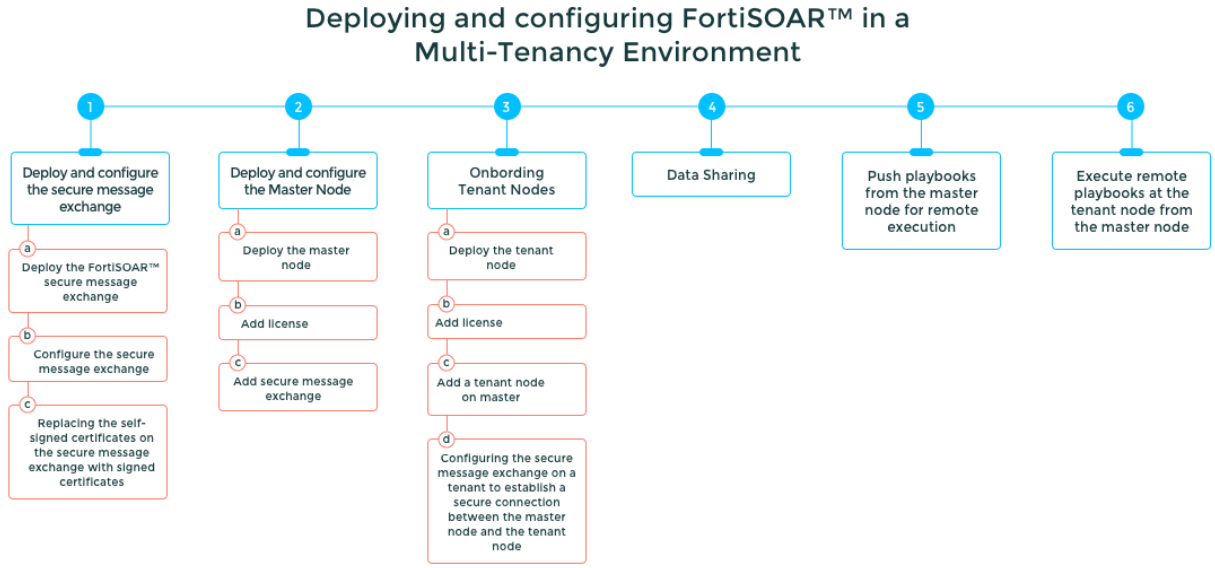
Distributed Tenancy Support

FortiSOAR introduced a distributed, multi-tenant deployment model especially tailored for Managed Security Service Providers and organizations with Globally Distributed SOC teams. In the case of the distributed tenancy model, tenants are remote, i.e., the master and each of the tenants have their own FortiSOAR instance.

Benefits of the FortiSOAR Distributed Multi-Tenancy Model

- **Autonomous Tenant Nodes**: Tenant nodes work independently.
- **Secured Communication**: Data and actions are exchanged securely.
- **Privacy and Integrity of Data**: Tenant's data remains in their environment, and they control how much data they want to share with the master node. All sensitive information stays with the tenant node. Since the actual workflow execution happens at the tenant node itself, the master node requires only the summary of information to help identify what investigations are to be run.
- **Diversity Handling**: Multiple tenants' having their unique procedures, and SLAs can be handled.
- **Simplified Remediation**: Use standard playbooks to triage or remediate incidents on the tenant node from the master node, and the procedures look simplified and manageable at the master node.
  For example, if you have a Phishing Email Alert, the analyst at the master node just invokes a generic “Investigate Phishing” workflow. This automatically resolves and executes the procedure at the tenant node the alert came from. Hence, the workflow run aligns with the tools and procedures on every tenant.
- **Scalability**: As data primarily resides at the tenant nodes, and the computations also happen at the tenant, a single console at the master node can easily handle multiple customers.
- **No overhead on Network Infrastructure**: The FortiSOAR distributed managed security service provider model does not need a VPN setup between the customer and service provider environment. Both tenants, as well as master node only, need an outbound TCP connectivity to the secure message exchange. Hence the extra network setup overhead and switching of VPNs is avoided.

Deploying and configuring FortiSOAR in a multi-tenancy environment



1. Deploy and configure the Secure Message Exchange.
   a. Deploy the FortiSOAR Secure Message Exchange.
   b. Configure the Secure Message Exchange.
   c. Replacing the self-signed certificates on the secure message exchange with signed certificates.
2. Deploy and configure the Master Node.
   a. Deploy the master node.
   b. Add license: See the *Licensing for Multi-Tenancy* section in the Overview chapter.
   c. Add Secure Message Exchange.
3. Onboarding Tenant Nodes.
   a. Deploy the tenant node.
   b. Add license: See the *Licensing for Multi-Tenancy* section in the Overview chapter.
   c. Add a tenant node on master.
   d. Configure the secure message exchange on a tenant to establish a secure connection between the master node and the tenant node.
4. Data Replication.
5. Push playbooks from the master node for remote execution.
6. Execute remote playbooks at the tenant node from the master node.

Deploying and configuring the FortiSOAR Secure Message Exchange

A secure message exchange establishes a secure channel using which you can relay information to your agent or tenant nodes.

Contact FortiSOAR Support to provide you with secure message exchange appliance that must be installed in addition to the FortiSOAR OVA on a different Virtual Machine (VM) from your FortiSOAR VM.

You do not require any additional licensing for the FortiSOAR secure message exchange.

From version 7.0.2 onwards, you can use client certificate based authentication to create connections between the distributed tenants and the secure message exchange. Prior to the 7.0.2 release, basic authentication using username and password was used to create connections between distributed tenants and secure message exchange. Going forward, you can configure the following types of authentication to connect distributed tenants and secure message exchange:

- **Basic Authentication**: Uses username and password to create connections between distributed tenants and secure message exchange.
- **Basic Authentication with Peer Verification**: Uses username and password to create connections between distributed tenants and secure message exchange, and also performs 'Certificate Verification'. This process will verify that the clients which are attempting to connect can be trusted by presenting a certificate that is signed by a CA and trusted by the server; thereby ensuring that only trusted clients can connect to the secure message exchange.
- **Client Certificate Authentication**: Presents a certificate to the server which is signed by a trusted CA, and which uses the Common Name (CN) as the username when an agent or tenant tries to connect with the secure message exchange. It is recommended that you create the certificate with the common name as the name of your agent or tenant. This provides enhanced security as this gives the facility to connect only to trusted clients.

To enable client certificate authentication, you can specify the authentication type as '**Certificate Auth**' while adding a distributed tenant.

To enforce client certificate verification, you must provide a pair of exchange event listener client certificates and exchange event listener client key when you are adding a secure message exchange. Client verification ensures that whenever any client wants to connect to secure message exchange that client must present the client certificate to the secure message exchange for verification. You must also provide the pair of exchange event listener client certificates and exchange event listener client key if you have enabled mutual TLS (mTLS). Use the `csadm mq mtls` command to enable or disable mTLS. For more information on `csadm`, see the *FortiSOAR Admin CLI* chapter in the "Administration Guide.

Deploying the FortiSOAR Secure Message Exchange

1. Deploy the FortiSOAR Secure Message Exchange, which is the same as deploying a FortiSOAR Virtual Appliance. See the *Deploying FortiSOAR* chapter in the "Deployment Guide" for detailed information on how to deploy the FortiSOAR Virtual Appliance. Steps are as follows:

a. Review and ensure that you meet the recommended resources required to deploy the FortiSOAR secure message exchange (same as FortiSOAR Virtual Appliance).
b. Import the FortiSOAR secure message exchange (same as FortiSOAR Virtual Appliance).
c. Deploy the FortiSOAR secure message exchange using vSphere/vCenter or AWS (same as FortiSOAR Virtual Appliance).

Configuring the Secure Message Exchange

After you have completed deploying the secure message exchange appliance, and when you log in to the secure message exchange appliance using `ssh` for the first time, the **FortiSOAR Secure Message Exchange Configuration Wizard** is displayed. The wizard guides you through the RabbitMQ configuration process with appropriate instructions so that you can efficiently perform the configuration required for RabbitMQ.

The wizard performs the following configuration steps:

1. **Hostname Change (Optional)**: You can change the hostname for your RabbitMQ VM. The wizard checks if the hostname is valid or not; and throws an error in case of an invalid hostname.
   You must change the hostname to a resolvable hostname using which the master and tenant nodes can connect to the secure message exchange.
2. **Add DNS Name or IP Address (Optional)**: You can change the DNS Name or IP address for your RabbitMQ VM.
3. **Add Username (Optional)**: Username that you will use to connect to the RabbitMQ management console. If you do not specify any username, then by default, the username is set as `admin`.
4. **Add Password**: Password that you will use to connect to the RabbitMQ management console. You must mandatorily specify the password.
   Note: Ensure you retain a copy of this password. FortiSOAR does not save the password in any file on the secure message exchange for security reasons. Therefore, this password cannot be recovered.
5. **Add SSL port (Optional)**: Specify an SSL port between `49152` and `65535` for RabbitMQ management console. By default, this port is set as `15671`.
6. **Add TCP port (Optional)**: Specify the management TCP port between `49152` and `65535`. By default, this port is set as `5671`.
   Once you specify the above parameters, the FortiSOAR Secure Message Exchange Configuration wizard gets ready to configure your secure message exchange, which includes generating Self-Signed Certificates for RabbitMQ.
   Important: You get logged out after the FortiSOAR Secure Message Exchange is configured so that the changes can take effect. Therefore, you are required to `ssh` again to the FortiSOAR Secure Message Exchange VM.
   Note: The configuration log for FortiSOAR Secure Message Exchange is located at `/var/log/cyops/install/config_vm_<timestamp>`.
   For example, `/var/log/cyops/install/config-vm-30_Oct_2018_12h_50m_38s.log` `
   The FortiSOAR Secure Message Exchange Configuration Wizard also displays the path of the Secure Message Exchange configuration log.

Replacing the self-signed certificates on the secure message exchange with signed certificates

It is highly recommended that the certificates used for encrypted communication between the FortiSOAR nodes and the secure message exchange should be signed from the Certificate Authority.

To replace the self-signed certificates on the secure message exchange with signed certificates, do the following:

1. Replace the following files on your FortiSOAR secure message exchange with the corresponding signed files from the Certificate Authority:
   - CA Certificate: `/opt/cyops/configs/rabbitmq/ssl/cyopsca/cacert.pem`
   - Server Certificate: `/opt/cyops/configs/rabbitmq/ssl/server/cert.pem`
   - Service Private Key: `/opt/cyops/configs/rabbitmq/ssl/server/key.pem`
     Note: A `.key` file has the path to a PEM encoded file containing the private key. A `.pem` file has the path to a PEM encoded file containing the certificate (or certificate chain) that will be presented when requested.
2. Restart the RabbitMQ server and all its related services using the following command:
   `# systemctl restart rabbitmq-server`.

Deploying the Master Node

See the *Deploying FortiSOAR* chapter in the "Deployment Guide" for detailed information on how to deploy the FortiSOAR Virtual Appliance. Steps included in deploying the master node:

1. Review and ensure that you meet the recommended resources required to deploy the FortiSOAR Virtual Appliance.
2. Import and deploy the FortiSOAR Virtual Appliance using vSphere/vCenter or AWS.
3. FortiSOAR Configuration Wizard
   FortiSOAR Configuration Wizard runs automatically on the first `ssh` login by the `csadmin` user and performs the initial configuration steps that are required for FortiSOAR and performs tasks as is the case with the FortiSOAR enterprise OVA. It generates a Device UUID for your FortiSOAR instance. Use this Device UUID to generate the FortiSOAR license in FortiCare so that you can begin using FortiSOAR. Ensure that you generate a license that has its edition set to "Multi-Tenant" and role set to "Manager".

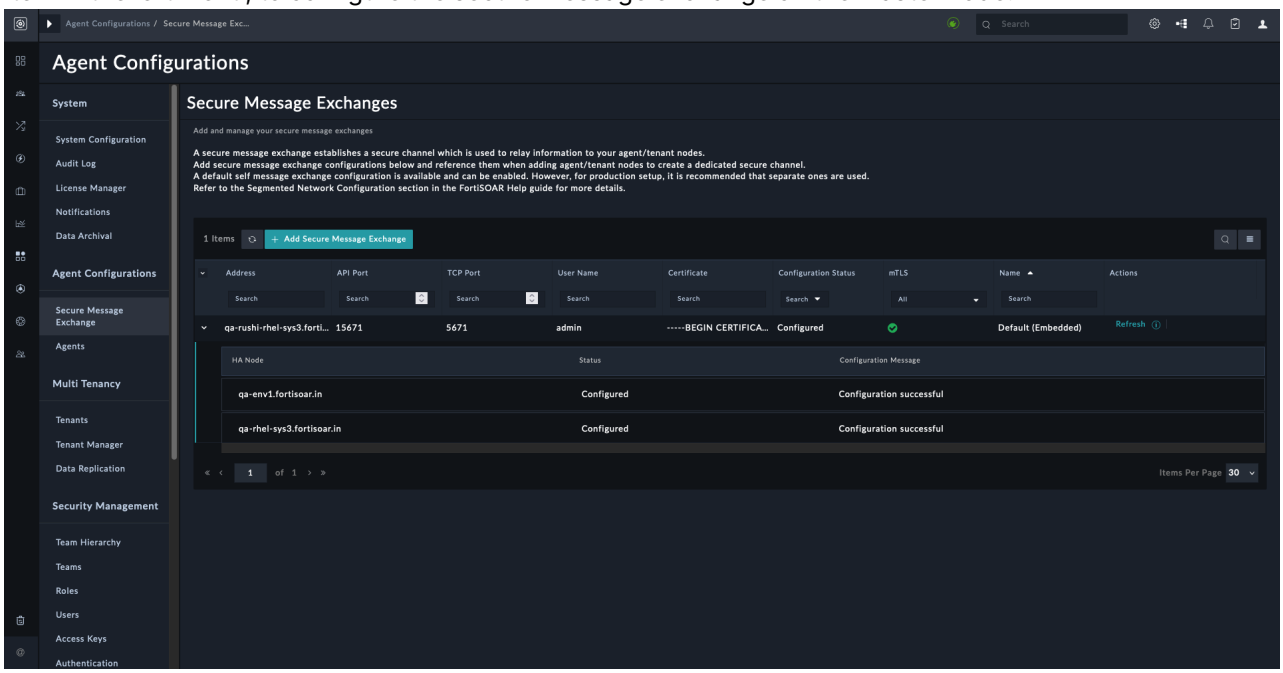
Adding the secure message exchange on the master node

Only if your FortiSOAR license has been enabled for multi-tenancy you will see a **Multi Tenancy** section on your `System` page.

To add the secure message exchange on the master node, do the following:

1. Log on to FortiSOAR as an administrator.
2. Click the **Settings** () icon to open the `System` page.

3. On the `System` page, you will see the `Agent Configurations` section. Click the **Secure Message Exchange** item in the left menu, to configure the secure message exchange on the master node.



4. Add a secure message exchange to the master node and add the reference of this secure message exchange in the tenant or agent node(s) to create a dedicated secure channel of communication.
   You can have more than secure message exchange in the configuration. You can distribute tenants across secure message exchanges based on the geographical locations, scale, or compliance policies of the respective customers.
   Note: You should have already configured the secure message exchange using the steps mentioned in Configuring the Secure Message Exchange.
   To add a secure message exchange, click **Add** on the `Secure Message Exchanges` page.
   Important: To add a secure message exchange and configure tenants, you must have a role that has a minimum of `Create`, `Read`, and `Update` permissions on the **Secure Message Exchange** and **Tenants** modules.
   To edit the configuration of an existing secure message exchange, click the secure message exchange row whose configuration you want to update. This displays the `Edit Secure Message Exchange` dialog. Update the configuration parameters, as required, in the dialog and click **Update**.
5. In the `Add New Secure Message Exchange` dialog, configure the following parameters:
   a. In the **Name** field, enter the name of the secure message exchange that you have configured to act as a secure channel of data replication between the master and tenant nodes.
   b. In the **Address** field, enter the FQHN (Fully Qualified Host Name) of the secure message exchange.
      Important: Ensure that the FQHN matches the Certificate Name (CN) or the Subject Alternative Name (SAN) provided in the SSL certificate used to configure the secure message exchange.
   c. In the **Username** field, enter the username you will use to login to your secure message exchange as an administrator.
      By default, it is set as `admin`.
   d. In the **Password** field, enter the password you will use to login to your secure message exchange as an administrator.

e. In the **Server Name Indication** field, enter the Server Name Indication (SNI) address for the Secure Message Exchange. You must specify the SNI address when the Secure Message Exchange is behind a reverse proxy or in a cluster behind a load balancer such as FortiADC.

f. In the **API Port** field, enter the RabbitMQ Management port number that you had specified while configuring the secure message exchange, and ensure that the master node has outbound connectivity to the secure message exchange at this port.
By default, it is set as `15671`.

g. In the **TCP Port** field, enter the TCP port number that you had specified while configuring the secure message exchange, and ensure that the master node has outbound connectivity to the secure message exchange at this port.
By default, it is set as `5671`.

h. In the **CA Certificate** field, copy-paste the certificate text of the Certificate Authority (CA) that has signed the secure message exchange certificate in the `pem` format. If it is a chain, then the complete chain must be provided.
By default, the CA certificate for the FortiSOAR self-signed certificate is present at the following location: `/opt/cyops/configs/rabbitmq/ssl/cyopsca/cacert.pem`.
Important: If in the future, your secure message exchange certificate expires, and you need to deploy a new certificate, then the new certificate must be copied back to the master node as well as the tenant's router entry.

i. (Optional) If you have enabled mTLS, i.e., you require that clients that want to connect to secure message exchange must present the client certificate to the secure message exchange for verification, then you must also provide a pair of exchange event listener client certificates and exchange event listener client key, as follows:

   i. In the **Exchange Event Listener Client Cert** field, copy-paste the client certificate text or you can also upload the client certificate file.

   ii. In the **Exchange Event Listener Client Key** field, copy-paste the client key text or you can also upload the client key file.
   Note: If you have enabled mTLS on secure message exchange, and you have added the secure message exchange client certificate and key after the tenant is added or if you have updated the secure message exchange client certificate and key after they have expired, then you require to first disable and again enable the agent to re-trigger the event listener and update agent status correctly.

j. To save the configuration for the secure message exchange on the master node, click **Create**.

A default "Self" entry is created for every master node and you can directly link agents to the master node. For more information on agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide."

Onboarding tenant nodes

From version 6.4.1 onwards, a default agent is automatically created and added to a dedicated tenant as part of the tenant creation process. For more information on agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide." You can add multiple agents to a tenant, therefore, tenants become a wrapper that can contain various agents that can connect to various disparate networks and execute actions remotely.

Deploying tenant nodes

Deploy tenants as required. See the *Deploying FortiSOAR* chapter in the "Deployment Guide" for detailed information on how to deploy the FortiSOAR Virtual Appliance. Steps included in deploying a tenant node:

1. Review and ensure that you meet the recommended resources required to deploy the FortiSOAR Virtual Appliance.
2. Import and deploy the FortiSOAR Virtual Appliance using vSphere/vCenter or AWS.
3. FortiSOAR Configuration Wizard
   FortiSOAR Configuration Wizard runs automatically on the first `ssh` login by the `csadmin` user and performs the initial configuration steps that are required for FortiSOAR and performs tasks as is the case with the FortiSOAR enterprise OVA. It generates a Device UUID for your FortiSOAR instance. Use this Device UUID to generate the FortiSOAR license in FortiCare so that you can begin using FortiSOAR. Ensure that you generate a license that has its edition set to "Multi-Tenant" and role set to "Tenant".

Adding a tenant node on the master

On the top of the Tenants page, you will see the tenant ID of the master node, and also see Licensed as Master node text. You must use the tenant ID of the master when you are adding details of the master node on the tenants' node.

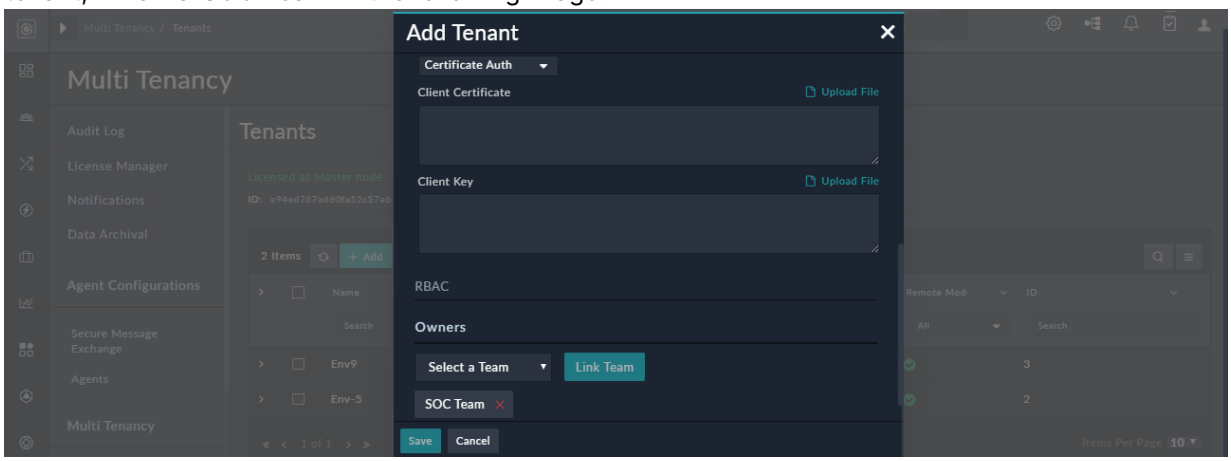
1. Log on to your FortiSOAR master node as an administrator and click the **Settings** icon to open the `System` page.
2. To add tenants associated with a master node, in the `Multi Tenancy` section, click **Tenants** in the left menu and on the `Tenants` page, click **Add**.
   To edit the configuration of an existing tenant, click the tenant whose configuration you want to update, which opens the agent record in the detail view. Update the configuration parameters as required, and click **Save**.
   If you no longer require an existing tenant you can deboard that tenant. Deboarding a tenant is an irreversible operation which also deletes all data related to that tenant from the master node. For more information see the *Deboarding Tenants* section in the Shared Tenancy chapter.
   You can deactivate agents associated with the tenants. To deactivate an agent, open the agent record, and clear the **Enabled** checkbox in that record.
3. In the `Add Tenant` dialog, configure the following parameters:
   a. From the **Choose Tenant Type** field choose the type of tenant you want to add.
      In this case, select **Dedicated** to add this tenant as a remote or dedicated tenant. Remote or Dedicated tenants have their own FortiSOAR instance and do not share the current FortiSOAR instance with the Master.
   b. In the **Tenant ID** field, enter the ID of the tenant, which is the Device UUID of the tenant's FortiSOAR instance.
   c. In the **Tenant Name** field, enter the name of the tenant.
   d. (Optional) In the **Description** field, enter the description of the tenant.
   e. From version 6.4.1 onwards, a default agent is automatically created and added to a dedicated tenant as part of the tenant creation process. For more information on agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide."
      i. From the **Secure Message Exchange** drop-down list, choose the secure message exchange that you have configured as the secure channel using which you can relay information to your agent or tenant nodes.

ii. From the **Auth Type** drop-down list, select the type of authentication you want to enforce for tenants or clients to connect to the secure message exchange

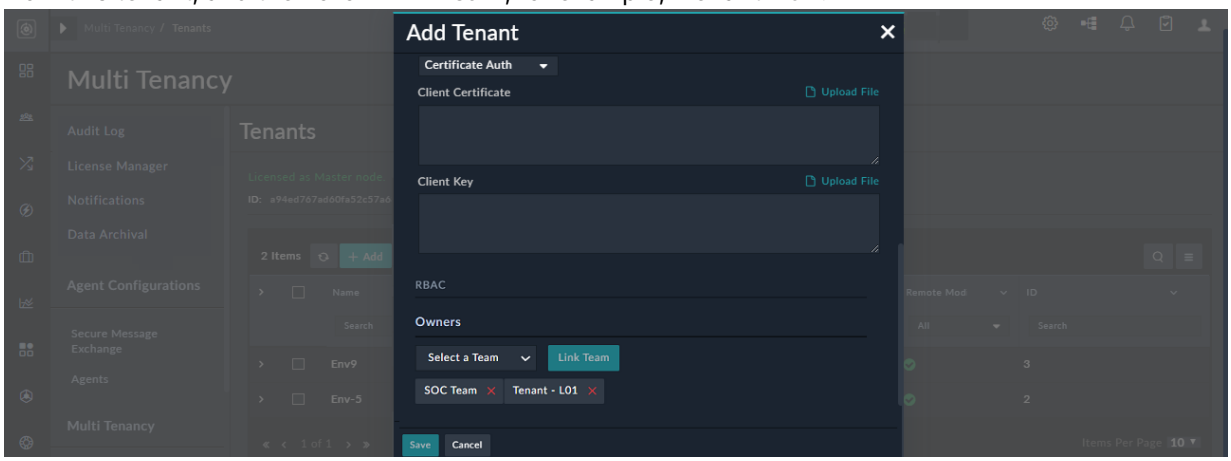
iii. If you have selected **Certificate Auth** from the **Auth Type** drop-down list, then in the **Client Certificate** field, copy-paste the client certificate text of the Certificate Authority (CA) that has signed the secure message exchange certificate in the `pem` format. You can also upload the client certificate file.
If you want to enforce client certificate verification with **Basic Auth** then also you must provide client certificate in this field, so that the secure message exchange will verify the certificate before allowing connection to any client.
Note: If you are using CA signed certificates, you must add these certificates to the truststore. For more information, see the *FortiSOAR Admin CLI* chapter in the "Administration Guide."

iv. Similarly, in the **Client Key** field, copy-paste the client key text or you can also upload the client key file.

f. In the **Owners** section, the teams of the logged-in user are listed as pre-selected as the owner of the tenant, which is 'SOC Team' in the following image:



Select the other teams that you want to add as owners of the tenant/agent and the records originating from this tenant, and then click **Link Team**, for example, 'Tenant-L01'.



The teams that you have selected are listed as owners of any records created for that tenant. As shown in the following image, the 'SOC Team' and the 'Tenant L-01' team are assigned to any records that originates from this tenant.
You can also delete the default team as the owner and replace it with another team. However, to create a tenant, you must designate at least one team as the owner. If you remove all teams from the Owners section, the following warning message is displayed: '*As a best practice, ensure that you*

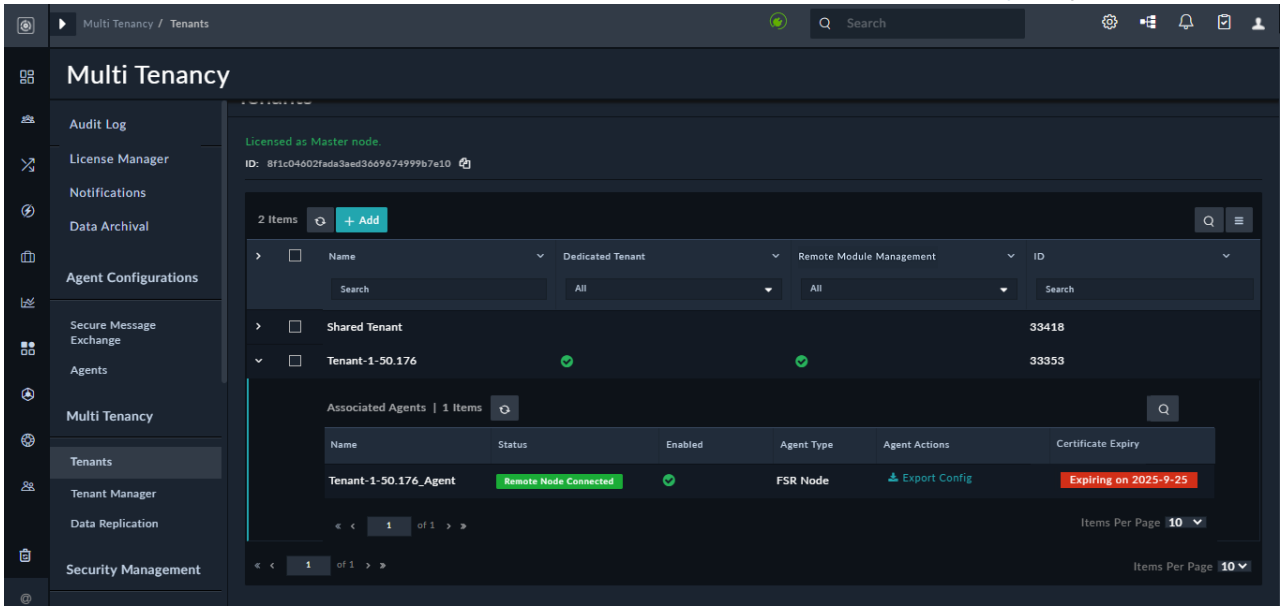
associate at least one of your teams with this tenant/agent else you will no longer able to access the tenant/agent settings or associated records after tenant/agent creation'

NOTE: When a record is created on the master and a tenant is selected during record creation, all teams of the user who created the record receive ownership, along with teams of the tenant. From release 7.6.0 onwards, you can restrict record ownership to only teams of the tenant by changing the value of the '`restricting_record_owners_to_tenant`' parameter to '`true`' in the `/opt/cyops-api/config/parameters_prod.yaml` file and saving the file. After saving the file, run the following command:

```
systemctl restart php-fpm && sudo -u nginx php /opt/cyops-api/bin/console cache:clear && systemctl restart php-fpm.
```

4. To complete adding a new tenant associated with the master node, click **Save**.

   As tenants as just a wrapper, there will be no configuration state associated with a tenant. However, in case of dedicated tenants, a default agent is automatically created and added to a dedicated tenant as part of the tenant creation process. Therefore, you can expand the row of the tenant and check the connection between the agent and the master node using secure message exchange. Once the connection is established from the master node to the agent, the **Status** field displays "Remote Node Connected". To know more about the statuses, see the *List of Statuses* section in the Shared Tenancy chapter.



You can export the configuration of an agent in the `JSON` format using the **Export Config** link, in the agent row that represents the dedicated node and therefore contains the configuration of the tenant. This `JSON` file containing the configuration of the tenant can then be imported on the `Master Configuration` page of the tenant node on which you are configuring the secure message exchange, by clicking the **Import Master Config** link.

You might also see a **Warning** symbol in the `Agents Action` column, if the master node cannot remotely execute or manage connector actions on the agent node.

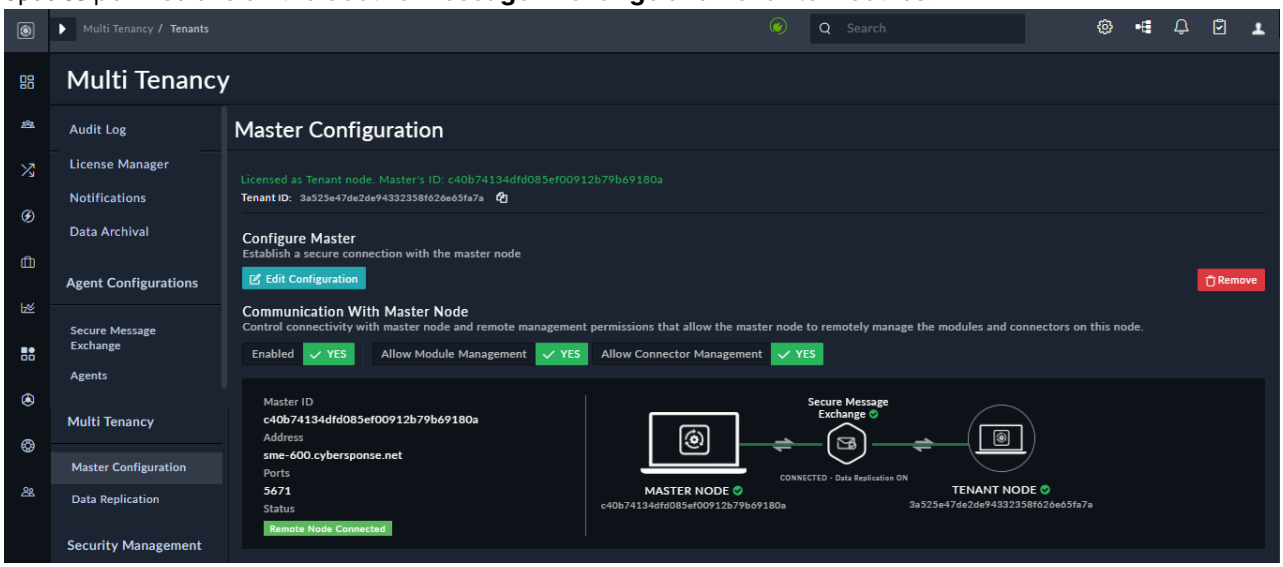
The tenant's agent grid contains a **Certificate Expiry** column, which displays when the client certificate will expire in case of Certificate Authentication or Basic Authentication with Peer Verification. Whereas it will display a blank in case of Basic Authentication.

You can add multiple agents to a tenant by linking the respective agents to the tenant by opening the detail view record of the tenant and in the `Owners` section, click the **Agents** tab, and click **Link** to open the `Link Agents` dialog that contains a list of configured agents. Select the agent that you want to associate with the tenant and

click **Save Relationship**. Note that the linking of agents to tenants is just representational, and RBAC is governed by the "Owners" that you specify for the agent.

Configuring the secure message exchange on a tenant to establish a secure connection between the master node and the tenant node

1. Log on to your FortiSOAR tenant node as an administrator and click the **Settings** icon to open the `System` page.
2. On the `System` page, you will see the `Multi Tenancy` section. Click the **Master Configuration** item on the left menu, to configure your tenant node.
   On the `Master Configuration` page, you will see the tenant ID of the tenant. You must use the Tenant ID of the tenant when you are adding details of the tenant node on the master node. You will also see the text such as: `Licensed as Tenant node: Master's ID: <ID value>`
   `Tenant ID: <ID value>`.
   `Tenant ID` is the Device UUID of the tenant.
   `Master's ID` is the Device UUID of the master's FortiSOAR instance.
   Important: To configure the master node, you must have a role that has a minimum of `Create`, `Read`, and `Update` permissions on the **Secure Message Exchange** and **Tenants** modules.



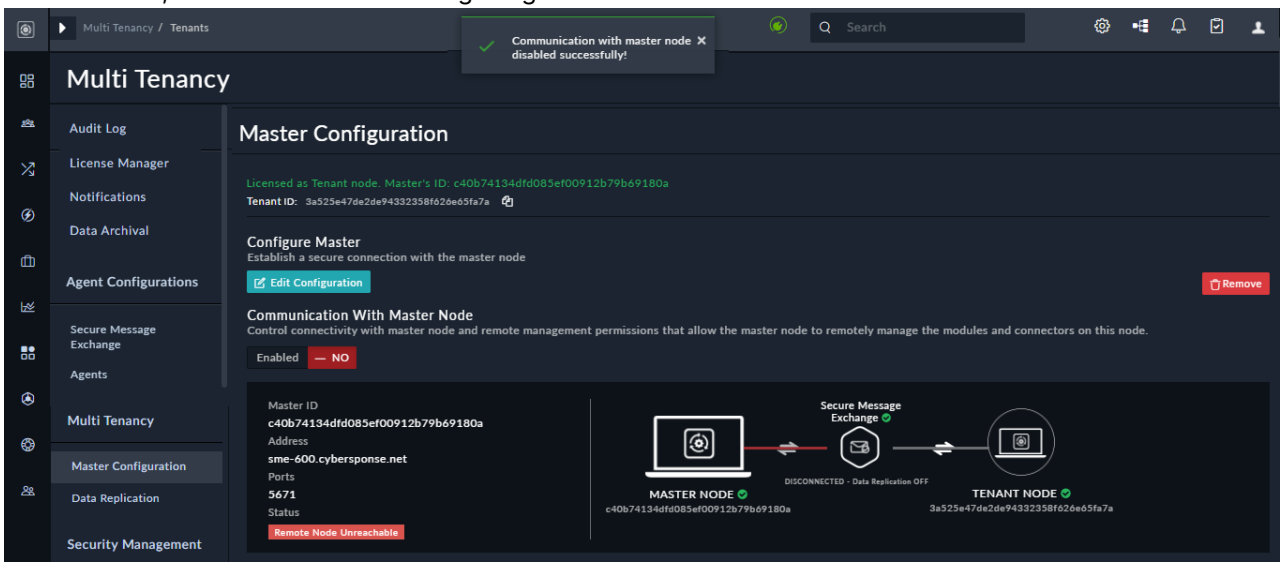
The `Configure Master` section displays the master and secure message exchange configuration details, and the status of configuration. You can see the following labels, in the `Status` field and in the image that displays the connection between the master and the secure message exchange. These labels signify the connection status between the master and secure message exchange:

- **Not Configured**: Secure Message Exchange has not been configured.
  This will display **Inactive** in the image displaying the connection between master and secure message exchange.
- **Configuration In Progress**: Configuration of the secure message exchange is in process.
  This will display **Inactive** in the image displaying the connection between master and secure message exchange.
- **Configured**: Connection between the master and secure message exchange is established without errors.
  This will display as **Connected - Data Replication ON** in the image if data replication is on from this

tenant node to the master node.
This will display as **Connected - Data Replication OFF** in the image if data replication is turned off from this tenant node to the master node.

- **Configuration Failed**: Connection between the master and secure message exchange is not established, due to either, wrong configuration or network failure.
This will display **Disconnected** in the image displaying the connection between master and secure message exchange.

3. (Optional) To stop communication and replication of data, from this tenant to the master, click the **YES** button appearing beside **Enabled**. Disabling stops the master from receiving any data from this tenant and now the master cannot remotely manage modules and connectors of this tenant and you will see **NO** in the **Enabled** field, as shown in the following image:

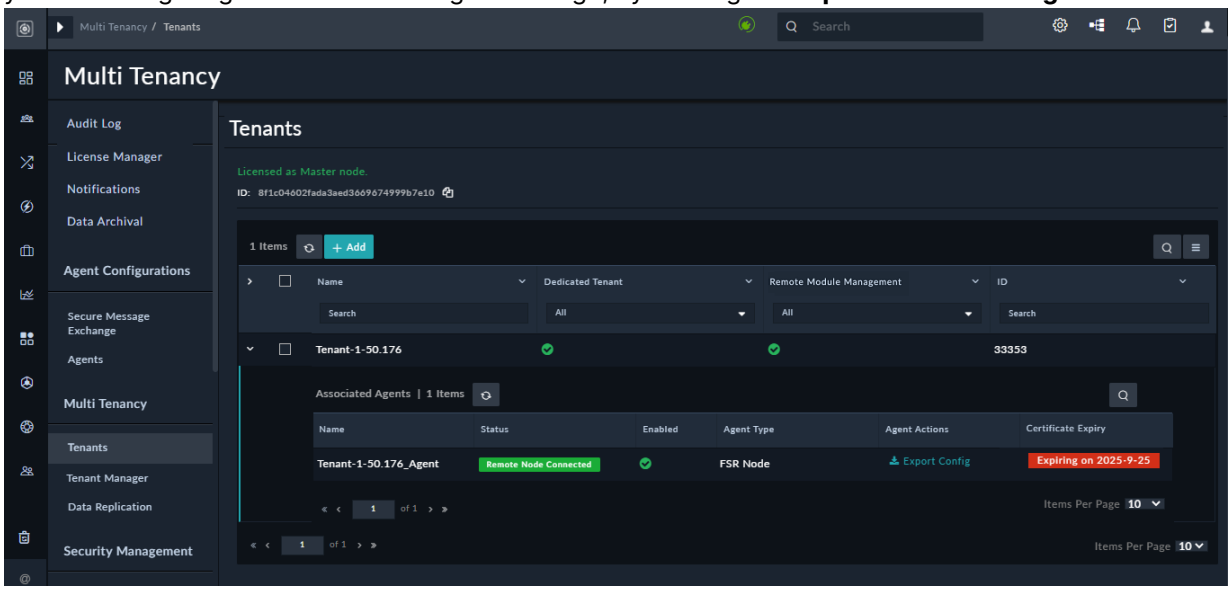


This global enable/disable button stops the communication from the tenant to the secure message exchange, and the tenant can no longer send data. However, data coming from the master is yet available at the secure message exchange, and once the tenant node resumes communication, the data from the master is available for consumption by the tenant. The tenant node continues to receive data from the master if sharing of data is enabled from the master node to the tenant node. You can use this option to suspend external communication from the tenant to the secure message exchange/master in cases such as, a planned downtime for regular maintenance. Once such activity is completed, the tenant node can enable the communication again and replay the stored messages.

4. (Optional) To disallow the master to push picklists and make changes to its (the tenant's) model metadata (MMD) and push those changes to the tenant node, i.e., to disallow the master to make changes to the tenant's mmd, click the **YES** button appearing beside **Allow Module Management**, which turns this button to **NO**.
5. (Optional) To disallow the master the ability and permissions to remotely execute and manage the connector actions at the tenant node, click the **YES** button appearing beside **Allow Connector Management**, which turns this button to **NO**. For more information, see the Managing connectors of distributed tenants section.
6. (Optional) To delete the configuration of the master node from a tenant node, click **Remove**.
Note: To remove configurations of the master from the tenant node, you must have `Read` and `Delete` permissions on the `Secure Message Exchange` module.
7. To configure the secure message exchange on the tenant and establish a secure connection with the master, click **Edit Configuration** in the `Configure Master` section, and choose one of the following

methods to complete the configuration:

a. Export the configuration of the tenant in the `JSON` format, from your master node, by clicking the **Export Config** link in the agent row. The agent that has the **Export Config** link represents the dedicated node, and therefore contains the configuration of the tenant and its `Agent Type` will be set as "FSR Node". You can view the agents added to the tenant by expanding the tenant row. If the tenant has multiple agents associated with it, then only the dedicated node will have the **Export Config** link, all remaining agents will have the **Download Installer** link and their `Agent Type` will be set as "FSR Agent".
You can then import this JSON file on the `Master Configuration` page of the tenant node on which you are configuring the secure message exchange, by clicking the **Import Master Config** link.



OR

b. Add the details manually:

i. To add a secure message exchange, click **Add Master** on the `Master Configuration` page.
To edit an existing secure message exchange configuration, click **Edit Configuration** in the `Configure Master` section.

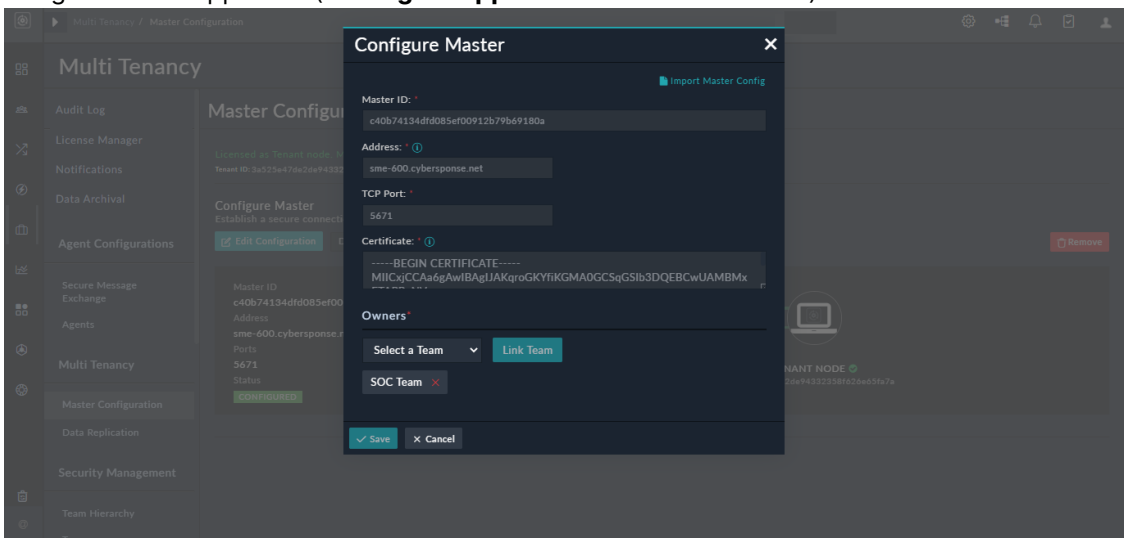
ii. In the **Configure Master** dialog, configure the following parameters:
Note: The **Master ID** is a read-only field that displays the Device UUID of the master's FortiSOAR instance.

i. In the **Address** field, enter the IP address of the secure message exchange, which should match the IP address that you have specified when configuring the secure message exchange.

ii. In the **TCP Port** field, enter the TCP port number that you had specified when configuring the secure message exchange for this tenant. By default, it is set as `5671`.

iii. In the **CA Certificate** field, copy-paste the certificate text of the Certificate Authority (CA) that has signed the secure message exchange certificate in the `pem` format. If it is a chain, then the complete chain must be provided.
By default, the CA certificate for the FortiSOAR self-signed certificate is present at the following location on the secure message exchange:
`/opt/cyops/configs/rabbitmq/ssl/cyopsca/cacert.pem`.
Important: If in the future, your secure message exchange certificate expires, and you need to deploy a new certificate, then the new certificate must be copied back to the master node as well as the tenant's router entry.

iv. From the **Owners** drop-down list, select the teams that you want to add as an owner to the records coming from this tenant, and click **Link Team**. The teams that you have selected will be associated as owners of any record created for that tenant on the master, either manually or using an Custom API Endpoint trigger.
Note: If a record is created on the master manually, then the teams that are selected in the **Owners** drop-down list will be associated as owners of that record. Similarly, if any record is created using an *Custom API Endpoint trigger*, then that record will also be owned by the teams selected in this **Owners** drop-down list, irrespective of whether these are the teams assigned to the appliance (**Settings** > **Appliances** > **Team and Role**).



v. To save the configuration for the master node on a tenant node, click **Save**.
Once you click save, FortiSOAR checks the connection between the master and secure message exchange, and if a connection can be established without errors, the status is displayed as **Connected**.

If you have your master and secure message exchange that are in one network and the tenants on another network, then you must ensure that both the master and tenant must be able to resolve the secure message exchange hostname using either a Public or Private IP.

To assist with debugging issues on the Agent, you can obtain agent logs from the master node starting with release 7.4.2, as explained in the Downloading Agent Logs topic.

Downloading Agent Logs

The ability to download Agent logs from the master node aids in debugging of Agent issues as follows:

- In many cases, it is not possible to access the Agent's CLI for debugging any issue reported for the Agent.
- If there is a connector failure on the Agent, the agent's `connector.log` is required to debug the issue.

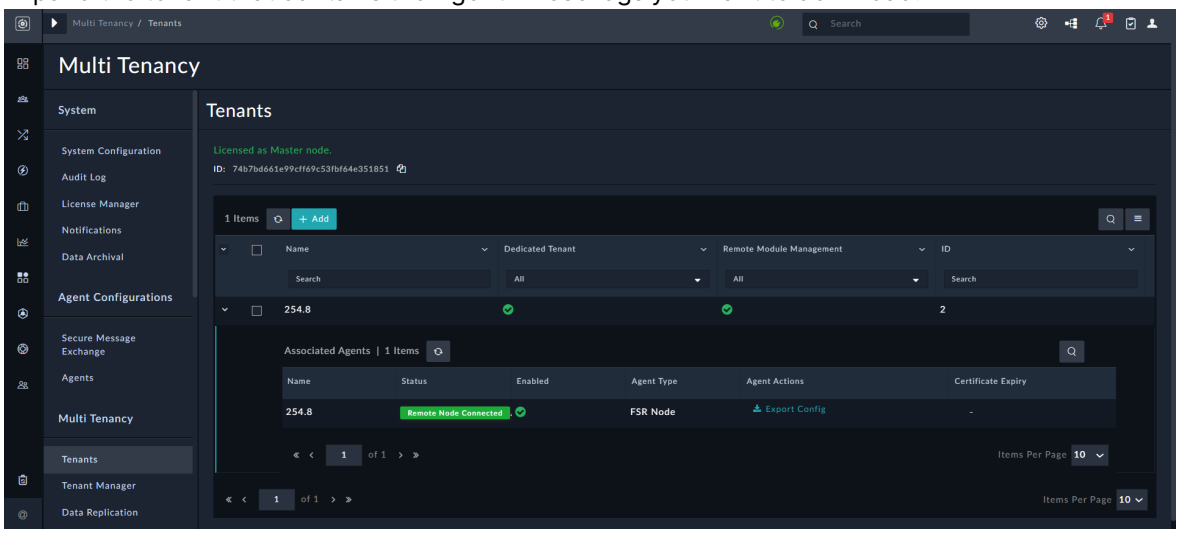
Permissions Needed:

- To initiate log collection from Agent, users must have a minimum of `Execute` permission on 'Connectors' and `Read` permission on 'Agents'.
- To download the Agent logs, users must have `Read` permission on 'Files.'

Ensure both the master and the agent are on release 7.4.2 to download 'FSR Agent' logs. The log download fails with an error such as "`For agents, log collection is accessible on version 7.4.2`" if the master node is on release 7.4.2 and the agent is on a release earlier than 7.4.2.

Users can download Agent logs as follows:

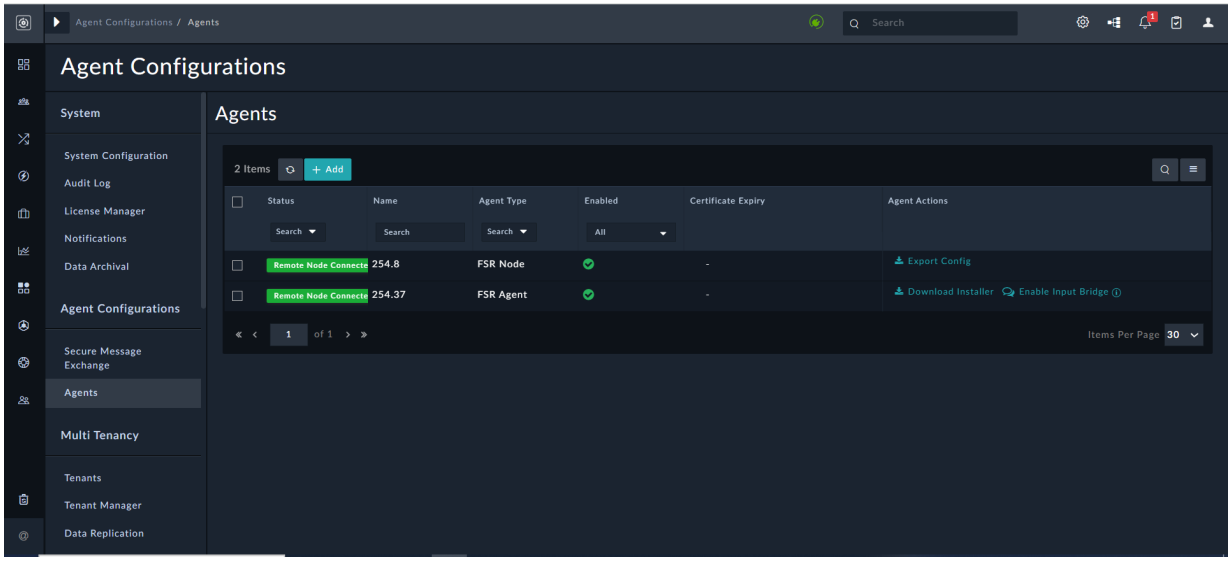
1. Log on to your master node as an administrator and click the **Settings** icon to open the `System` page.
2. Open the Agent's record whose logs you want to download using one of the following methods:
   a. In the Multi Tenancy section, click **Tenants** to open the `Tenants` page:
      i. Expand the tenant that contains the Agent whose logs you want to download.



      Note: The status of the Agent must be 'Remote Note Connected.

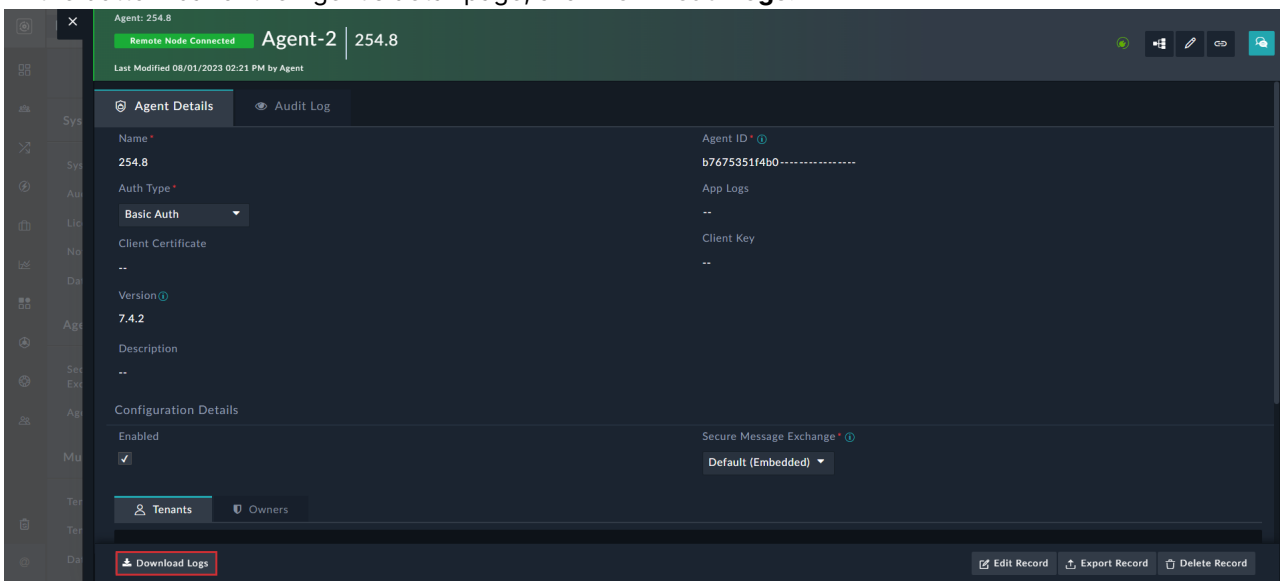
      ii. Click on the row of that Agent to open its detail page.

   b. In the `Agent Configurations` section, click **Agents** to open the `Agents` page:

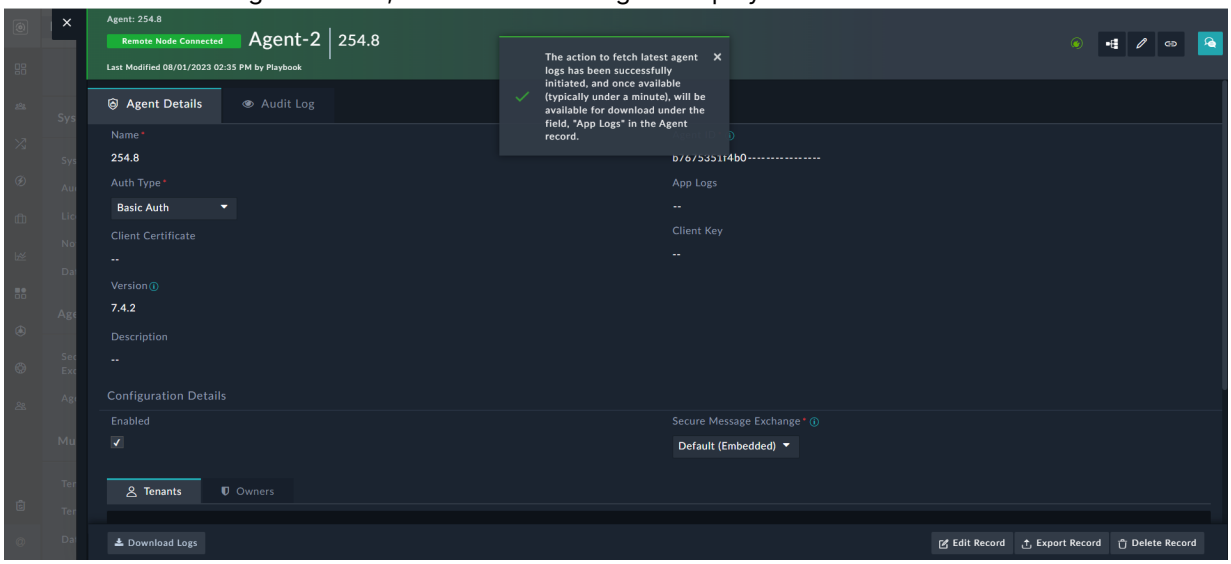


   Click the row of the Agent whose logs you want to download to open its detail page.

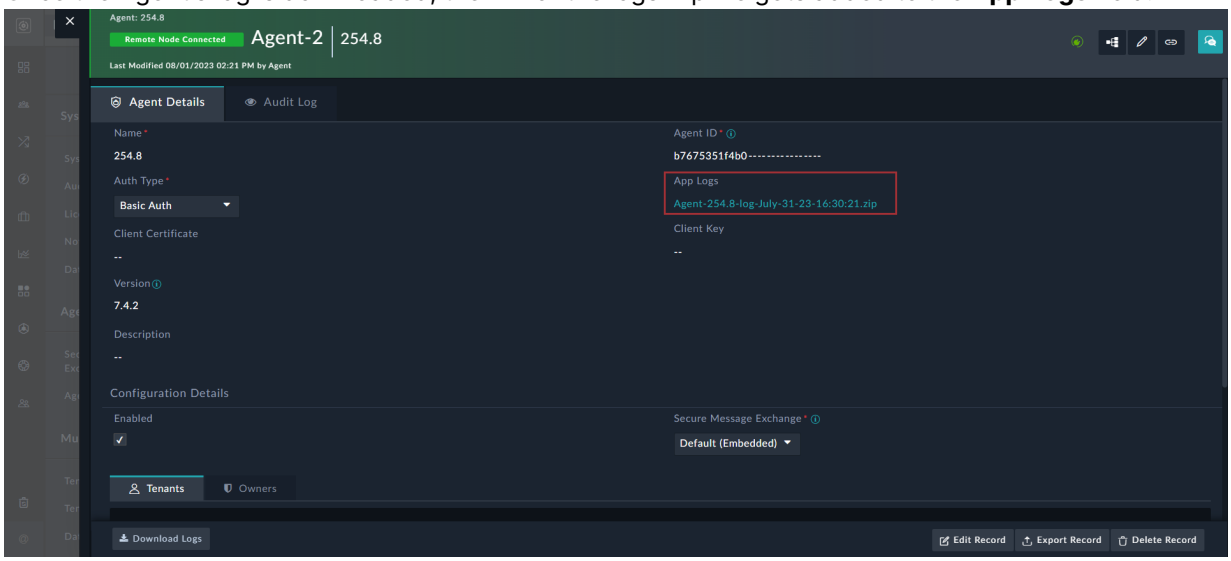
3. In the bottom bar of the Agent's detail page, click **Download Logs**:



4. Clicking **Download Logs** does the following:
   a. On initiation of the log download, a 'Success' message is displayed:



b. Once the Agent's log is downloaded, the link of the logs' zip file gets added to the **App Logs** field:



Note: The size of the log files determines the time taken to download the logs.

c. Click the log link to download the Agent logs as a zip file.

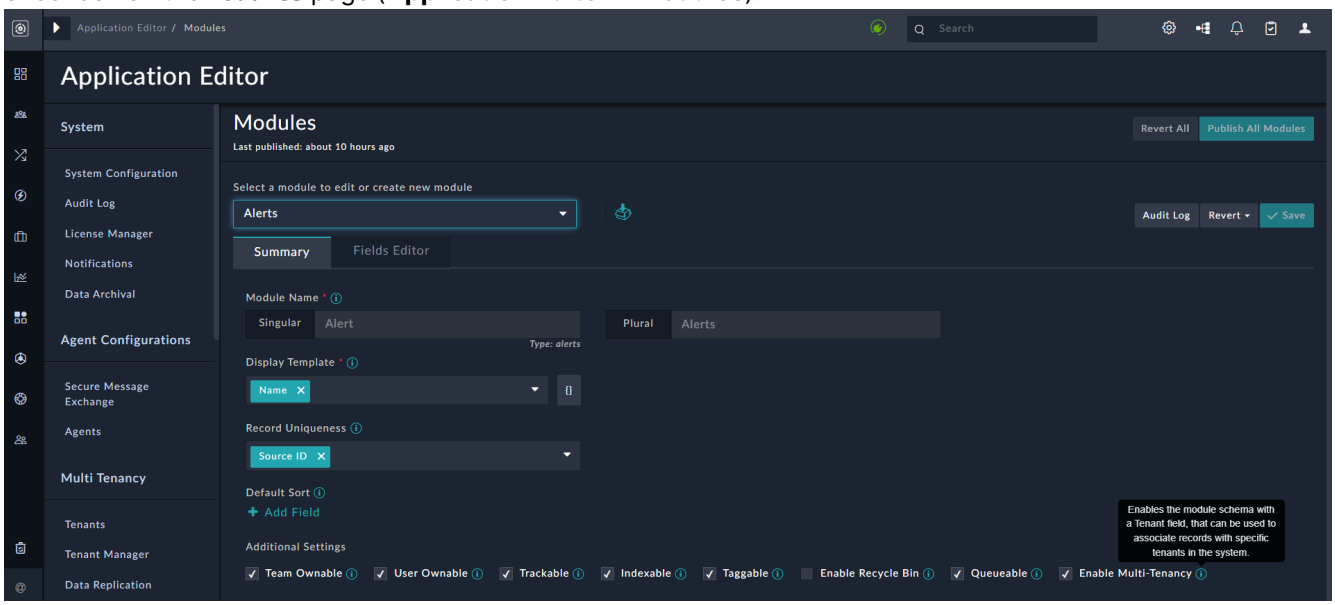
Notes on the Agent's log file:

- The logs' zip file contains all of the files and content from the `/var/log/cyops/cyops-integrations` folder, including `connectors.log`, `integrations.log`, and so on. Additionally, the `services_status.log` file that contains status of services will also be added to the logs' zip file.
- The maximum file size supported is 500 MB.
- The Download Logs action deletes the log files that were previously downloaded for the Agent, and replaces them with the most recent log files.

Replicating data between the master and the tenant nodes

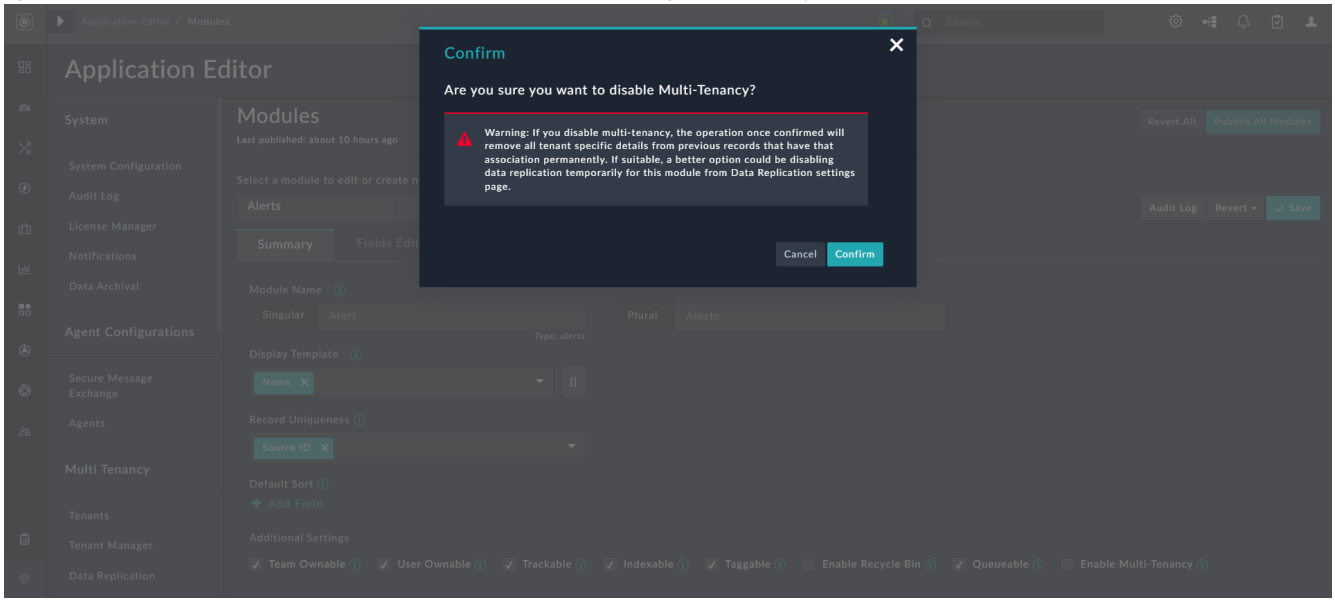
To leverage bidirectional synchronization of data for updates between the master and the tenant nodes, you must enable data replication of modules on both the master and the tenant nodes.

To mark any module as replicable on their peers, select the module and click the **Enable Multi-Tenancy** checkbox on the `Modules` page (**Application Editor** > **Modules**):



When the **Enable Multi-Tenancy** checkbox is selected, the 'Tenant' field is added to the schema for the selected module, which can then be used to associate records in the system with specific tenants.

To stop replication, clear the **Enable Multi-Tenancy** checkbox. When you clear the checkbox, FortiSOAR displays a warning dialog informing you that if you confirm the action and stop replication, then all tenant-specific details and associations with the tenant, will be permanently removed:



If it is not your intention to remove all the tenant-specific details, then you should temporarily disable data replication for this module using the Data Replication page.

File replication is also enabled between the master and tenant nodes. As a result, records containing "file" type fields or records with correlations that map to modules containing "file" type fields are also replicated. For example, you can replicate 'Alert' records that contain 'Attachments' correlations.

In case a conflict is observed between the master copy and the tenant copy for record (s), the tenant copy of the data will be retained.

Various options available for replicated data, such as selective data sharing, synchronization of specific records using the FortiSOAR UI, etc., are available only when multi-tenancy is enabled, and replication is enabled for a module.

Starting with release 7.5.0, FortiSOAR includes support for pre- and post-processing rules for records being ingested into FortiSOAR. For more information, see the `Pre-Processing Rules` topic in the *Application Editor* chapter of the "Administration Guide".

Data Replication

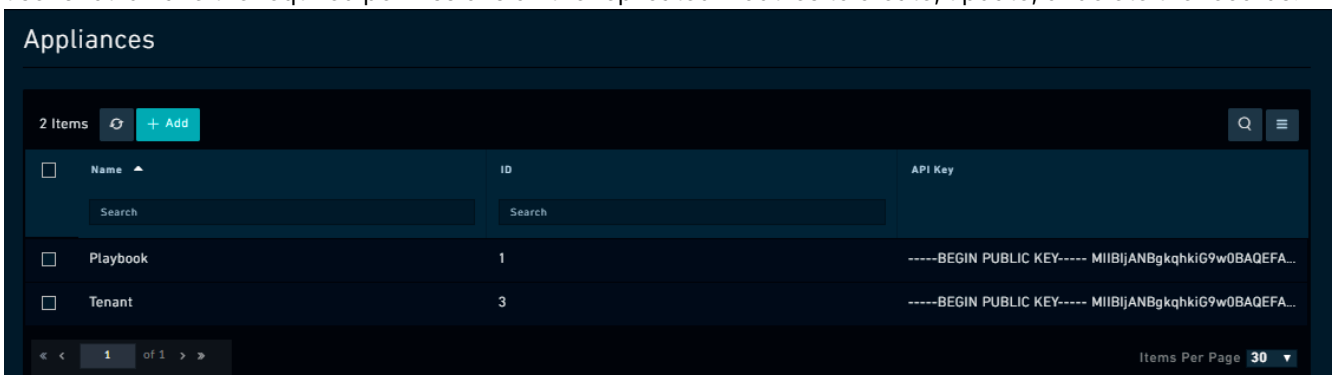
On the `Data Replication` page, you will see a list of modules that have been set up for replication, i.e., the **Enable Multi-Tenancy** checkbox for those modules is selected. You can also opt to temporarily disable data replication for a module on this page, as well as control the data that you want to share at the module and field levels. The ability to manage the data that is shared allows you to preserve data security and protect sensitive data.

On the Tenant node, by default, the `Alerts`, `Comments`, `Incidents`, and `Indicators` modules and all their fields are marked as replicable. On the Master node, by default, all modules are marked as replicable.

From release 7.4.2 onwards, support is added for selective data sharing from tenant to master and vice-versa. For more information, see the Selective Data Sharing topic.

Permissions Required

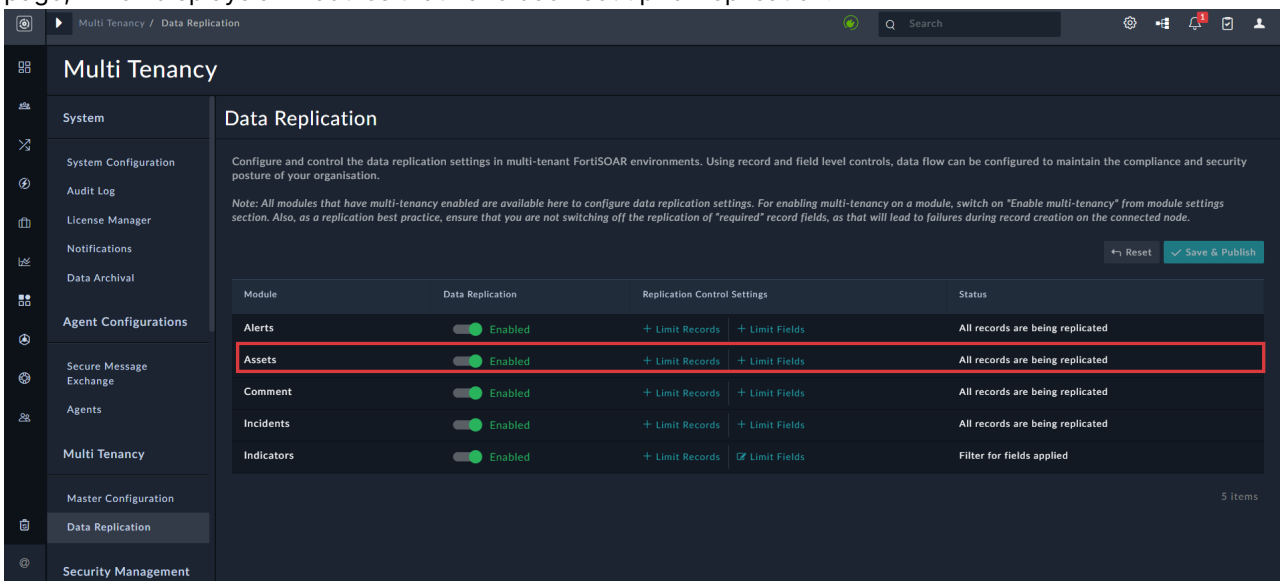
The FortiSOAR Routing Service hosted at each of the nodes uses an `Appliance` User to apply updates coming from the remote node. On a tenant node, this Appliance User is shown as the '`Master`' User, while on the master node, it is shown as the '`Tenant`' user. For all records to be updated from one node to the other, this appliance user should have the required permissions on the replicated modules to create, update, or delete the records.



Customizing Data Replication for modules

This process defines customizing data replication for a module on the 'Tenant' node. You can use the same method to customize replication for any module on the 'Master' node.

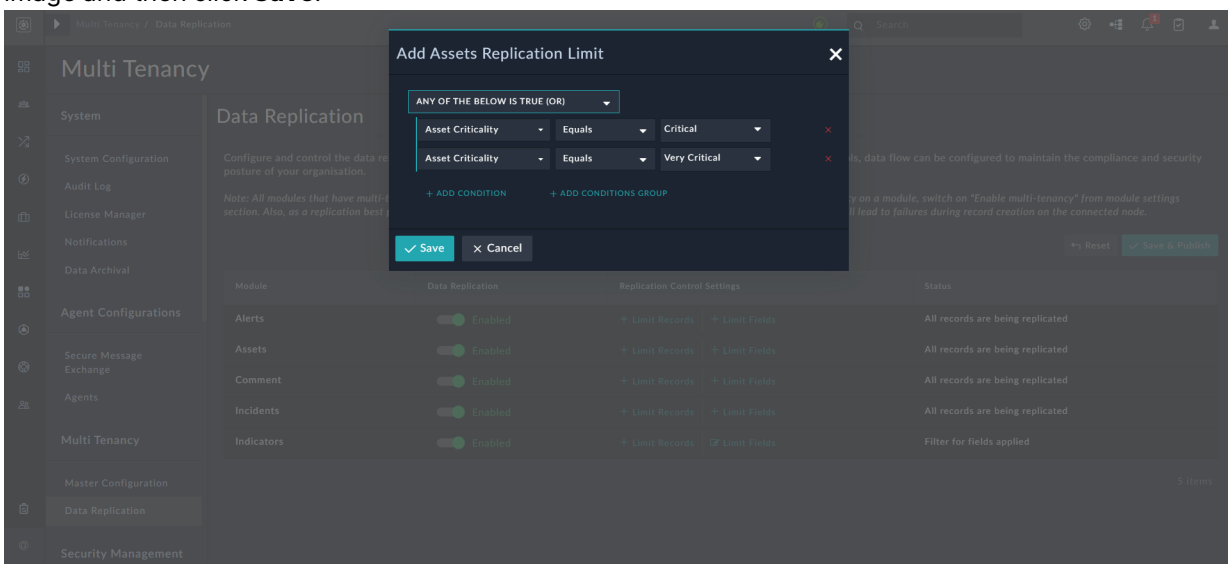
1. Log on to your FortiSOAR tenant node as an administrator and click the **Settings** icon to open the `System` page.
2. In the `Multi Tenancy` section, click **Data Replication** in the left menu to display the `Data Replication` page, which displays all modules that have been set up for replication:



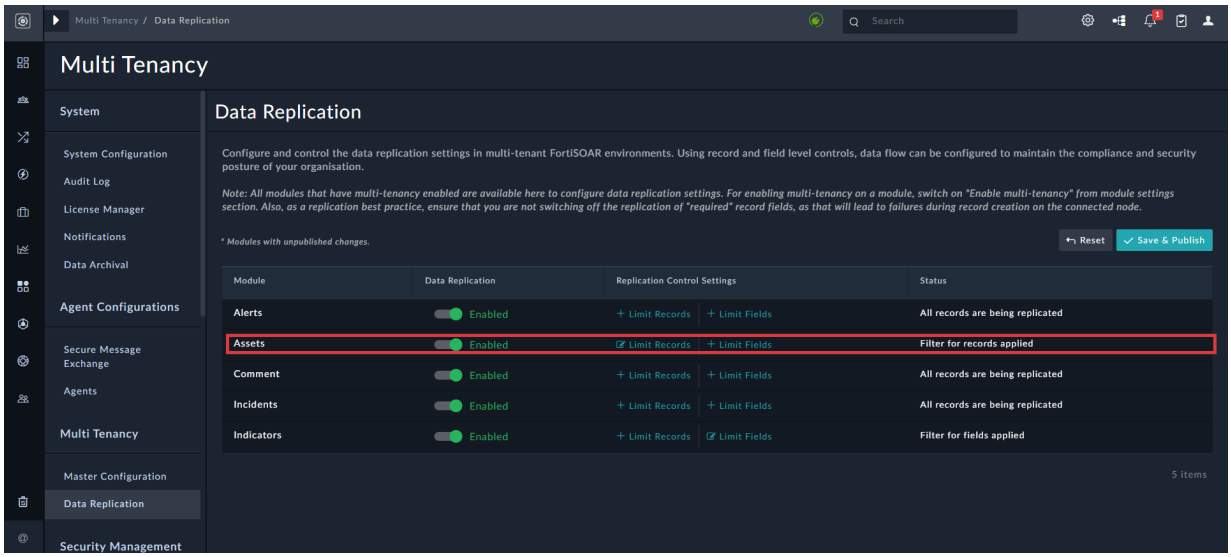
Note: For this example, we have set up the `Assets` module for replication. By default, on the Tenant node, the `Alerts`, `Comments`, `Incidents`, and `Indicators` modules, and all their fields (except the `Indicator` module) are marked as replicable.

Use the `Data Replication` page to configure the following:

a. To add conditional replication of records for a module, click the **+ Limit Records** link in the row of that module, for example, 'Assets'. In the `Add <Module Name> Replication Limit` pop-up, add the condition to be fulfilled for replication of the records. For example, if you want assets to be replicated only if their 'Asset Criticality Equals Critical or Very Critical', add the condition as shown in the following image and then click **Save**:

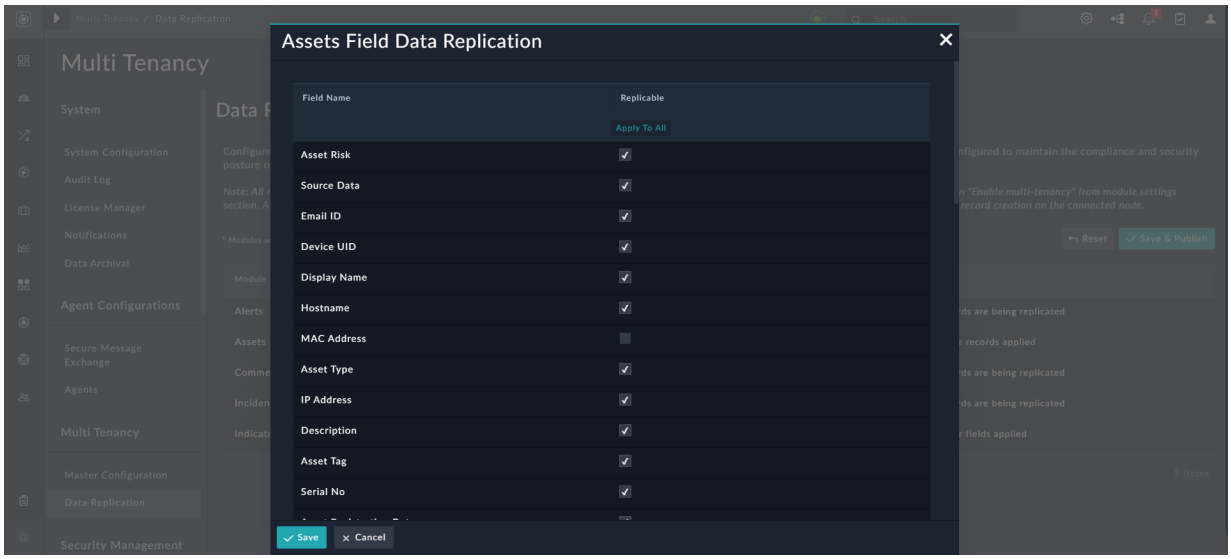


Click **Save & Publish** to save the changes and publish the module to reflect the changes in the system or click **Reset** to clear any changes made since the last saved event.

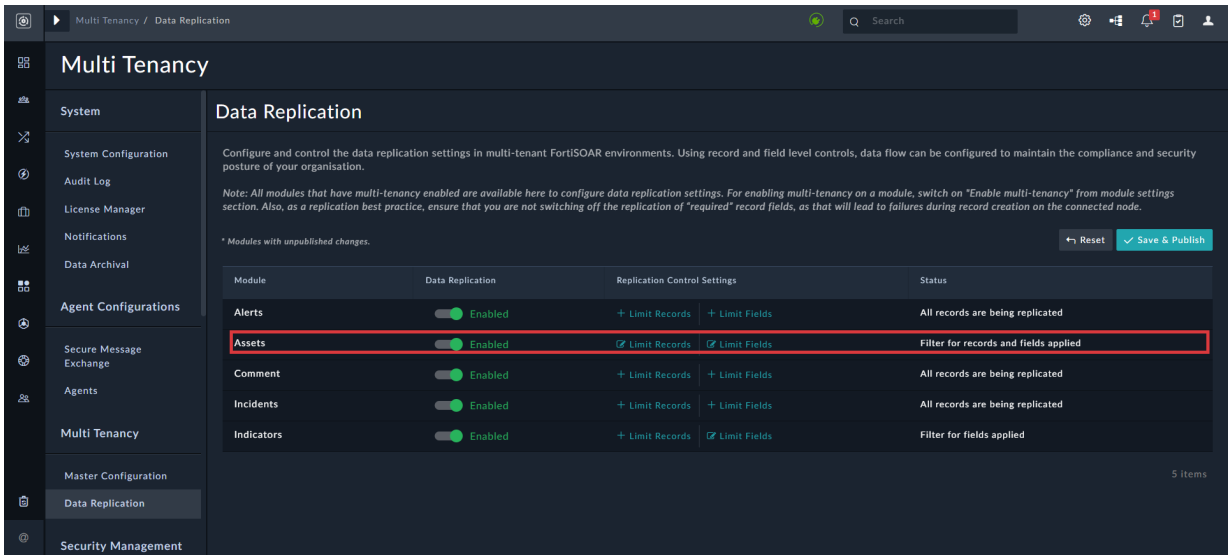


To make any further changes to the conditions for record replication, click the **Edit Limit Records** link and redefine the replication conditions.

b. To specify the fields that should be replicated between the tenant node and the master node, click the **+ Limit Fields** link to display the **<Module Name> Field Data Replication** dialog.
Note: By default, generally all fields in a module are set up as replicable. In the case of some modules, for example, `Indicators` module, a few fields are not replicated by default.
The `<Module Name> Field Data Replication` pop-up contains a list of fields that are part of the selected module. To remove data replication for a particular field, clear the checkbox that appears in the **Replicable** column in the field's row. To make all fields of a module replicable, click **Apply to All**.
Important: Ensure that you do not switch off the replication of the “required” record fields, as that will lead to the failure of record creation on the master node.



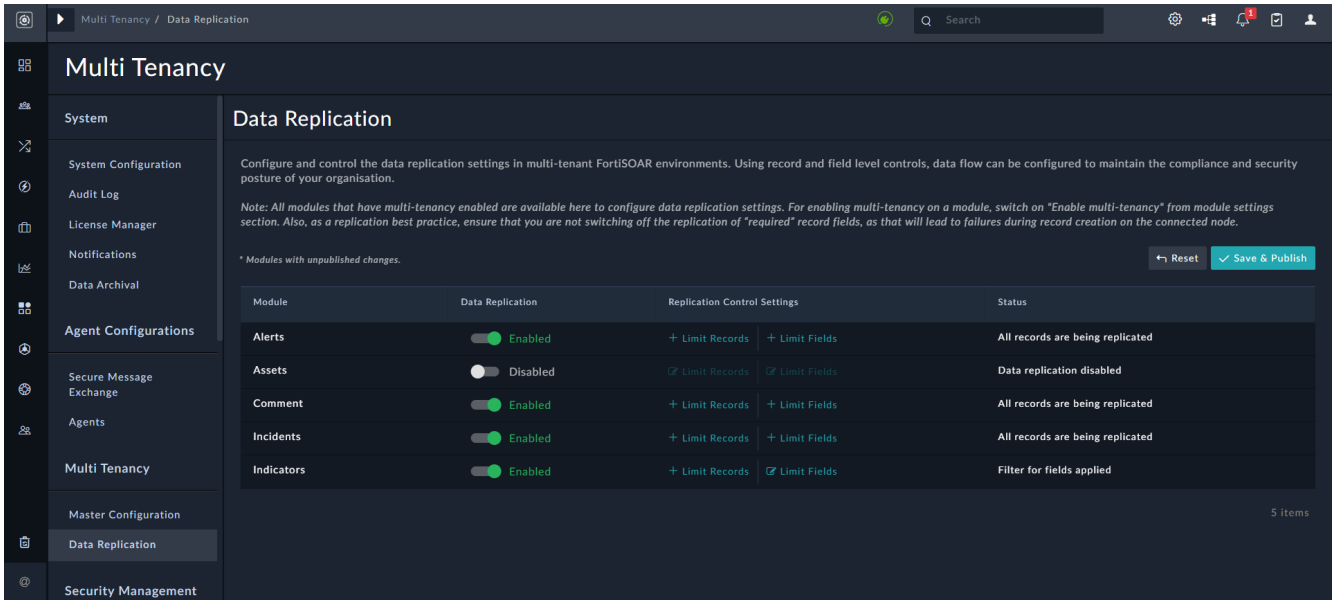
c. To save your changes, click **Save**.
Click **Save & Publish** to save the changes and publish the module to reflect the changes in the system or click **Reset** to clear any changes made since the last saved event.



To make any further changes to the conditions for field replication, click the **Edit Limit Fields** link and redefine the replication conditions.

Disabling Data Replication for a module

To temporarily disable data replication for a module, click the toggle switch in the **Data Replication** column and then click **Save & Publish**:



Default Ownership of records replicated from the tenant nodes to the master

Records that are replicated from a tenant node to a master node by default have the same ownership as the tenant. Therefore, to define which team should have visibility of records coming in from a particular tenant, assign the specific team as the owner of the particular tenant. In addition, the records will be visible to the agent appliance teams. You can assign owners of the records belonging to a tenant directly when you are adding tenants on the master node, see the Adding a tenant node on the master topic.

Considerations for Recycle Bin

The 'Recycle Bin' supports soft delete of workflow and module records; so that in the case of accidental deletion of playbook collections, playbooks or module records these records can be restored. For more information on the Recycle Bin, see the "Administration Guide."

In the case of MSSP environments, if any module is enabled for the Recycle Bin, then it is recommended that it should be enabled on both the master and tenant systems.

In the case of MSSP environments, if both the master and tenant systems have enabled recycle bin on a specific module, then any record that is moved to the recycle bin on the master node also gets moved to the recycle bin on the tenant nodes. However, if only the master or any tenant enables the recycle bin and not vice-versa, and if a record is deleted from the master (who has enabled the recycle bin) then that record gets permanently deleted from the tenants (who have not enabled the recycle bin. Similarly, if a record is restored on the master, then the record also gets replicated back on the tenant nodes; however, the record gets created as a new record on the tenant side.

Custom Module Considerations

When you create a custom module on a tenant node, you must consider the following points:

For the Tenant Node

- By default, replication is turned off (for both master and tenant) for any custom module you create on the tenant node. Therefore, if you want to replicate the fields of the tenant node to the master node, you can enable replication of that module by enabling multi-tenancy at the module level, by clicking the **Enable Multi-Tenancy** checkbox on the `Modules` page (**Application Editor** > **Modules**). Then, set up and data replication as required for that module from the Data Replication page.

For the Master node

- Ensure that the custom module that has been created on the tenant node is present on the master node, with data replication enabled for all the required fields. You can achieve this by exporting the custom module from the tenant node and importing the same into the master node. You must ensure that all the relationships of the custom module are kept intact during the export-import process.
- Ensure that the custom picklists are exported and imported correctly into the master node.

Selective Data Sharing

From release 7.4.2 onwards, support is added for selective data sharing from tenant to master and vice-versa. Now, you can configure conditional replication of records from tenant to master or from master to tenant in a distributed multi-tenancy environment. This is useful in cases where the tenant or the master wants to restrict the sharing of records. For example, tenants might want to share only high- or critical-level records such as alerts, or records of a certain type, with the master. Another example might be a case where all feeds are ingested into the Indicator module as part of a threat-feed ingestion. However, not all feeds need to be replicated to the peer node unless the indicator is sighted in the tenant environment. In this case, you can set the 'Indicator' module to replicate conditionally, only when the 'sighted' flag is 'true'. For more information, see the Customizing Data Replication for modules topic.

Synchronization of specific records from the FortiSOAR UI

Occasionally, due to some reasons, some records might not be replicated on peer systems, or the some of the replicated records might contain outdated information. To resolve this problem starting with release 7.4.2, you

can choose a record from the grid and synchronize it in real time.

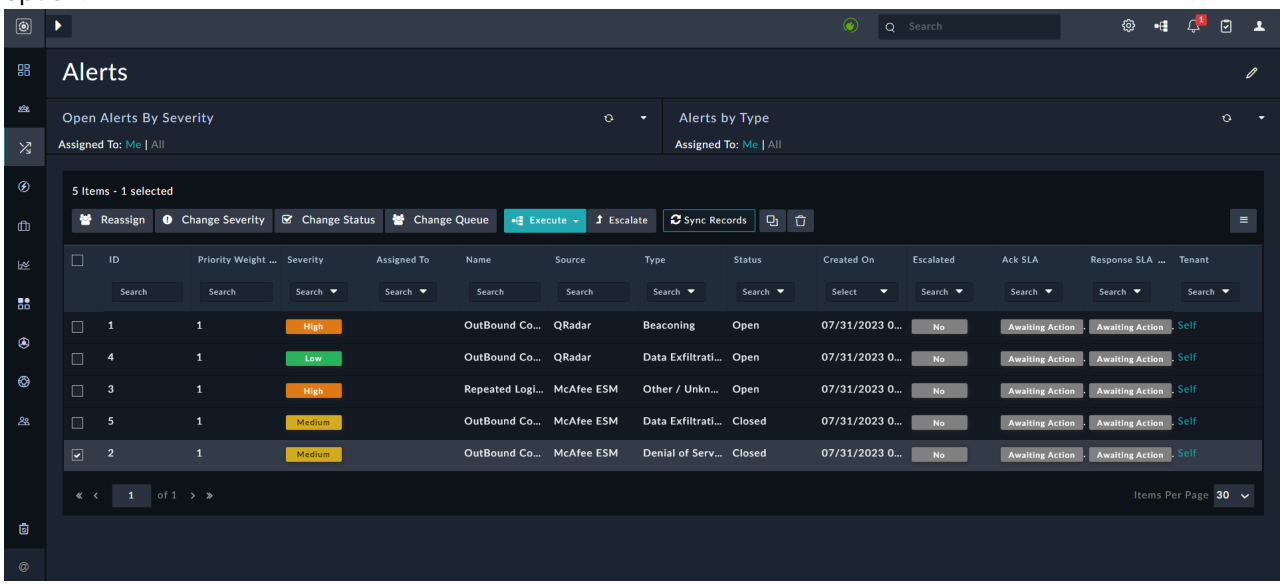
The following prerequisites must be satisfied before records can be synchronized:

- The system must be multi-tenanted, i.e., the license edition applied must be 'Multi-Tenant'.
- The module that contains records that you want to synchronize must be enabled for multi-tenancy, i.e., the **Enable Multi-Tenancy** checkbox on the `Modules` page (Application Editor > Modules) must be selected.
- The module must be enabled for replication on the `Data Replication` page.
- To synchronize records, users must have a minimum of `Update` access on the module containing the record, along with other requisite permissions.

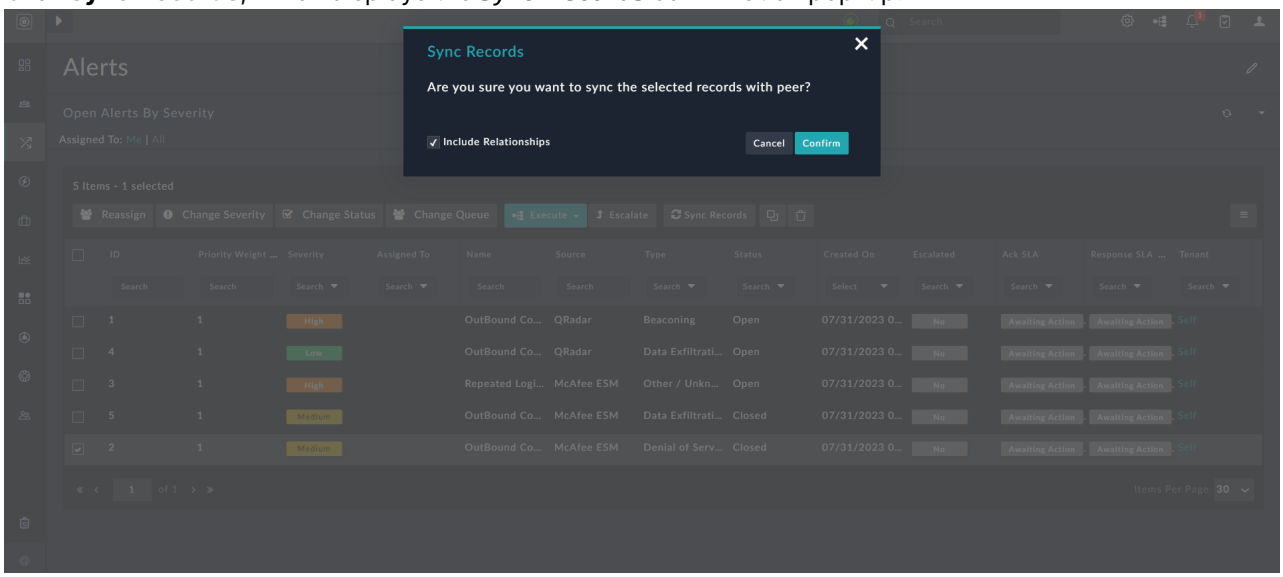
If your master node is on release 7.4.2 and your tenant nodes are on releases earlier than 7.4.2, such as 7.4.1 or 7.4.0, it is recommended that you synchronize records along with their relationships.

If all the criteria are met, then users can synchronize records as follows:

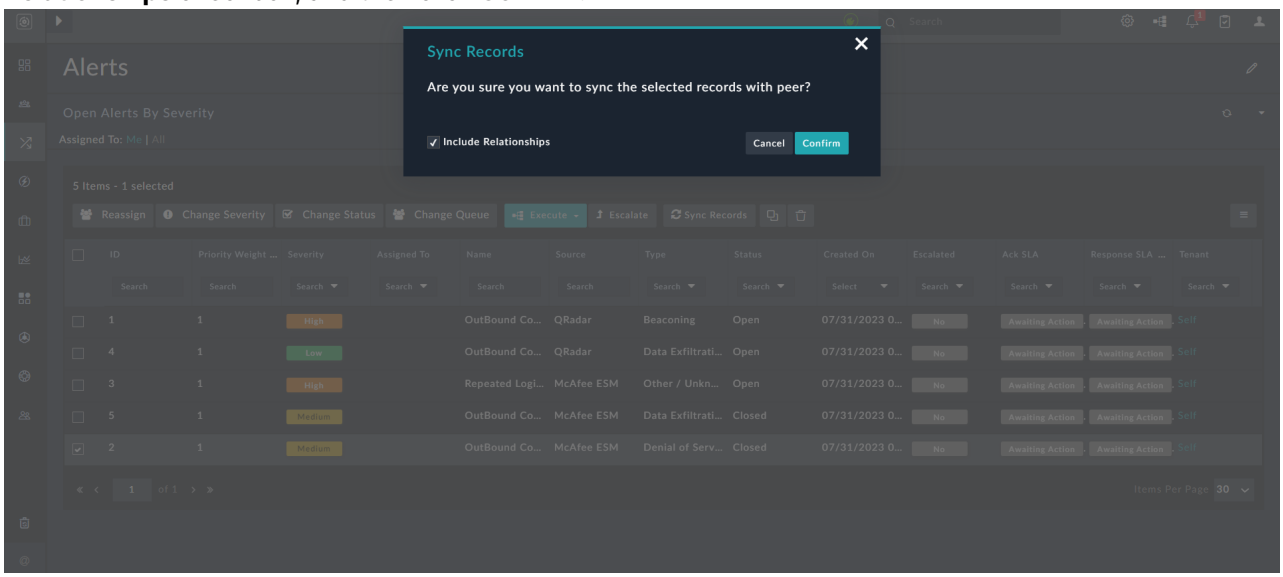
1. Navigate to the module containing the record that needs to be synchronized, for example, the 'Alerts' module.
2. On that module's grid, select the record or records to be synchronized, which displays the **Sync Records** option:



3. Click **Sync Records**, which displays the Sync Records confirmation pop-up:

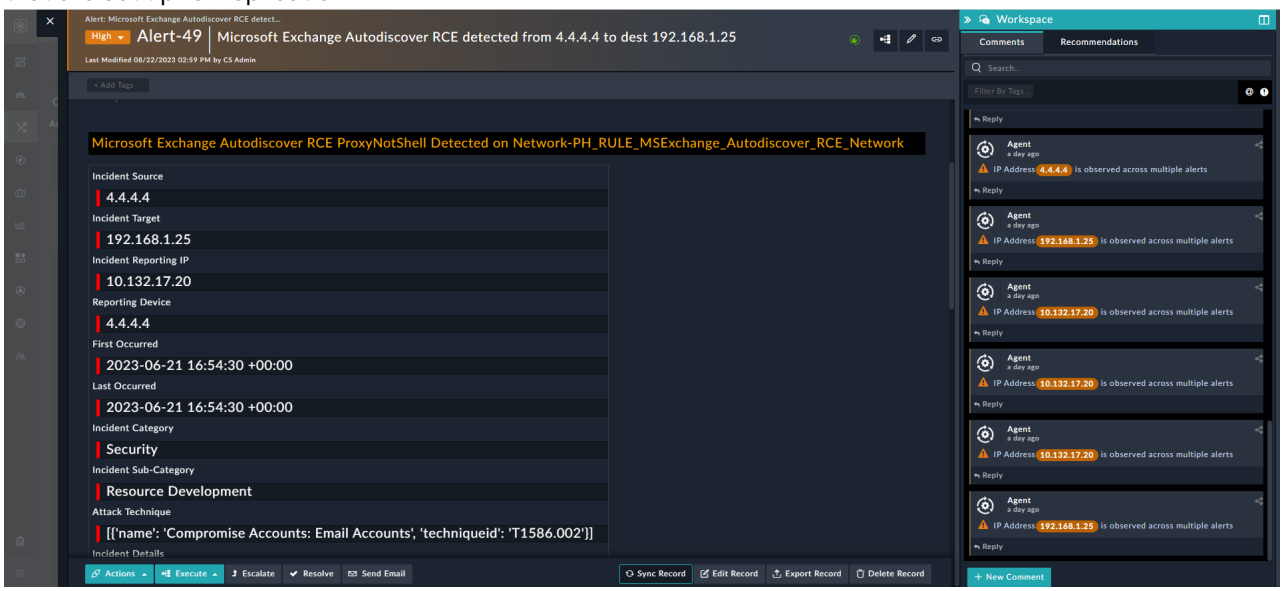


4. On the Sync Records pop-up, click **Confirm** to synchronize the record.
NOTE: If you want all the related records to be linked to the record on the replicated side, select the **Include Relationships** checkbox, and then click **Confirm**.



NOTE: If your master node is on release 7.4.2 and your tenant nodes are on releases earlier than 7.4.2, such as 7.4.1 or 7.4.0, it is recommended to select the **Include Relationships** checkbox, so that records including their relationships are synchronized.
The 'Sync Record' operation replicates the record(s) to the peer systems or updates the record in the peer systems as per the information contained in the record(s) on which synchronization was run.
NOTE: Only when there is connectivity between the master and tenant nodes will the records be synchronized.
Similarly, you can also sync records using the **Sync Records** button in the record's detail view of modules

that are set up for replication:



Record replication will not fail for missing relationship fields

From release 7.4.2 onwards, record replication will not fail for a related record that might not be available for replication, i.e., missing relationships do not cause record replication failures. This improvement has been made to prevent replication failure related data loss and SLA violations. Record replication might fail for a number of reasons, including:

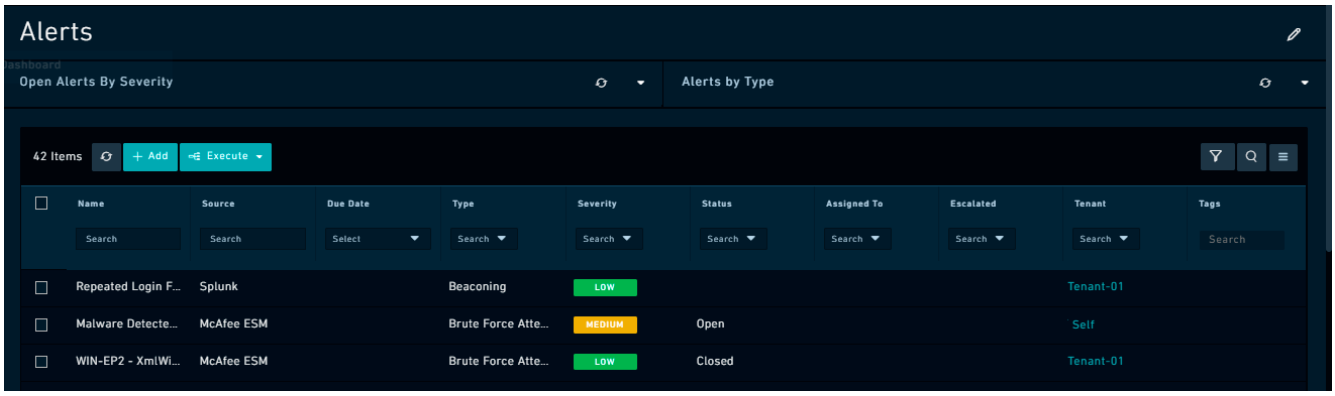
- Addition of a new or custom module for replication, and if this module contains existing records, then these records by default would not be synchronized on peer systems.
- Mismatched MMDs or picklist UUIDs can lead to the unavailability of some records on peer systems.
- Failure to add known IP addresses to the 'allowlist'. Now, every new record created will fail to replicate on peer systems due to the missing relation, i.e., the failure to link the IP address, if one of the known IP addresses is added as an Indicator to newly-made records and, for some reason, this IP address is not replicated on peer systems.

A single record can frequently result in replication failures for numerous correlated records, resulting in SLA violations and data loss. This improvement ensures that records will be replicated on peer systems, even if replication for related records fails.

Working with replicated records on the Master node

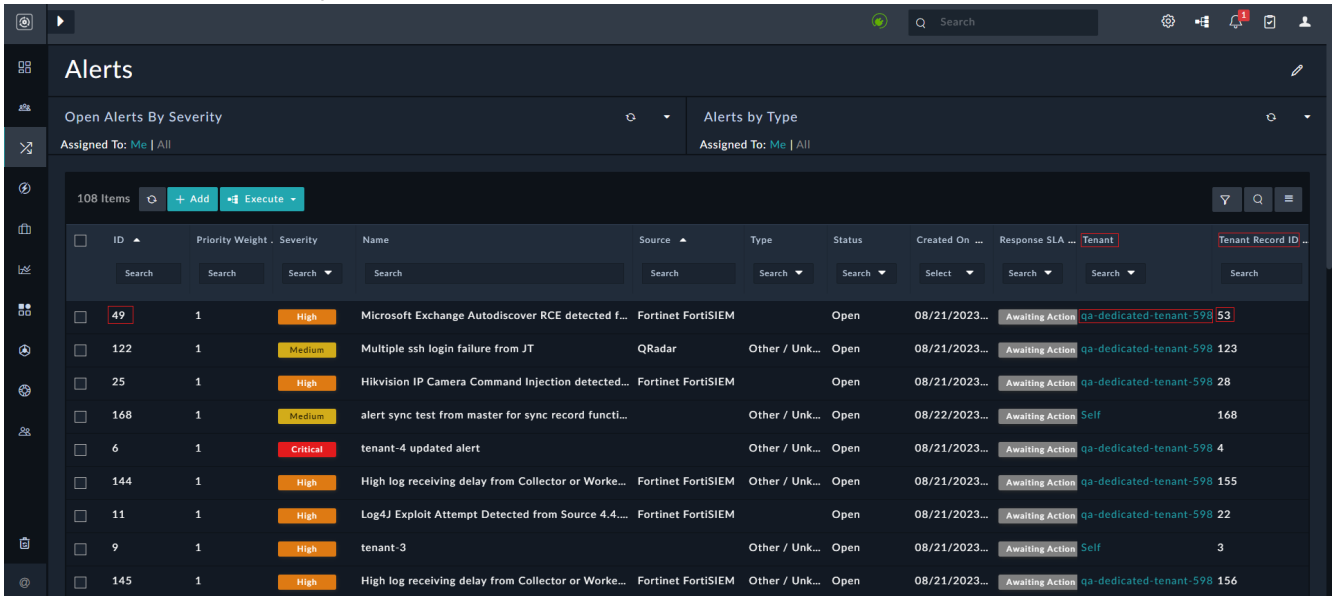
Ensure that the modules that require to be replicated and synchronized bidirectionally are enabled for replication on the master node. By default, all modules are marked as replicable on the master node. If you want to change this configuration follow the same steps on the master node, as mentioned in the Customizing Data Replication for modules topic.

On the master node, users can identify records using the *UUID* of the record, which is common for both master and tenants' records. You can get the UUID of the record by clicking the record, and the UUID will be shown on the URL bar in the format: `/view-panel/<moduleName>/UUID?`. Also, on the master node, in the **Grid** or **Listing** view, each module that is replicable will have **Tenant** as a column, by default, as shown in the following image:



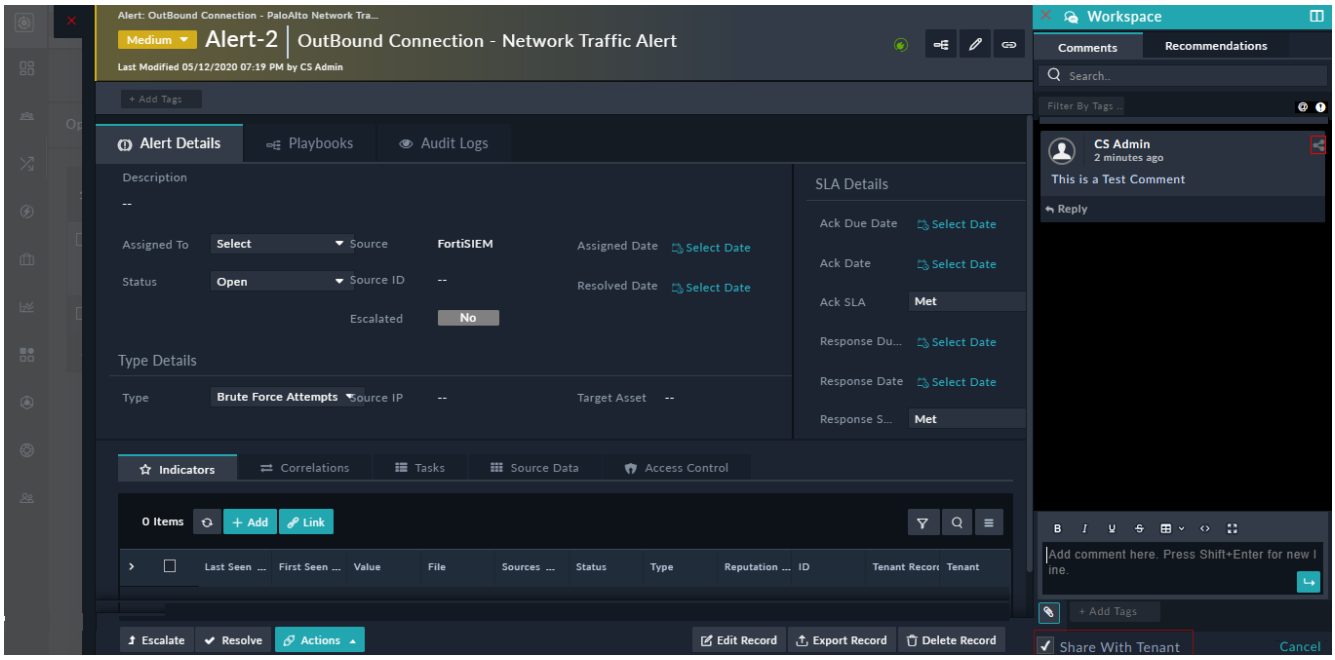
If no tenant is assigned while adding the alert, i.e., if the alert is created on the master, then that record is created as a "`Self`" record, i.e., **Self** will appear in the Tenant column. Note that if the value of the tenant associated with a record is set to "Self", then you are allowed a one-time edit of the tenant field.

When a record is replicated from a tenant node onto the master node, in the module's grid view for replicated records, you can see the name of the tenant as well as ID of the record on the tenant node in the **Tenant Record ID** field. When a record is replicated from a tenant node to the master node, its ID differs on both the nodes. Therefore, knowing the both the name of the tenant and the ID of the record on the tenant node makes it easier to identify the replicated record. For example, in the following image, the ID of the alert on the master node is '49', the tenant's name is 'qa-dedicated-tenant-598', and the ID of the record on the tenant node is '53':



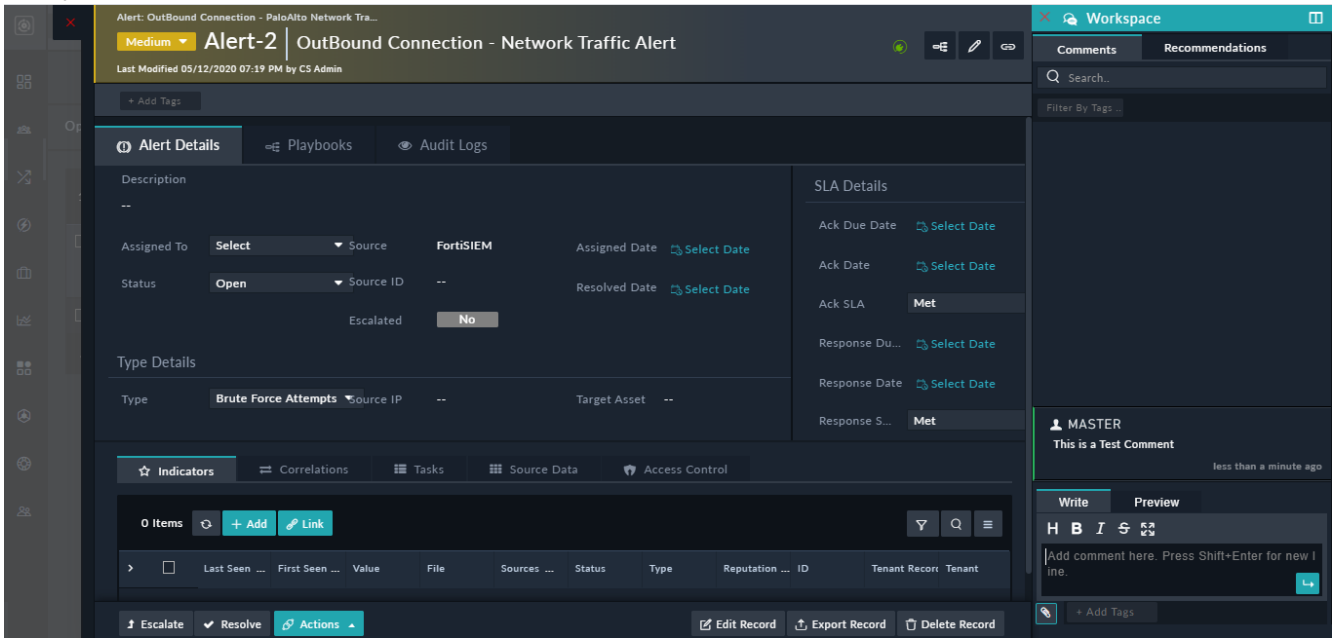
Posting of comments from the master node to the tenant node

By default, comments are pushed from the tenant node to the master node. Similarly, adding a comment to a tenant record on the master node pushes it to the tenant node. This is because the **Share With Tenant** checkbox is enabled by default in the collaboration section of the master node's record. Any comment that is pushed to the tenant appears with a shared icon on the master node:



Having the **Share With Tenant** checkbox selected by default ensures that the added comments are added to the record at the tenant node. Also, note that the 'Tenant' field in the record is set to the tenant of the parent record. The 'Tenant' field had a value of 'Self' in releases prior to 7.4.2. If you want to restrict the sharing of comments with peers, you must explicitly clear the **Share With Tenant** checkbox.

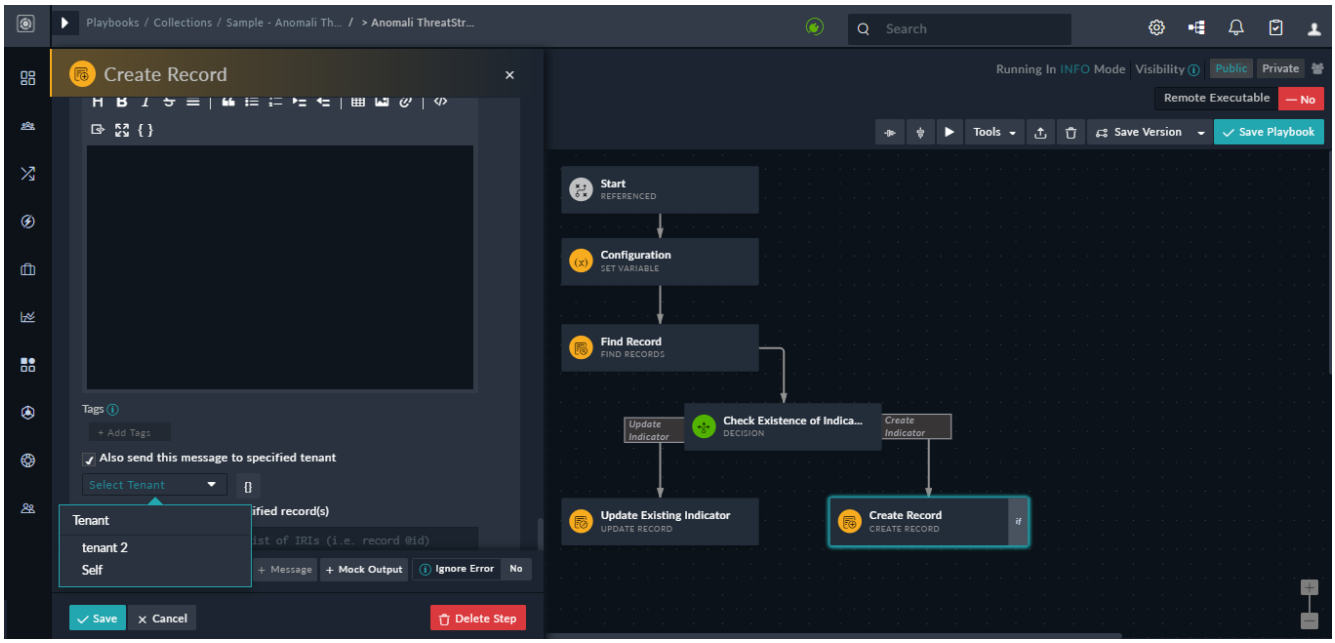
The pushed comment is labeled "Master" on the tenant node:



You can add attachments (files) to comments, and those comments, along with the related files, get replicated between the respective master and tenant nodes.

Users with appropriate permissions can edit or delete comments, and the same will be reflected on the respective master or tenant nodes, i.e., if a comment is edited on a master node, the same will be reflected on the tenant node. For more information on editing comments, see the *Working with Modules - Alerts & Incidents* chapter in the "User Guide."

In order for a user on the tenant node to follow the investigation being conducted on the record, you can choose to replicate the comments that are linked to the record on the tenant node if your playbook contains steps that add "Messages" to the record that initiates the playbook on the master node. To replicate comments on the tenant node, in the playbook, select the **Also send this message to specified tenant** checkbox, and from the **Select Tenant** drop-down list, select the tenant node on which you want to replicate the comments, or click the **Add Custom Expression {}** icon to specify tenant IRIs in this field.

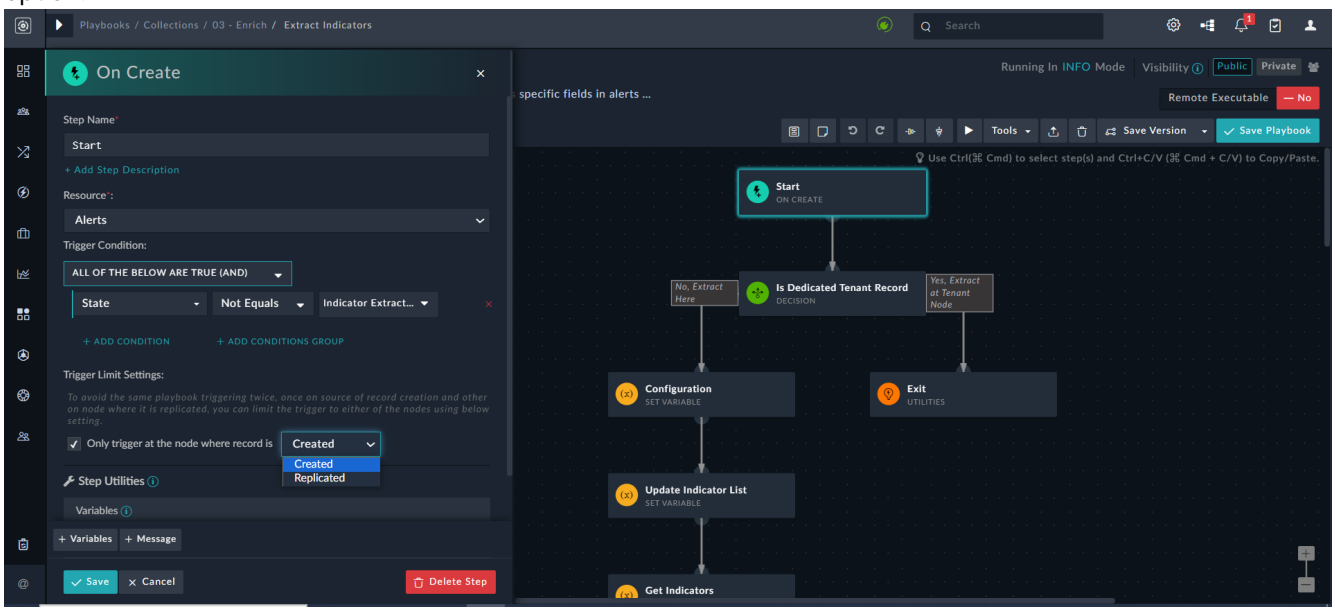


Ability to execute the "On Create," "On Update," or "On Delete" playbooks on either the source or a replicated instance of the record

Prior to version 7.4.2, for modules that had multi-tenancy enabled and configured for data replication, the On Create, On Update, and On Delete playbooks executed on both the instances where the record is created as well as on the instance where the record is replicated.

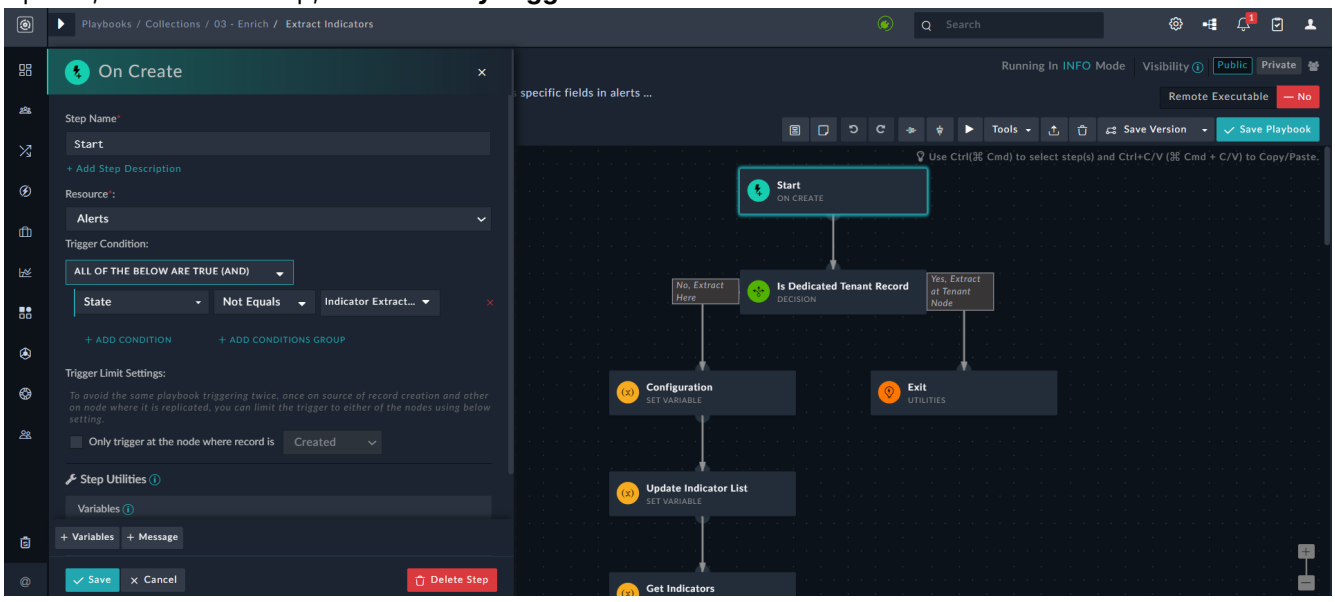
From release 7.4.2 onwards, you can choose to run these playbooks either on the instance where the record is created or where it is replicated. For example, once records are manually created or created using playbooks or ingestion on the tenant node, they get replicated on the master node. By default, 'On Create' playbooks are launched concurrently to retrieve IOCs related to the record on the tenant node. In parallel, the 'Extract IOC' playbook is run on the replicated record on the master node, which causes indicators to be extracted on both the master and tenant. More crucially, replication also fails, since the related indicators now have different UUIDs on master and tenant. Therefore, if you want the On Create playbook to be run only on the tenant node where the record is created, open the playbook that contains the 'On Create' step, and in the `Trigger Limit Settings` section, select the '**Only trigger at the node where record is**' checkbox, and then select the **Created**

option:



In a similar way, you can update the 'On Update' or 'On Delete' playbooks to run on either the instance where the record is created, where it is replicated, or both.

From release 7.4.2 onwards, the default behavior of the On Create, On Update, and On Delete playbooks is to run only on the instance where the record is created. If you want to change the default behavior of a playbook, so that it runs on both the source and replicated instances, open the playbook that contains the On Create, On Update, or On Delete step, clear the **Only trigger at the node where record is** checkbox:



Similar to this, to run the On Create, On Update, and On Delete playbooks only on the instance where the record is replicated, open the playbook that contains the On Create, On Update, or On Delete step, select the **Only trigger at the node where record is** checkbox and then select the **Replicated** option.

You can choose to retain the previous behavior for these playbooks, i.e., run the On Create, On Update, and On Delete playbooks on both the source and the replicated instance of the record, steps of which are mentioned in the 'Post-Upgrade Tasks' chapter of the *Upgrade Guide*.

Also, note that when a record that is not present on the master node is updated on tenant node, then a new record gets created on the master node. Due to this enhancement, when the record is updated on the tenant node an "On Update" playbook gets triggered on the tenant node and at the same time, record creation on the master node triggers the "On Create" playbook.

Troubleshooting Tips for Data Replication

Resolving of conflict of data between the master and tenant nodes

Post creation of a record, only the changed or updated attributes of the record are synced bi-directionally. However, when a record is updated simultaneously at both the nodes, it could lead to conflicts and override the updates at the respective nodes. If there is such a conflict, then the tenant copy of the data will be retained.

To address such scenarios, when an update to a record, which is updated at the master node, arrives at the corresponding tenant node, the routing service compares the modified time of the record at the remote node against the last modified time of the record at the local node. If the local modified time is found to be greater, the update from the master node is rejected, and the record is marked for conflict. Subsequently, a complete sync of the latest copy of the record at the tenant node is automatically sent to the master node.

Since the modified time of the records at the distributed nodes are compared, it is important that the time at the nodes are in sync with a common NTP server. For example, time.apple.com.

Handling of network outages or loss of connectivity between a node and the secure message exchange

When a node loses connectivity to the secure message exchange, all outbound messages are stored locally by the routing service hosted on the node. This replicator log is automatically replayed once the connectivity is restored. All inbound messages from other nodes to the said node are persisted at the secure message exchange. These messages get automatically delivered once the connectivity is restored.
For more information, see the Troubleshooting chapter.

Handling a mismatch in the module metadata of replicated modules

When a module, say the `Alert` module, is configured for replication from the tenant node to the master node, the module replication would work seamlessly, if the metadata at both the nodes is identical. However, if there is an MMD (module metadata) difference or in case of custom picklists, the following must be ensured:

- If there is a difference in the module metadata at the two nodes, or if some fields are excluded from replication, these must be fields that are "not-required" for record creation. Else, record replication to the target node will fail due to missing required fields.
- For custom picklists added to a module being replicated, the picklist item IDs should be the same at both the nodes. You can ensure this by exporting picklists from one node and importing them to the other node using the `Configuration Manager`. For more information, see the *Application Editor* chapter in the

"Administration Guide."
Creating the picklists manually at each node would result in a difference in the IDs, leading to record replication failure. The record replication fails as the record replication works on record IDs, and the mismatch in picklist item IDs leads to picklists not being replicated.

Whenever a record replication fails to get applied at the target node due to the reasons mentioned above, an entry will be created in the Audit Logs with the type `Replication Failure`. These audit entries can also be seen on the respective records if any update from a remote node fails to be applied due to any metadata changes done later.

Managing connectors of distributed tenants

Prior to version 6.4.4., all the connector installations or configurations done at the distributed tenant node replicated back to master, by default. To disallow the master from remotely executing connector actions on the tenant, the `ENABLE_REMOTE_CONNECTOR_OPERATION` parameter required to be set to `false` in the agent's (associated with the tenant) rpm `config.ini` file (`/opt/cyops-integrations/integrations/configs/config.ini`). The `ENABLE_REMOTE_CONNECTOR_OPERATION` parameter is set to `true` by default. However, in this case, the master node did not get notified that its tenant had turned off this remote operation setting. Due to which if the master triggers any remote request then that would get ignored by the tenant since the remote operation had been disallowed. Therefore, from version 6.4.4 onwards, you can allow or disallow the master from managing and executing connector actions, including actions executed using remotely executable playbooks ("Trigger Tenant Playbook") on the tenant node, using the FortiSOAR UI. Remote connector management is achieved using the **Allow Connector Management** button on the on the `Master Configuration` page of the tenant node. To remove the master's ability and permissions to remotely manage and execute connectors on the tenant node, toggle the **Allow Connector Management** button to **NO**.

To disallow the master from remotely executing connector actions on an agent, ensure that the agent's version must be 6.4.4 and later

Once you set the **Allow Connector Management** button to **NO**, the master node is notified and the master node can no longer send remote connector requests to such tenants.

If you have upgraded a tenant node to version 6.4.4 and later from a version prior to 6.4.4, in which you had changed the value of the `ENABLE_REMOTE_CONNECTOR_OPERATION` parameter in the `config.ini` file from `true` to `false`, then the value (`false`) will not be honored, i.e., after the upgrade the master will yet be able to remotely manage connectors on the tenant node. This is because in the case of tenant nodes, the value of the `ENABLE_REMOTE_CONNECTOR_OPERATION` parameter in the `config.ini` file has no effect from version 6.4.4 onwards, as post version 6.4.4 , remote connector management is handled by the **Allow Connector Management** button.

Managing Tenants

The "Tenant Manager" helps the master remotely manage playbooks by pushing and mapping playbooks and tenants' data, including picklist and modules. The master node can make changes to the tenants' model metadata (MMD) and push those changes to the tenant node.

Release 7.3.1 adds support for invoking playbooks by adding aliases even in a shared tenancy model, enabling you to construct agnostic playbooks. In the previous releases, you were required to add the 'Reference A Playbook' step for shared tenants and the 'Reference Remote Playbook' (renamed to 'Trigger Tenant Playbook') step for dedicated tenants, which was inefficient. Now, you can map both shared and dedicated tenants in the 'Trigger Tenant Playbook' step. For more information about managing playbook mappings and allowing remote execution of playbooks, see the Managing Playbook Mappings and Executing remote playbooks at the tenant node from the master node topics.

Pushing playbooks from the master node for remote execution

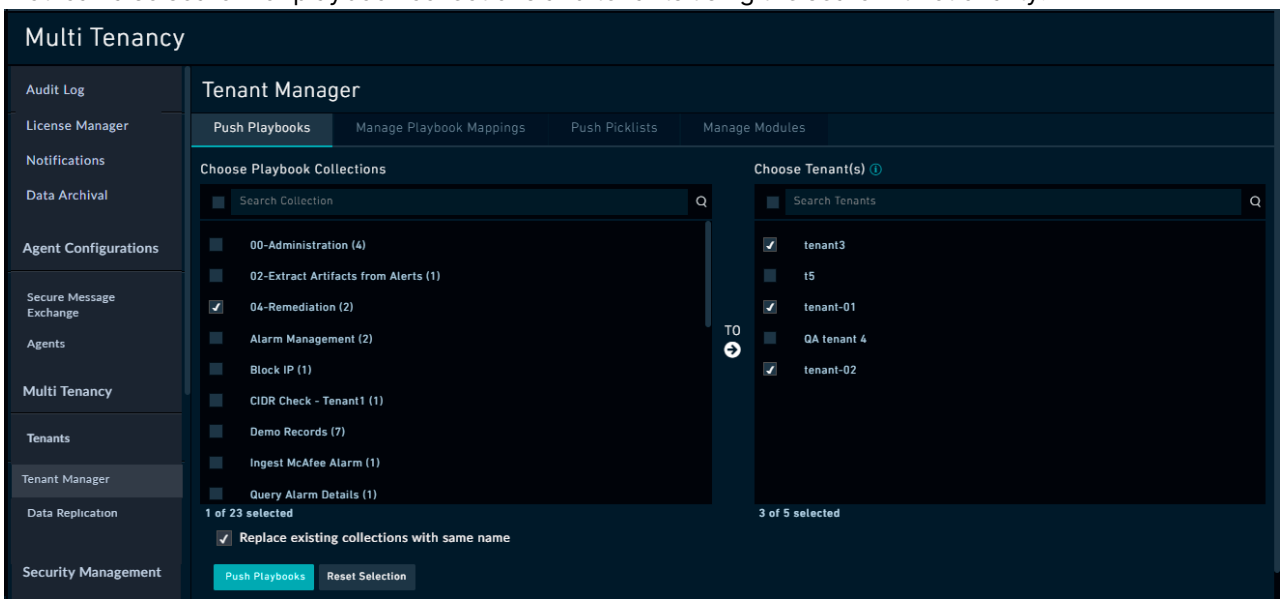
You can create a playbook on the master node and then push it to dedicated tenant nodes, enabling you to create and maintain playbooks on the master node and run them as required on the dedicated tenant nodes.

When a playbook is pushed from the master node to a dedicated tenant node, and if you have assigned a playbook step, for example, a manual input step to a specific user or team, then the ownership of that playbook step gets reset.

To push playbook collections to the dedicated tenant nodes, do the following:

1. Log on to your FortiSOAR master node as an administrator and click the **Settings** icon to open the `System` page.
2. In the `Multi Tenancy` section, click **Tenant Manager** in the left menu.
3. Click the **Push Playbooks** tab.
   The `Push Playbooks` page displays a list of playbook collections and their mappings to various tenants. For example, you might have a collection of Remediation playbooks that you want to push to any or all of your tenants.
4. On the `Push Playbooks` page, from **Choose Playbook Collections**, select the playbook collection(s) that you want to push to the tenant nodes from the master node.
5. From **Choose Tenant(s)**, select the tenant(s) to whom you want to push the playbooks and click the **TO** arrow.
   The **Choose Tenant(s)** list lists only those tenants that are active and whose connections are verified.

You can also search for playbook collections and tenants using the search functionality.



6. By default, the **Replace existing collections with the same name** checkbox is selected, which signifies that if there is an existing playbook collection with the same name (ID) on the tenant node, then that playbook collection will get replaced on the tenant node.
   In case you want to keep the existing playbook collection on the tenant node, then you must clear the **Replace existing collections with same name** checkbox. In this case, the existing playbook collection is not replaced on the tenant node.
7. To push the mapped playbooks to the tenant nodes, click **Push Playbooks**.
   The pushed playbook collections are maintained on the master node, and they are cloned and mapped on the tenant node, i.e., the Push action creates a copy of the playbook on the tenant node. Therefore, the playbook on the master and the copy of the playbook on the tenant is maintained separately on the respective nodes.
   If any changes are required to the playbooks, then the playbook(s) can be updated in the following ways:
   a. Changes can be made on the master node and then pushed again to the tenant node. In this case, ensure that you select the **Replace existing collections with the same name** checkbox.
   b. Changes can be made on the tenant node.
   Note: Any changes made to playbooks on the tenant node do not replace playbooks on the master node.

Managing Playbook Mappings

The `Manage Playbook Mappings` page displays a list of playbook aliases and their mappings to various dedicated and shared tenants. A playbook alias makes it easier to reference tenant playbooks in the **Trigger Tenant Playbook** step in the playbook designer.
For example, you might have a collection of "Remediation" playbooks, such as playbooks to block an IP address and a playbook to check the reputation of an IP address using a 3rd party threat intelligence tool such as Anomali ThreatStream. Or you might also have a number of "Investigate BFA" playbooks such as Investigate BFA using Splunk, Investigate BFA using QRadar, or Investigate BFA using LogRhythm, which could map to different tenants having or using different SIEM products.

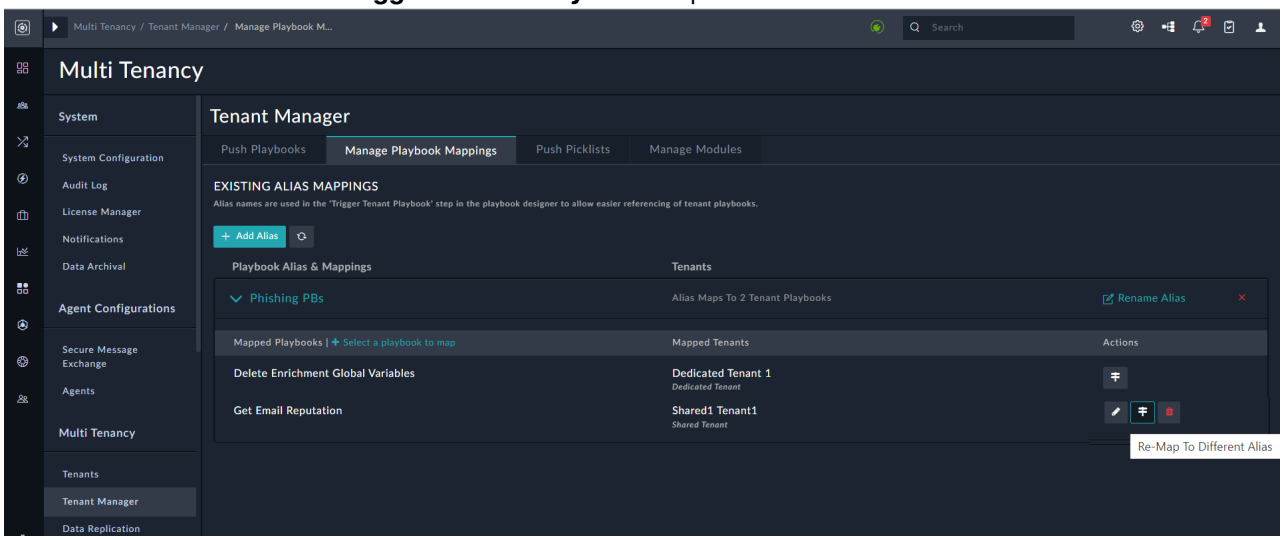
You can manage your playbook mappings by adding alias names, which facilitates the referencing of a tenant's (**both shared and dedicated**) playbook. You can also rename an existing alias or re-map an existing alias to a different alias.

To create an alias, you must ensure that the **Remote Executable** setting for the playbook is enabled in the playbook designer. For more information, see the Executing remote playbooks at the tenant node from the master node section.

In the case of a dedicated tenant, `Manage Playbook Mappings` lists the playbooks that have the playbook alias mapped to the dedicated tenant. In the case of dedicated tenants, you can only remap an existing alias to a different alias.

In the case of shared tenants, you can create the mapping between the playbook and the shared tenants, i.e., you can choose both the playbook and select a shared tenant to create the mapping. In the case of dedicated tenants, you can edit or delete the mapping between the playbook and the shared tenants and remap an existing alias to a different alias.

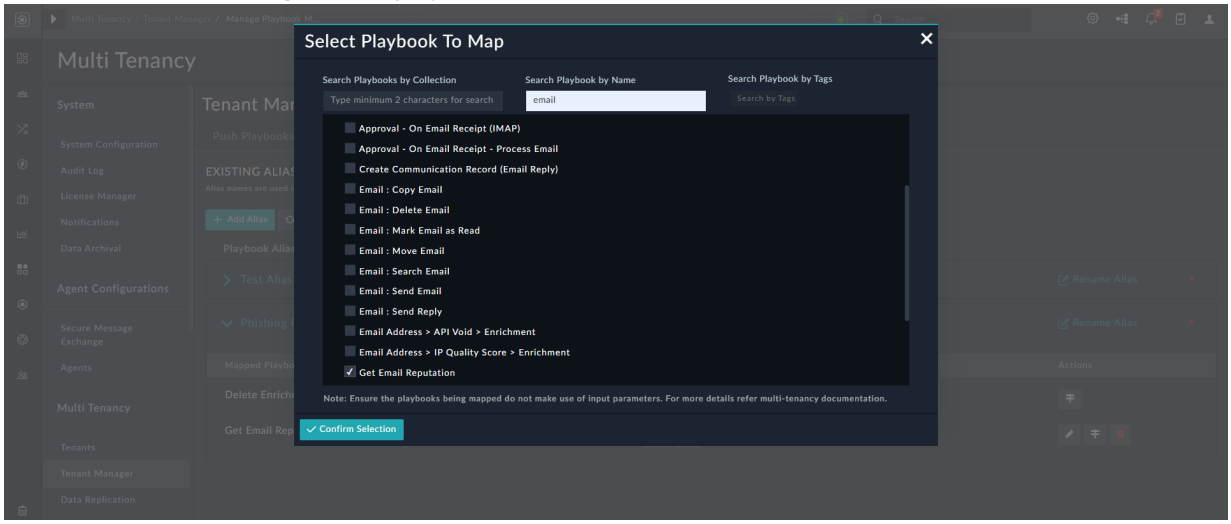
1. Log on to your FortiSOAR master node as an administrator and click the **Settings** icon to open the `System` page.
2. In the `Multi Tenancy` section, click **Tenant Manager** in the left menu.
3. Click the **Manage Playbook Mappings** tab.
4. To add a new alias name, click **Add Alias** to display the `New Alias Name` dialog. Enter the name of the alias in the dialog, and click **Create**.
   Alias names are used in the **Trigger Tenant Playbook** step.



   In the case of a dedicated tenant, the `Manage Playbook Mappings` page lists the playbooks that have the playbook alias mapped to the dedicated tenant. In the case of dedicated tenants, you can only remap an existing alias to a different alias or rename an alias.
   In the case of shared tenants, you can use the `Manage Playbook Mappings` page to create a mapping between the playbook and the shared tenants, i.e., you can choose both the playbook and select a shared tenant to create the mapping. Additionally, you can also edit or delete the mapping between the playbook and the shared tenants, remap an existing alias to a different alias, or rename an alias.
5. Perform any or all of the following actions on the `Manage Playbook Mappings` page:
   a. **Rename the alias**: Supported for both shared and dedicated tenants. Click the **Rename Alias** link on the Alias Name row, for example, `Phishing PBs`, to display the **Rename Alias** dialog. Update the name of the alias in the dialog, and click **Save**.

b. **Re-map an existing alias to a different alias**: Supported for both shared and dedicated tenants. If you want to remap an existing alias to a different alias, expand the Alias Name row and click the **Re-map To Different Alias** icon in the row of the mapped playbook that you want to remap to display the `Remap Playbook` dialog. In this dialog, from the **Select A Different Alias To Remap** drop-down list, select the alias to which you want to remap the playbook:

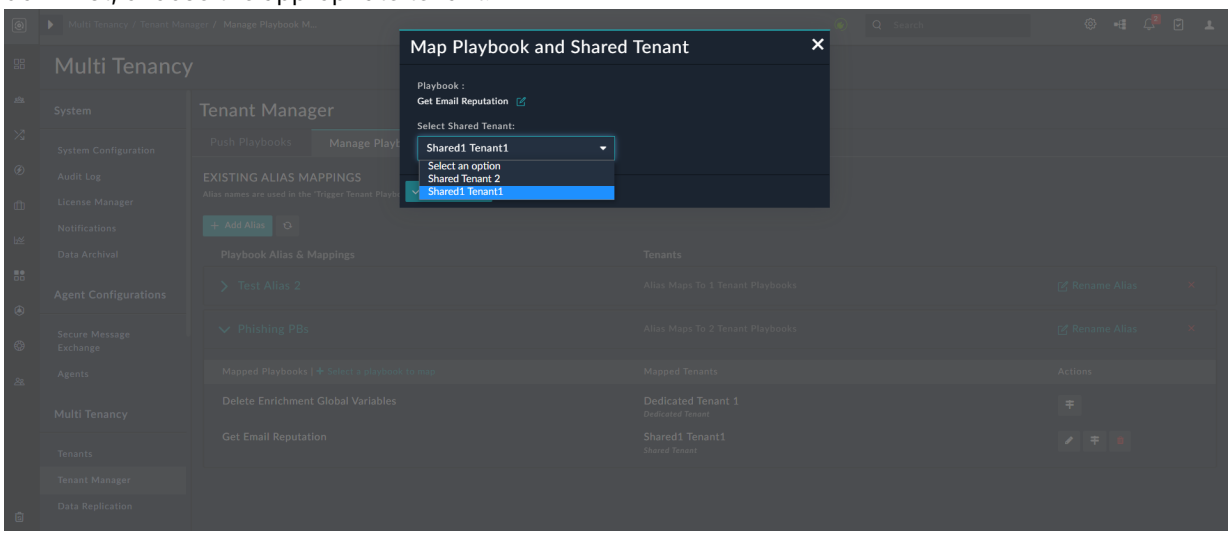
c. **Mapping playbooks to shared tenants**: Supported for only shared tenants. Click the **Select a Playbook to map** link to set the mapping between a playbook and a shared tenant, or click the **Edit Mapping** icon in the row of the mapped playbook to update the mapping between a playbook and a shared tenant. Both of these cases display the `Map Playbook and Shared Tenant` dialog. In this dialog, click the **Choose Playbook** link (to set the mapping) or click the **Edit** icon in the **Playbook** field (to update the mapping). Both of these cases display the `Select Playbook To Map` dialog. In this dialog, use the **Search Playbooks by Collection**, **Search Playbook by Name**, or **Search Playbook by Tags** filters to search for playbooks that you want to map. Select the appropriate playbook and click **Confirm Selection** to save your choice. For example, to search for playbooks that perform actions on emails such as getting the reputation of an email address, type `email` in the **Search Playbook by Name** filter, select the **Get Email Reputation** playbook, and click **Confirm Selection**:



Note: Ensure that the playbooks that are being mapped do not use any parameters. Usage of input parameters might cause the remote playbook to fail.

To change the shared tenant to which this playbook is mapped, from the **Select Shared Tenant** drop-

down list, choose the appropriate tenant:



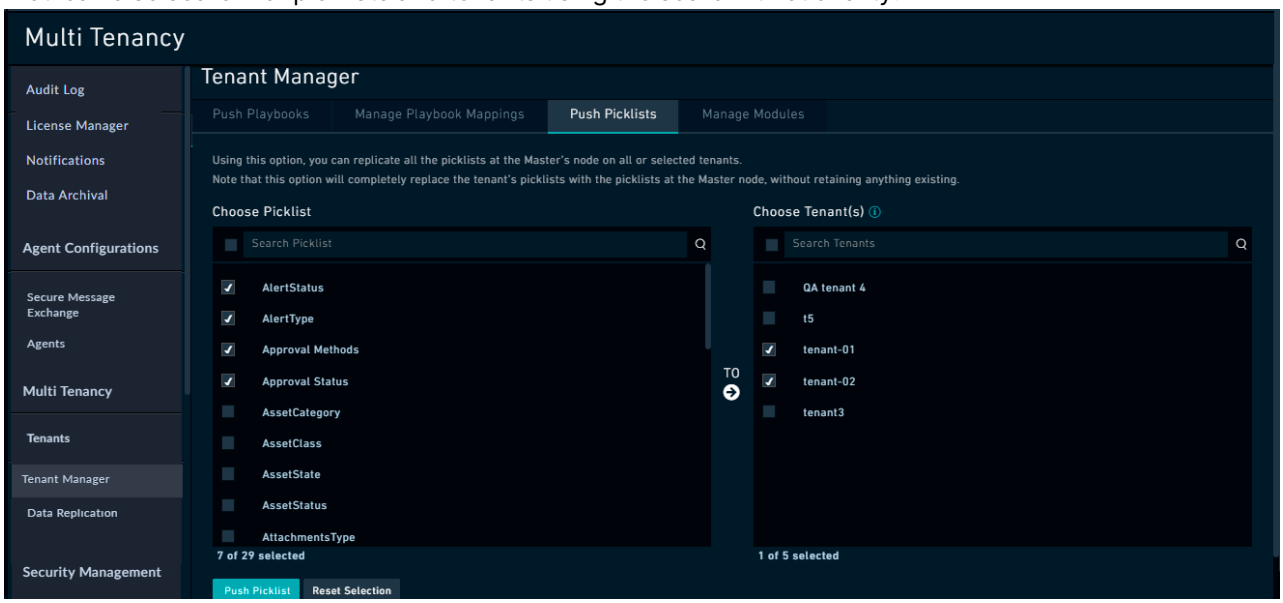
d. **Deleting an existing mapping of playbooks with shared tenants**: Supported for only shared tenants. Click the **Delete Mapping** icon in the row of the mapped playbook to delete the mapping between that playbook and a shared tenant.

Pushing picklists from the master node to the dedicated tenant node(s)

You can push picklists from the master node to the selected dedicated tenant node(s). This operation will replace the picklists that are present in the selected tenant node(s) with the picklists from the master node.

1. Log on to your FortiSOAR master node as an administrator and click the **Settings** icon to open the `System` page.
2. In the `Multi Tenancy` section, click **Tenant Manager** in the left menu.
3. Click the **Push Picklists** tab.
4. From the **Choose Picklist** list, select the picklists that you want to push from the master node to the tenant node(s).
5. From the **Choose Tenant(s)** list, select the tenant(s) to which you want to push the picklists and click the **TO** arrow.
   The **Choose Tenant(s)** list displays only those tenants that are active, whose connections are verified, and which have granted permission to the master to modify their MMD.

You can also search for picklists and tenants using the search functionality.



6. Click **Push Picklists**.
   Important: Clicking **Push Picklists** completely replaces the picklists at the tenants' picklists (none of the existing picklists will be retained at the tenants' node) with the master's picklists.

Editing a dedicated tenant's model metadata at the master and then pushing the model metadata to remote tenants

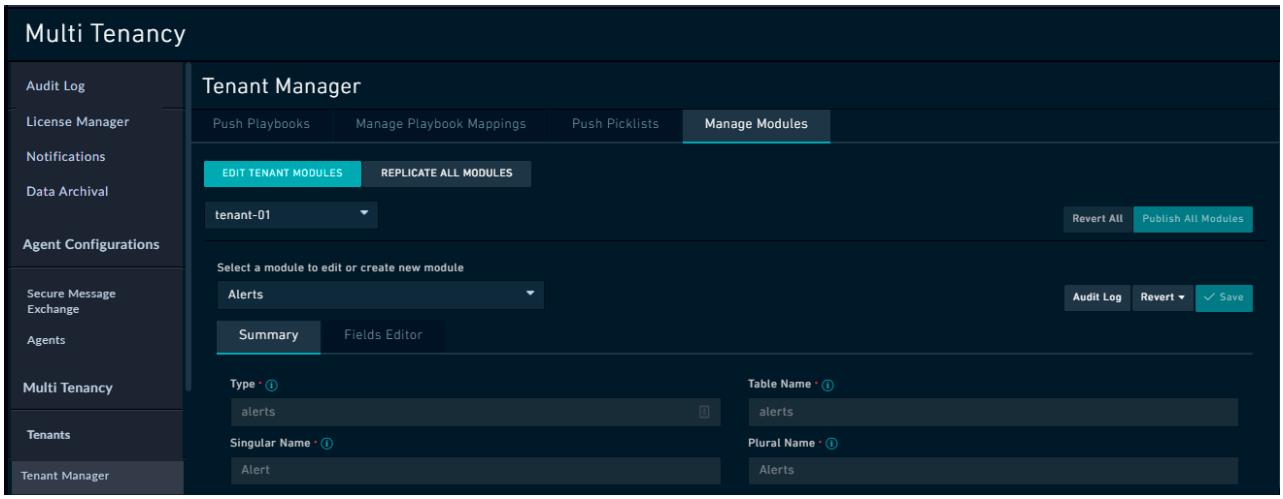
The master node can make changes to the model metadata (MMD) of dedicated tenants and push those module changes and picklist changes to the dedicated tenant node. Similarly, if dedicated tenants make any changes to their MMD, those changes will also reflect on the master.

To allow the master node to remotely edit the tenant's modules and push picklists, the tenant requires to grant permission to the master to control its MMD. This is managed by the **Allow Module Management** button on the `Master Configuration` page of the tenant node. To allow the master node to control the mmd of the tenant's node, ensure that the **Allow Module Management** button is set to **YES**.

On the master node, do the following to update tenants' mmd:

1. Log on to your FortiSOAR master node as an administrator and click the **Settings** icon to open the `System` page.
2. In the `Multi Tenancy` section, click **Tenant Manager** in the left menu.
3. Click the **Manage Modules** tab.
   By default, the **Edit Tenant Modules** button will be highlighted.
4. At the top-right of the `Edit Tenant Modules` page, from the **Select a Tenant** drop-down list, choose the tenant whose MMD you want to modify.
5. From the **Select a module to edit or create a new module** drop-down list, either create a new module or choose the module that you want to modify for the selected tenant.
   Note: The picklists that appear in the Fields Editor tab for the selected tenant are the picklists of the master node. Therefore, you must ensure that the picklists between the master and tenant nodes are synchronized for successful modification of a tenant MMD.
   For example, if you are adding a new module, then you must ensure that you have added any picklist that is

associated with the new module at the master node, and then use the **Push Picklist** tab to push the picklist to the tenant nodes that require that picklist. This ensures that the picklists are synchronized between the master and the tenant nodes.



6. Add or modify fields or attributes that you want to change in the tenant's MMD and click **Save**.
   This puts the changes in the *Staging* state for both the master and the tenant.
7. To update the database and make the changes permanent in both the master and tenant environments, click **Publish All Modules**.
   Important: Once you click **Publish**, the publishing operation begins in the tenant environment. At this time, FortiSOAR displays a message stating the same, and until the publishing operation is completed, users are unable to work in FortiSOAR. Therefore, it is recommended that you should send a prior notification to all users of a publish.
   Also, note that if on the master node a module is marked to be enabled for the recycle bin and the master pushes this module to the tenants, then that module gets enabled for the recycle bin also at the tenants' end.

Tenants might be required to edit their system view templates (SVTs) to view the fields that are added to records based on the changes made in the MMD. An example of how to modify an SVT is given in the Setting up a customer who has multiple sites topic, in *Step 2: Extend the Tenants Module and then edit the required SVTs* section. Step 2 demonstrates adding a custom field.

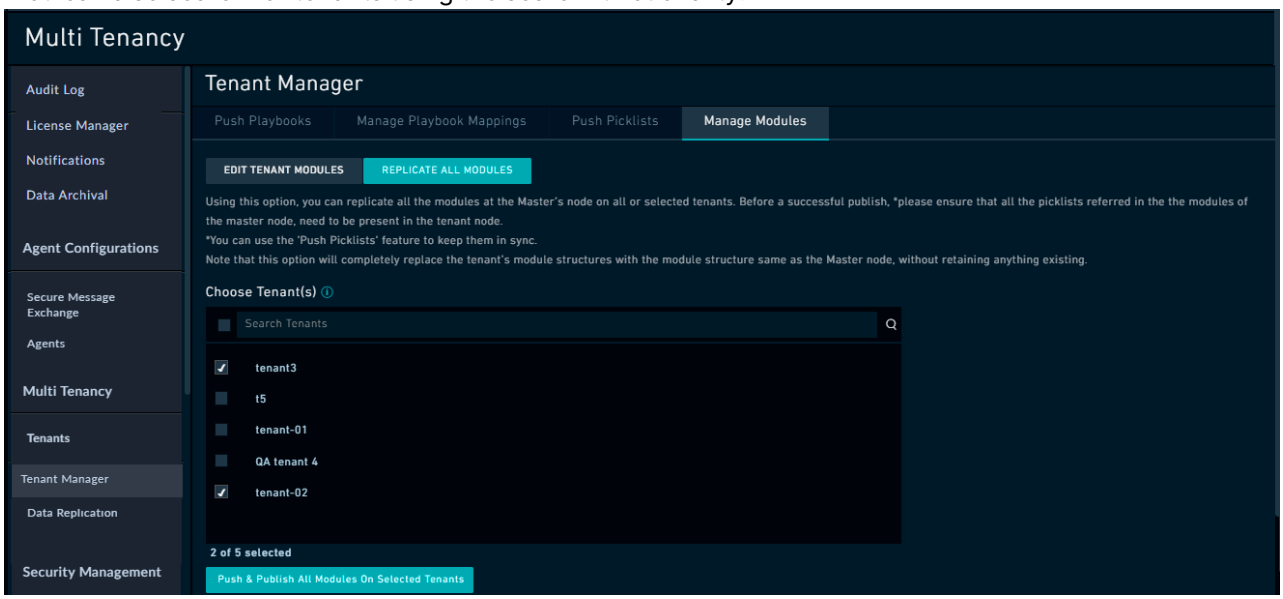
You can also push all modules from the master node, i.e., replicate the module structure of the master on the selected tenant node(s). The ability to replicate the module structure makes onboarding new tenants from the master node effective.

This operation will completely replace the tenant's module structure with the module structure (including all the picklists) of the master node.

In case you want to replace the complete tenant MMD with the master's MMD, do the following on the `Manage Tenant Modules` page:

1. Click the **Replicate All Modules** button.
2. From the **Choose Tenant(s)** list, select the tenant(s) whose MMD you want to replace with the master's MMD.
   The **Choose Tenant(s)** list lists only those tenants that are active, whose connections are verified, and who have granted permission to the master to modify their MMD.

You can also search for tenants using the search functionality.



3. Click **Push & Publish All Modules on Selected Tenants**.
   Warning: Clicking **Push & Publish All Modules on Selected Tenants** completely replaces the selected tenants' module structures (none of the existing module structures will be retained at the tenants' node) with the master's module structures.

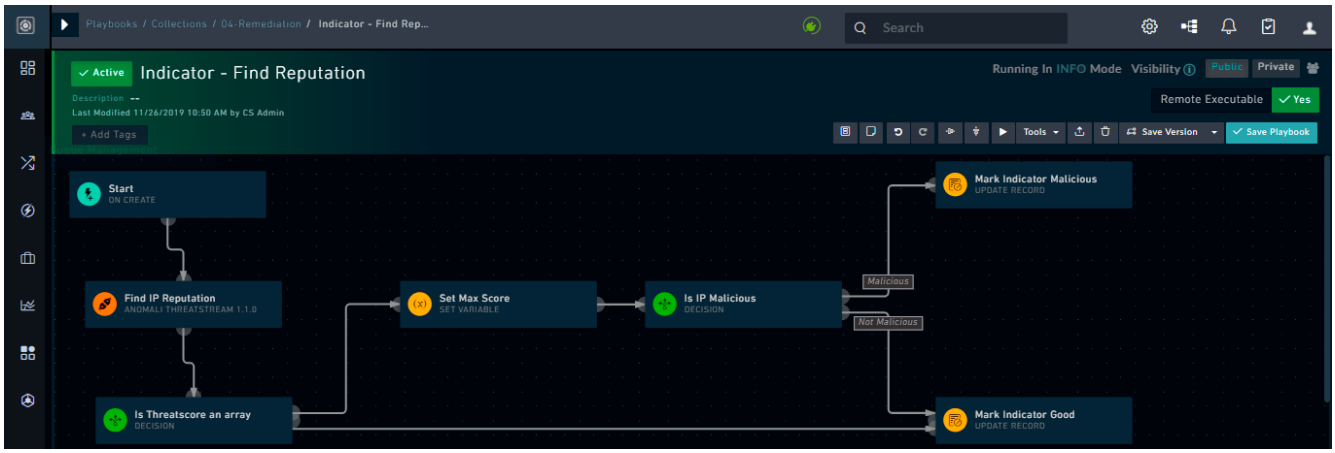
Executing remote playbooks at the tenant node from the master node

To allow remote execution of a playbook, i.e., to allow a FortiSOAR master node to execute a playbook on FortiSOAR tenant nodes, do the following on the master node:

Set the Remote Executable flag:

Enable the **Remote Executable flag** on the playbooks that you want to enable for remote execution from the master node.

In the Playbook Designer, open the playbook that you want to execute remotely, and click to enable the **Remote Executable** box.



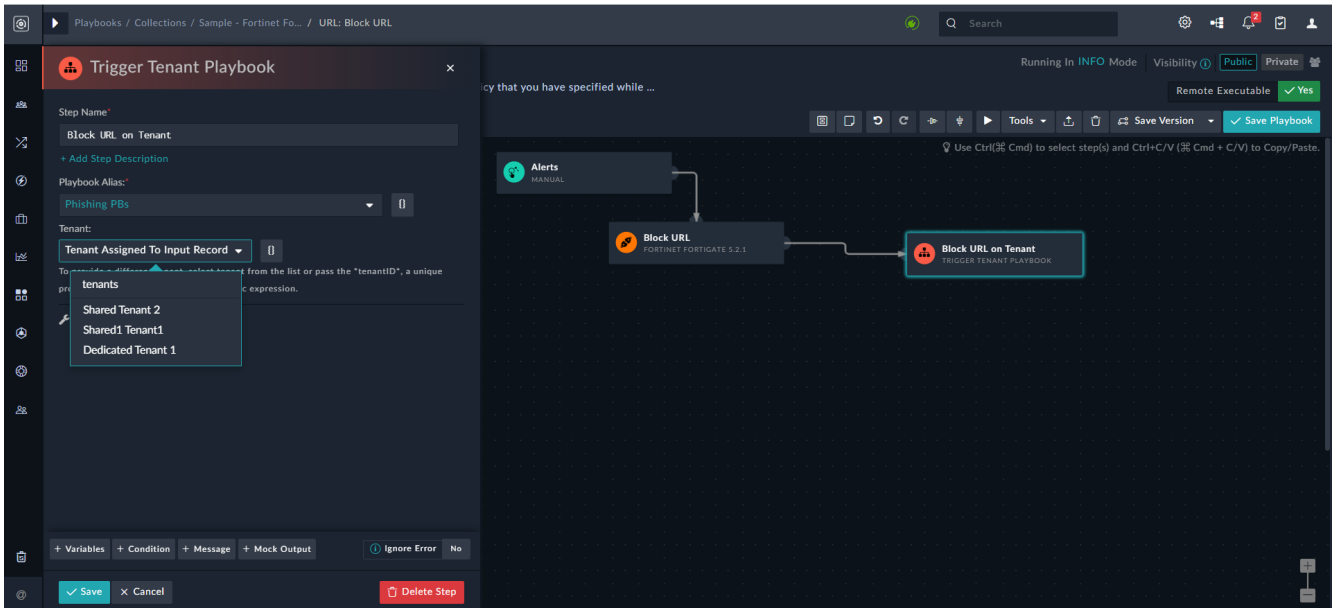
Alternatively, you can also enable the **Remote Executable** box in the playbook on the tenant node.

A **Remote Executable** column is also added to the `Grid` view of the `Playbooks Collection` page. This column displays whether the playbook is enabled for remote execution from the master node. If the playbook can be remotely executed, then the **Remote Executable** column displays a tick in a green circle. If the playbook cannot be remotely executed, then the **Remote Executable** column is blank.

Add a Trigger Tenant Playbook step

Add a **Trigger Tenant Playbook** step in the playbook that you want to trigger from the master node and remotely execute and retrieve details from the tenant node.

In the Playbook Designer, open the playbook in which you want to add the Trigger Tenant Playbook step, so that the master node can remotely execute/trigger the playbook on the tenant nodes, and add the **Trigger Tenant Playbook** step. Click the **Trigger Tenant Playbook** step, and in the **Step Name** field, type the name of the step. The **Playbook Alias** drop-down list displays a list of all the available playbooks for which you have defined alias names. Select the playbook alias that you want the master to remotely execute/trigger on the tenant nodes. Therefore, from the **Playbook Alias** drop-down list, search for and select the appropriate alias name for the playbook that can be remotely executed. This playbook will be mapped to the alias name, and this mapping will be displayed in the `Playbook Mapping` page in the master node.



You can also click the **Dynamic Values** ( ) button beside the **Playbook Alias** to specify the Jinja variable that contains the alias value of the reference playbook to be run on the selected tenant.

The **Tenant** drop-down list displays the tenants on which you can execute this playbook. By default, the selected tenant is the one who was assigned to the record that triggered the playbook, i.e., the **Tenant Assigned To Input Record** option. You can use the **Tenant** drop-down list to select a different tenant (both dedicated and shared tenants) on which to run the playbook. You can also add a `tenantId`, which is a unique property of a tenant node, using a dynamic expression.

The **Loop** variable has been removed from the Trigger Tenant Playbook step. If you want to loop through records, see the Use case of how to loop through records topic.

Ensure that you push the appropriate playbooks to the tenant node. See the Pushing playbooks from the master node for remote execution topic.

Ensure that you do not use any parameters in the playbook that you want to execute remotely. Usage of input parameters in remote executable playbooks causes that playbook to fail as the parameters are not passed to the tenant nodes.

Use case of executing remote playbooks

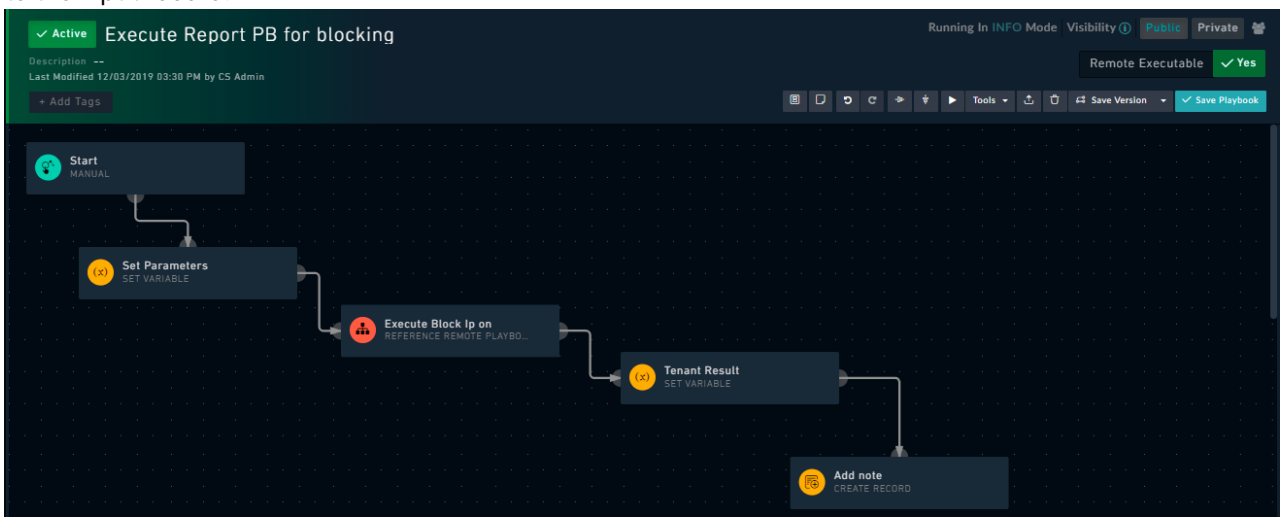
Consider a scenario where there is a master node and two tenant nodes, Tenant 1 has VirusTotal as its threat intelligence tool, and Tenant 2 has IBM xForce as its threat intelligence tool.

This use case demonstrates how a remote playbook is executed from the master node to a tenant node, and the result of the playbook is passed back to the master node. In this case, the master node executes a playbook, using the Tenant 1 record, that has a trigger tenant playbook step, with an alias mapped to Tenant 1, which in turn executes the playbook alias on Tenant 1, and returns the result of the playbook to the master node.

On the master node, create a playbook as follows:

1. **Trigger step**: Manual Trigger. This playbook will be triggered when you select records and execute this playbook. Also, ensure that you select the **Run Separately For Each Selected Record** option in this step.
2. **Set Variable step**: Set the variables that should be passed from the master node to the tenant node. You should add this step just before the "Trigger Tenant Playbook" step.
   For example, to extract the IP address from the record, add `ipAddr` as a variable, whose value will be `{{vars.input.params.ipAddress}}`.
3. **Trigger Tenant Playbook step**: In this step, ensure that you have mapped the playbook alias correctly and select the appropriate tenant on which you want to execute the playbook. By default, the selected tenant is the one who was assigned to the record that triggered the playbook, i.e., the **Tenant Assigned To Input Record** option. You can use the **Tenant** drop-down list to select a different tenant on which to run the playbook. You can also add a `tenantId`, which is a unique property of a tenant node, using a dynamic expression.
4. **Set Variable step**: Set a variable to consume the result from the tenant node. For example, add `execution_result` as a variable, whose value will be `{{vars.steps.<step_name>.keyname}}` such as, `{{vars.steps.<step_name>.data}}`.

5. Further, based on the execution result, you can create additional steps, such as adding a note or comment to the input record.



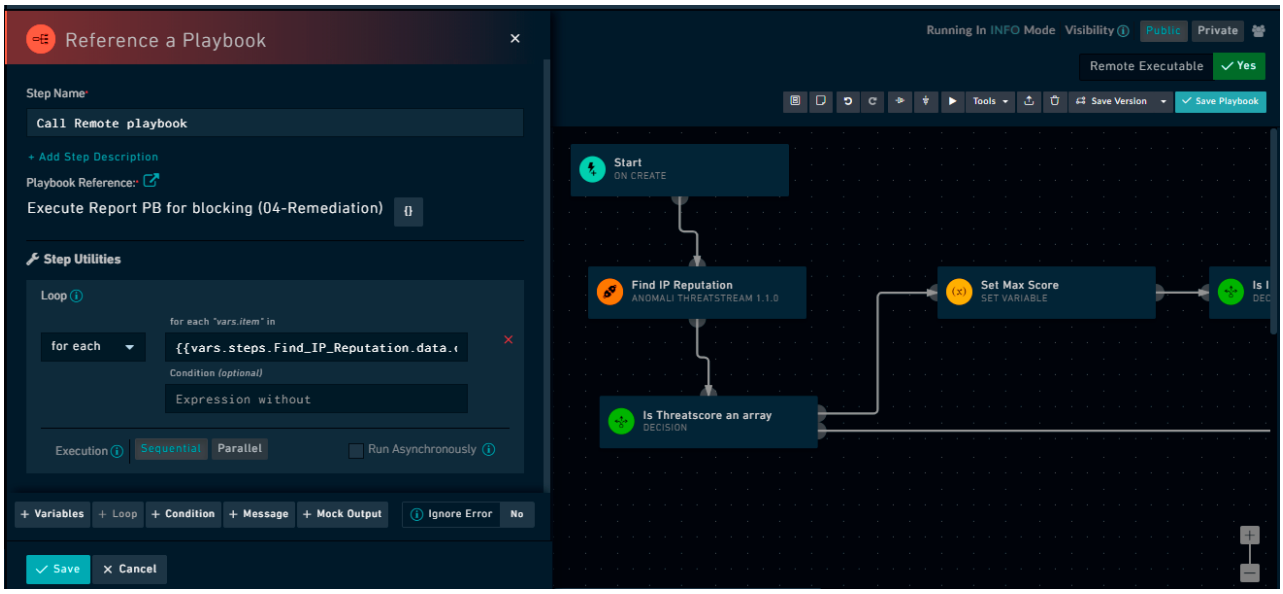
On the tenant node, you can either create a playbook using the following steps or push the created playbook (using the following steps) from the master node to the tenant node. You must ensure that the **Remote Executable flag** is enabled for this playbook.

1. **Trigger step**: Referenced, since the master node will trigger this playbook.
2. **Set Variable step**: Set this variable to consume the input sent from the master node. For example, to consume the IP address sent from the master node, add `ipAddr` as a variable, whose value will be `{{vars.ipAddr}}`.
3. **Connector step**: Add the VirusTotal connector step (since Tenant 1 has the VirusTotal threat intelligence tool) to analyze the IP address received from the previous Set Variable step and send its result to the next step.
4. **Set Variable step**: Set a variable so that the master node can consume the result from the tenant node. For example, add `execution_result` as a variable, whose value will be `{{vars.steps.<step_name>.keyname}}` such as `{{vars.steps.<step_name>.data}}`.

Use case of how to loop through records

Consider a scenario where on the master node there are 5 records, belonging to different tenants, on which you want to run an *Investigative* playbook. To achieve looping through records, do the following:

1. In the *Investigate* playbook add a "Reference A Playbook" step that contains the looping logic as shown in the following image:



The *Investigate* playbook in turn calls another playbook that contains the Trigger Tenant Playbook step,

2. In the called playbook, i.e., the playbook with the "Trigger Tenant Playbook" step, you must specify the `tenantId` in the **Tenant** field.

Importing and exporting of tenant, agent, and router details

You can use the Export and Import Wizards and Import Wizard to export and import tenant, agent, and router schemas along with records from MSSP systems so that you can recreate a near-exact replica of a FortiSOAR environment. For more information of the Export Wizard and Import Wizard, see the **Application Editor** chapter in the "Administration Guide." Steps, in brief, for data import and export are as follows:

1. Configure the MSSP setup.
2. Using the Export Wizard, export the tenant, agent, and router schemas, along with the records from the master machine in the MSSP setup.
3. Using the Import Wizard, import the exported schemas and records data to a new master node.
4. Enable a tenant on the new master node and update the master configuration on the tenant node, i.e., the tenant node should point to the new master node.
5. Delete the tenant configuration from the old master node.
   This is needed, since after importing the configuration the tenant node is reachable from both the old and the new master node which should not be the case, and ideally the tenant should be remotely connected only to the new master node. Therefore, you should delete the tenant configuration from the old master machine.

Extending the Tenants module

The `Tenants` module (included by default in FortiSOAR) is now extendable, i.e., you can add fields that you require to the tenant's module. The steps you can follow to extend the `Tenants` module are mentioned in *Step 2:*

Extend the Tenants Module and then edit the required SVTs section that is part of the Setting up a customer who has multiple sites section.

Additional Configurations

Changing the hostname

The FortiSOAR Configuration Wizard is available only on the first `ssh` login. If at a later stage, you require to change the hostname of your FortiSOAR secure message exchange, then you can use the FortiSOAR Admin CLI (`csadm`). For more information on `csadm`, see the *FortiSOAR Admin CLI* chapter in the "Administration Guide."

You must run `csadm` immediately after deployment of the FortiSOAR Virtual Appliance. Running this after the FortiSOAR system is active and online might lead to data loss.

To change the hostname, ensure that the hostname is resolvable and then do the following:

1. SSH to your FortiSOAR VM and login as a *root* user.
2. To change your hostname of your FortiSOAR secure message exchange, type the following command:
   `# csadm hostname --set[<hostname>]`
   This command changes your current hostname to the new hostname that you have specified, sets up the message broker, regenerates certificates, and restarts FortiSOAR services.

When you run the `# csadm hostname --set[<hostname>]` command on the secure message exchange, then all previously configured tenant user accounts and exchanges will be deleted. Also, if there are any pending messages, then they will be lost. After you have completed changing the hostname, you will have to reconfigure the tenants from the master node.

Important: When you change the hostname for your ***Secure Message Exchange VM***, you must ensure that the certificate the secure message exchange uses for communication with the FortiSOAR nodes has the same Common Name (aka CN) or is a wildcard certificate for the domain.

If you get the following error, when you are changing the hostname on your secure message exchange:

`unable to connect to epmd (port 4369) on rabbitmqserver: nxdomain (non-existing domain)`

Then you must terminate the current session, open a new session, and then run the `# rabbitmqctl status` command.

After the hostname has been reset, when users execute playbooks with an external manual input link, it is observed that the link that is generated in the email contains the original FQDN (hostname) rather than the one that has been updated. Therefore, users who are required to provide the input, have to manually update the FQDN (hostname) in the manual input link present in the email.

Monitoring and administering your multi-tenancy environment

Setting up High Availability in case of a multi-tenancy environment

To configure high availability for your master or tenant nodes, you must first configure high availability, and only then add the master or tenant configurations. For information about High Availability in FortiSOAR, see the *High Availability and Disaster Recovery support in FortiSOAR* chapter in "Administration Guide."

Setting up High Availability of the Secure Message Exchange

If the Secure Message Exchange (SME) fails, you can configure an alternate SME instance. Update the SME details on both master and tenant nodes to reference the new SME's IP address, certificate, etc. The master node automatically reconfigures queues and exchanges on the new SME instance.

Note: This process involves SME downtime until the new SME is provisioned. Messages pending delivery to the tenant will need to be manually resent. To avoid downtime, you can configure the SME in an Active-Passive or Active-Active HA setup.

Internally, SME uses RabbitMQ for message persistence and routing of messages across the FortiSOAR nodes. You can implement HA using standard RabbitMQ clustering methods. For example:

- Use shared storage (NAS or SAN) for the RabbitMQ data directory, `/var/lib/rabbitmq` across two nodes
- Use Pacemaker or similar tools for monitoring and automatic failover.

See the Disaster Recovery and High Availability 101 article for details.

FortiSOAR SME also supports clustering. Clustered instances should be fronted by a TCP Load Balancer such as HAProxy, and clients should connect to the cluster using the load balancer's address. For information about High Availability in FortiSOAR, see the *High Availability and Disaster Recovery support in FortiSOAR* chapter in "Administration Guide."

Prerequisites to configuring High Availability

- All nodes of a cluster must be on the same FortiSOAR version.
- Ensure that all nodes in the cluster use a uniform hostname format — either fully qualified domain names (FQDN) or short hostnames — for consistent cluster communication.
  IMPORTANT: Ensure that all cluster nodes use a consistent hostname format (for example, either node1.example.com or simply node1 on all nodes) to maintain stable cluster communication.
- All nodes of a cluster must be DNS resolvable from each other.
- Run the `tmux` command so your upgrade is not affected if your session times out.

- If you have a security group (AWS) or an external firewall between the HA nodes, then you must open the following ports between HA nodes on AWS or the external firewall:

Service TCP	Port
SSH	22
MQ Peer Discovery	4369
MQ inter-node	4369
MQ TLS	5671
Erlang distribution client ports	35672-35682 (range)

- Clustered instances should be fronted by a TCP Load Balancer such as HAProxy, and clients should connect to the cluster using the load balancer's address.

Configuring High Availability

While the following steps assume two SME instances, we recommend deploying an odd number of nodes (1, 3, 5, etc.) for an HA cluster. Additionally, note that we refresh the certificates are refreshed during this process. Existing Agents or Tenants will be disconnected and must reconnect with updated certificates.

For releases 7.6.2 onwards

1. Deploy and configure the first and second secure message exchange. See the Deploy and configure the secure message exchange section.
2. Ensure all hostnames (including the load balancer) are listed in the Subject Alternative Names (SAN). Additionally, server certificates must be signed by the same Certificate Authority (CA). To achieve this, follow these steps based on your SME version:
   a. Update SAN on each SME instance:
      i. Add hostnames, including the load balancer's hostname, to the SAN using the command:
      `csadm hostname --san <comma-separated-list-of-hostnames>`
      ii. Regenerate the MQ certificates using the command:
      `csadm mq certs generate --skip-ca-generation`
3. Open the necessary ports in the firewall on all the cluster nodes, using the following command:
`csadm ha allowlist --nodes <space-separated-node-names>`
4. Create the RabbitMQ cluster using one of the following methods:
   - *Join using the `csadm ha` command*:
      i. Run the following command on the primary node:
      `csadm ha join-cluster --primary-node <primary-hostname>`
   - *Join using the config file*:
      i. On the primary node, generate the config:
      `csadm ha export-conf`
      ii. Copy the exported config file to the secondary node.
      iii. On the secondary node, run the following command:
      `csadm ha join-cluster –-config-file <conf-file-path>`

5. (Optional) If you wish to enable mTLS, then you must enable it on all the SME instances using the command:
   `csadm mq mtls enable`
   After enabling mTLS, generate client certificates. These certificates are required for connecting to the SME.

For releases prior to 7.6.2

1. Deploy and configure the first secure message exchange. See the Deploy and configure the secure message exchange section.
2. Deploy and configure the second secure message exchange. See the Deploy and configure the secure message exchange section.
3. All referenced hostnames are present in the Subject Alternative Names (SAN). Additionally, server certificates must be signed by a common Certificate Authority (CA). To achieve this, follow these steps based on your SME version:
   For releases 7.6.0 and 7.6.1:
   a. Update SAN using on every SME instance:
      i. Add hostnames, including the load balancer's hostname referenced in the SAN using the command:
      `csadm hostname --san <comma separated list of hostnames`
      ii. Regenerate the MQ certificates using the command:
      `csadm mq certs generate --skip-ca-generation`
   b. To keep the same '`cacert`' across the HA cluster, use the following steps:
      i. Select one instance as your primary instance.
      ii. From the selected primary instance copy `/opt/cyops/configs/rabbitmq/ssl/cyopsca/cacert.pem` and `/opt/cyops/configs/rabbitmq/ssl/cyopsca/private/cakey.pem` to the `/home/csadmin` folder of the other SME instances.
      iii. Place these certificates in the appropriate location using the following commands:
      ```
      mkdir -p /opt/cyops/configs/rabbitmq/primary/private
      cp /home/csadmin/cacert.pem /opt/cyops/configs/rabbitmq/primary
      cp /home/csadmin/cakey.pem /opt/cyops/configs/rabbitmq/primary/private
      ```
      iv. Refresh the `cacert` and `server` certs for MQ using the command:
      `csadm mq ca reset-with-primary`

   For releases prior to 7.6.0:
   1. Update SAN on the SME instance:
      a. Edit the `/opt/cyops-rabbitmq/configs/ssl/openssl.cnf` file and add DNS.2 as the SME 2 hostname, DNS.3 as the load balancer's hostname in the `[alt_names]` section. DNS.1 is set to the current instance's hostname by default.
      b. Regenerate the MQ certificates using the command:
      `csadm mq certs generate`.
   2. To keep the same '`cacert`' across the HA cluster, use the following steps:
      a. Copy the entire contents of the `/opt/cyops/configs/rabbitmq/ssl/` directory from SME instance 1 to other instances, retaining permissions.
      b. Restart the RabbitMQ server and all its related services using the command:
      `# systemctl restart rabbitmq-server`
4. Open the necessary ports in the firewall, using the following commands:
   a. `firewall-cmd --zone=public --add-port=4369/tcp --permanent`
   b. `firewall-cmd --zone=public --add-port=25672/tcp --permanent`

c. `firewall-cmd --reload`
To use alternate ports, refer to the RabbitMQ: Port Access article.
At this point, all your instances share the same CA certificate, and the server certificates are signed by this CA certificate. Additionally, the subject alternative names contain all the hostnames that will be used.

5. To create RabbitMQ cluster, follow these steps, which are in accordance with the official RabbitMQ clustering guide, on the chosen secondary node.
   a. Stop the RabbitMQ app using the command:
   `rabbitmqctl stop_app`
   b. Reset the RabbitMQ app using the command:
   `rabbitmqctl reset`
   c. Copy the contents of the `/var/lib/rabbitmq/.erlang.cookie` file on the primary instance, then paste it into the same file on the secondary instance, ensuring there is no trailing whitespace or newline
   Use `truncate -s -1 /var/lib/rabbitmq/.erlang.cookie` if there is a newline.
   d. Restart the RabbitMQ server using the command:
   `systemctl restart rabbitmq-server`
   e. Stop the RabbitMQ app using the command:
   `rabbitmqctl stop_app`
   f. Copy the instance name of the primary instance using the command:
   `rabbitmqctl eval 'node().' | sed "s/'//g"`
   NOTE: The output of this command displays the node name in the format `rabbit@<hostname>`. This hostname must be resolvable from all nodes in the cluster.
   g. Join the cluster using the command:
   `rabbitmqctl join_cluster <instance-name from the previous step>`
   h. Start the RabbitMQ app using the command:
   `rabbitmqctl start_app`
   i. Verify cluster status using the command:
   `rabbitmqctl cluster_status`
   Check for running instances and disk instances.
   At this point, the RabbitMQ cluster is successfully formed. Now, set a policy to mirror queues across the cluster. To do this, you need to set the policy for tenant or agent vhosts.
6. Add a policy to setup mirroring using the command:
   `rabbitmqctl set_policy tenants-ha "^queue.postman\." '{"ha-mode":"all","ha-sync-mode":"automatic"}' -p <vhost>`
   NOTE: The `set_policy` command must be set for every vhost. When an agent is added or a tenant is onboarded, a new dedicated vhost gets created. Therefore, after adding an agent or onboarding a tenant, you need to run `set_policy` for any newly created vhosts. To list the vhosts, use the following command:
   `rabbitmqctl list_vhosts --no-table-headers --quiet | grep -e "^vhost_*`
   If you wish to enable mTLS, then you must enable it on all the SME instances using the command:
   `csadm mq mtls enable`
   Once mTLS is enabled, you must generate client certificates. These certificates are mandatory for adding the SME to your FortiSOAR instance.

Setting up HAProxy as a TCP load balancer fronting the two clustered nodes

The following steps list out the steps to install "HAProxy" as a load balancer on a Virtual Machine:

1. `# yum install haproxy`
2. Edit the `/etc/haproxy/haproxy.cfg` file to configure your HAProxy as follows:

```
defaults common_defaults
  log global
  timeout connect 5s
  timeout client 60s
  timeout server 60s
  option forwardfor
  option httpchk

defaults https_defaults from common_defaults
  mode http
  http-check connect port 443 ssl
  http-check send meth GET uri /auth/node?param=active
  http-check expect status 200

defaults tcp_defaults from common_defaults
  mode tcp
  http-check connect port 443 ssl
  http-check send meth GET uri /auth/node?param=primary
  http-check expect status 200




# ---------------------------------------------------------------------
# Front end for 443 i.e., nginx requests
# ---------------------------------------------------------------------
frontend main from common_defaults
  bind *:443 ssl crt <path-to-certificate>
  use_backend ha_cluster_443

# ---------------------------------------------------------------------
# Optional front end for 5671 i.e., RabbitMQ requests (for embedded SME)
# ---------------------------------------------------------------------
frontend mq from common_defaults
  mode tcp
  bind *:5671
  tcp-request inspect-delay 5s
  tcp-request content accept if { req_ssl_hello_type 1 }
  use_backend ha_cluster_5671

# ---------------------------------------------------------------------
# Backend for front end main
# ---------------------------------------------------------------------
backend ha_cluster_443 from https_defaults
  server <server-1-hostname> <server-1-ip-address>:443 check inter 30s ssl verify none
  server <server-2-hostname> <server-2-ip-address>:443 check inter 30s ssl verify none

# ---------------------------------------------------------------------
# Backend for front end mq
# ---------------------------------------------------------------------
```

```
backend ha_cluster_5671 from tcp_defaults
  server <server-1-hostname> <server-1-ip-address>:5671 check inter 30s verify none
  server <server-2-hostname> <server-2-ip-address>:5671 check inter 30s verify none
```

3. To reload the firewall, run the following commands:

```
$ sudo firewall-cmd --zone=public --add-port=<portspecifiedwhilebindingHAProxy>/tcp --
permanent
$ sudo firewall-cmd --reload
```

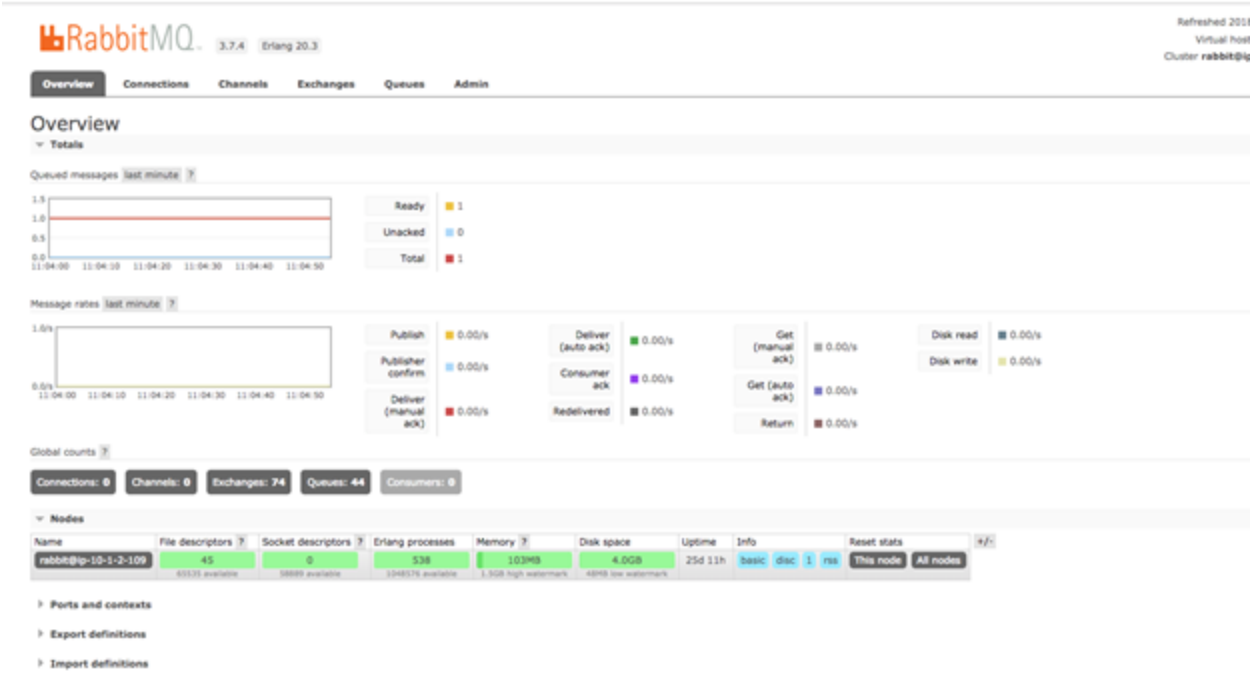
4. Restart `haproxy` using the following command:
   `# systemctl restart haproxy`
5. Use the bind address (instead of the IP address of the node in the cluster) for accessing the FortiSOAR UI. You must also ensure that the CA certificates on both the nodes are the certificates that were provided for secure message exchange configuration.

For more information on load balancers, see the High Availability and Disaster Recovery support in FortiSOAR chapter in the "Administration Guide."

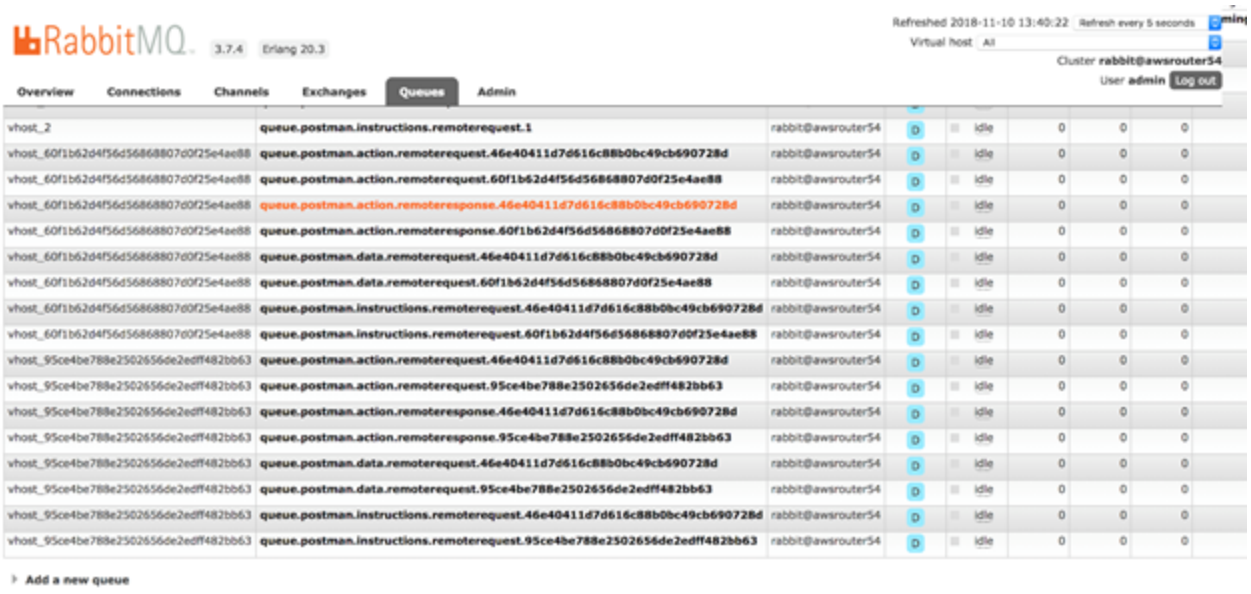
Monitoring the connectivity of the different nodes at the secure message exchange

Once you deploy and configure a secure message exchange, its remote management UI is enabled. The connectivity of the different nodes, messages pending on the data, action and instructions queues, and the message flow rate can be monitored using the remote management UI.

Open `https://<secure message exchange_address>:<management_port>` and connect using your administration credentials. The management port and administration credentials are configurable and specified at the time of configuring the secure message exchange. See the Configuring the Secure Message Exchange section.



All vhosts, exchanges, and queues associated with the node are created with the respective tenant ID.



Backing up and Restoring your FortiSOAR systems

Use the FortiSOAR Admin CLI (`csadm`) to regularly perform backups on master and tenant nodes and to restore the data seamlessly to a new FortiSOAR environment. To perform backup and restore, you must have *root access* on your FortiSOAR system.

For detailed information on backing up and restoring your FortiSOAR system, see the *Backing up and Restoring your FortiSOAR system* chapter in the "Administration Guide."

In case of a multi-tenant configuration, after you have completed restoring your master node or tenant node, you must restart the `postman` service on the new restored system using the following command:
`# systemctl restart cyops-postman`

Use Cases

Setting up a customer who has multiple sites

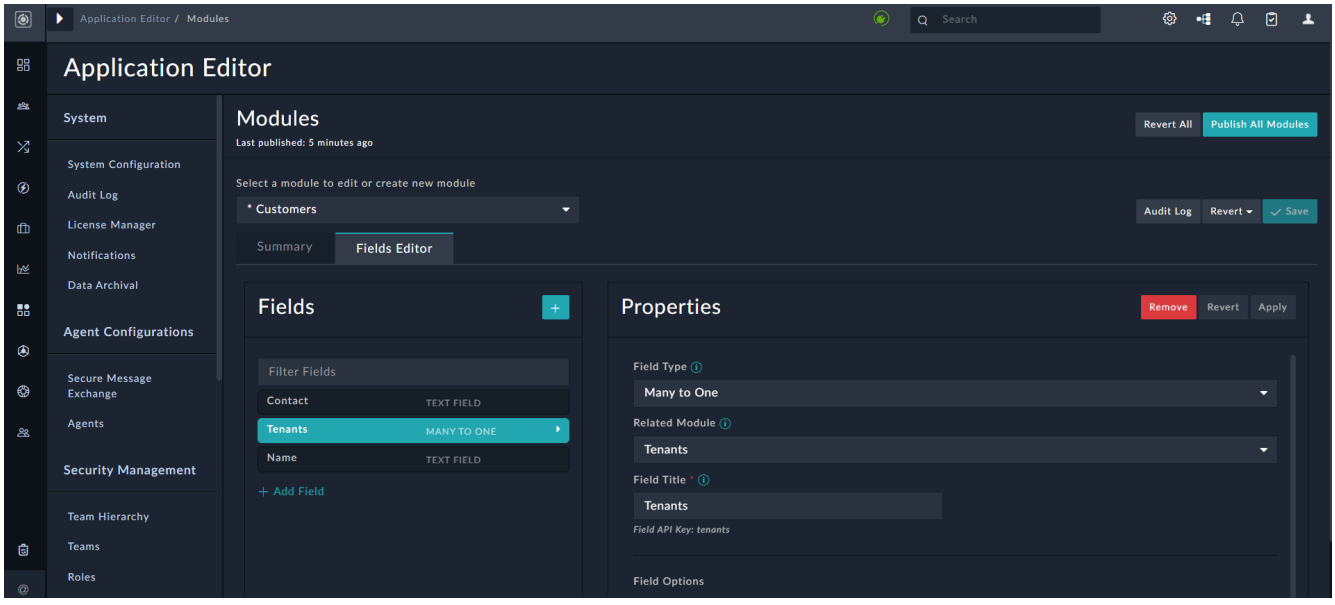
This use case describes the additional steps you need to perform, if the multi-tenant configuration has a customer with multiple sites (tenant). Each site (tenant) in this case has their own FortiSOAR instance.

All these steps must be performed on the Master node.

Step 1: Create a custom "Customers" Module

You must create a new "Customers" module in FortiSOAR by clicking the Setting icon on the top-right of the FortiSOAR UI. In the Application Editor section, click **Modules** and then click **+Create new module** to create a new module.

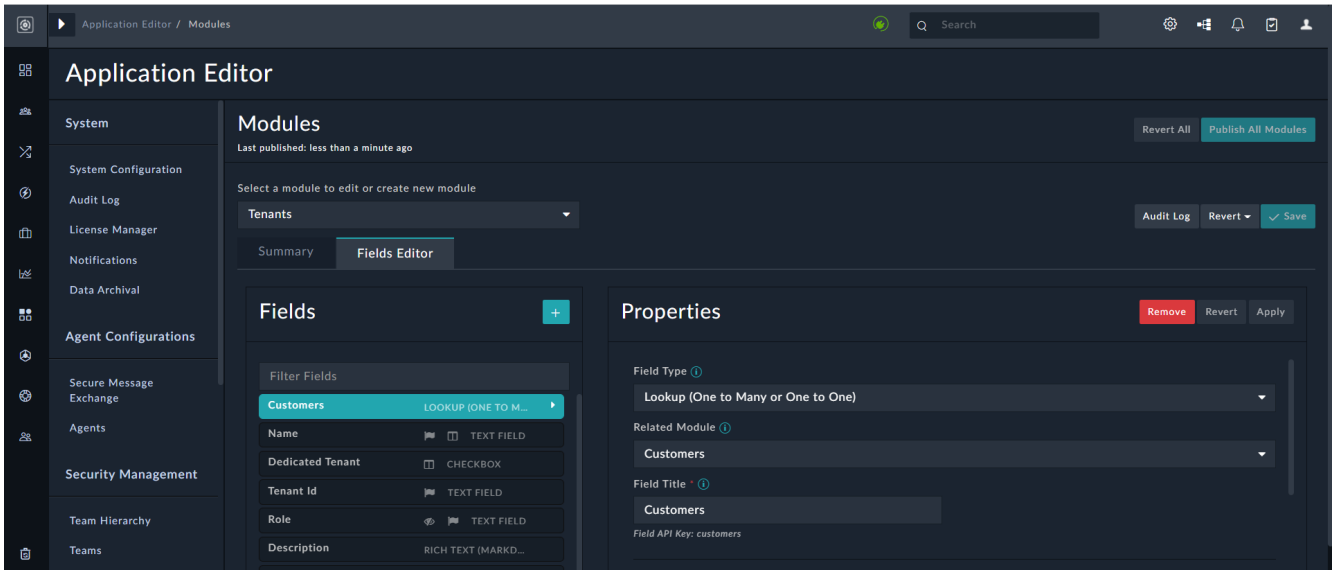
You can add whatever fields you require to this module, however, you must add a **Tenants** field that has a **Many to One** to `customer`.



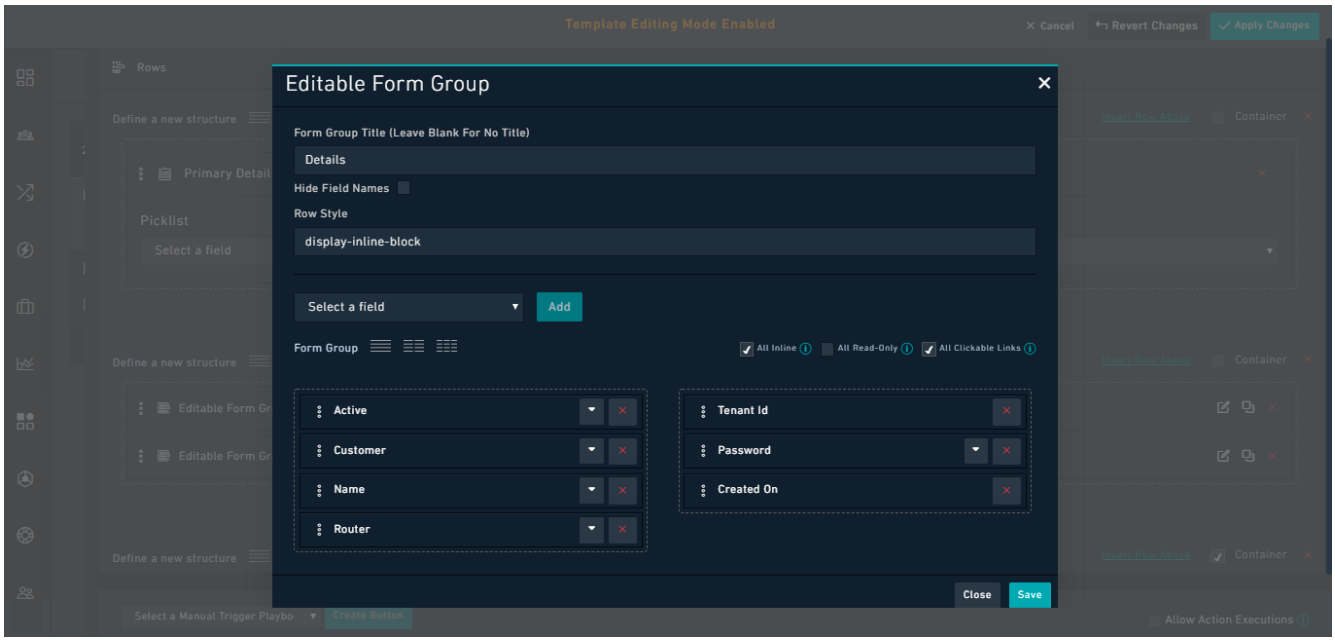
Once you create the Customers module, you must ensure that you assign the administrator's role appropriate permissions on this newly-created Customers module, so that you can add customers and associate tenants with customers.

Step 2: Extend the Tenants Module and then edit the required SVTs

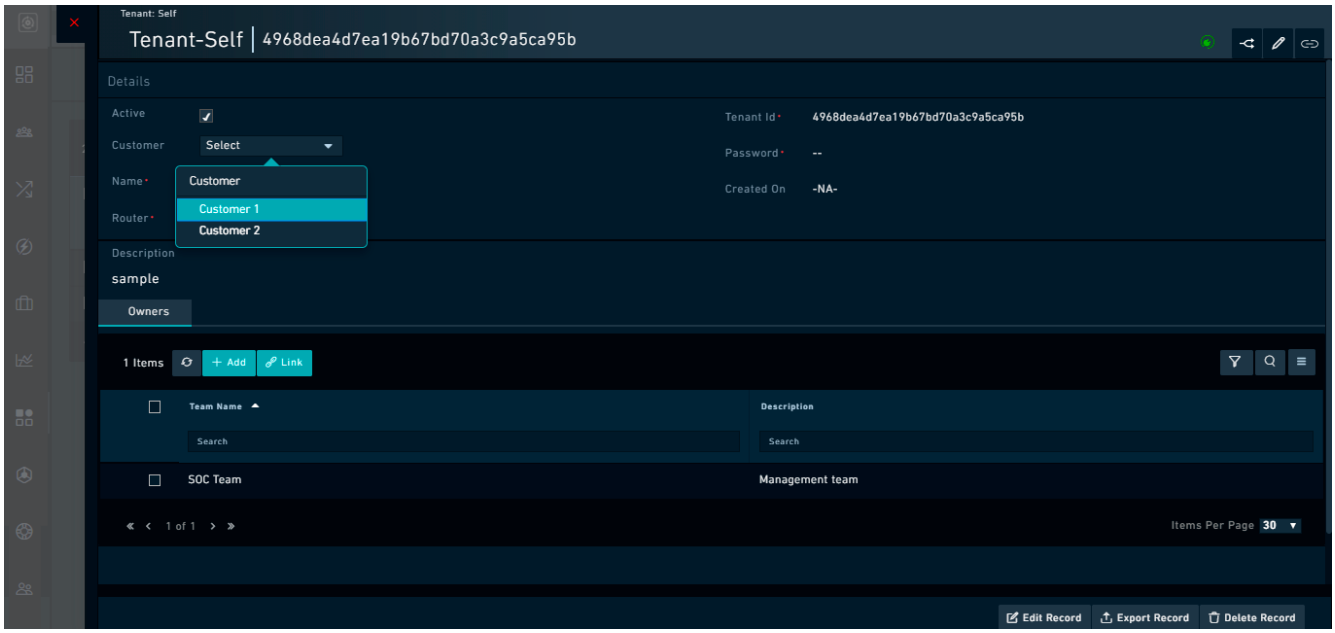
Open the `Tenants` module (included by default in FortiSOAR) in the **Application Editor** and extend this module by adding the **Customer** field. Add the **Customer** field which has the field type set as **Lookup (One to Many or One to one)**.



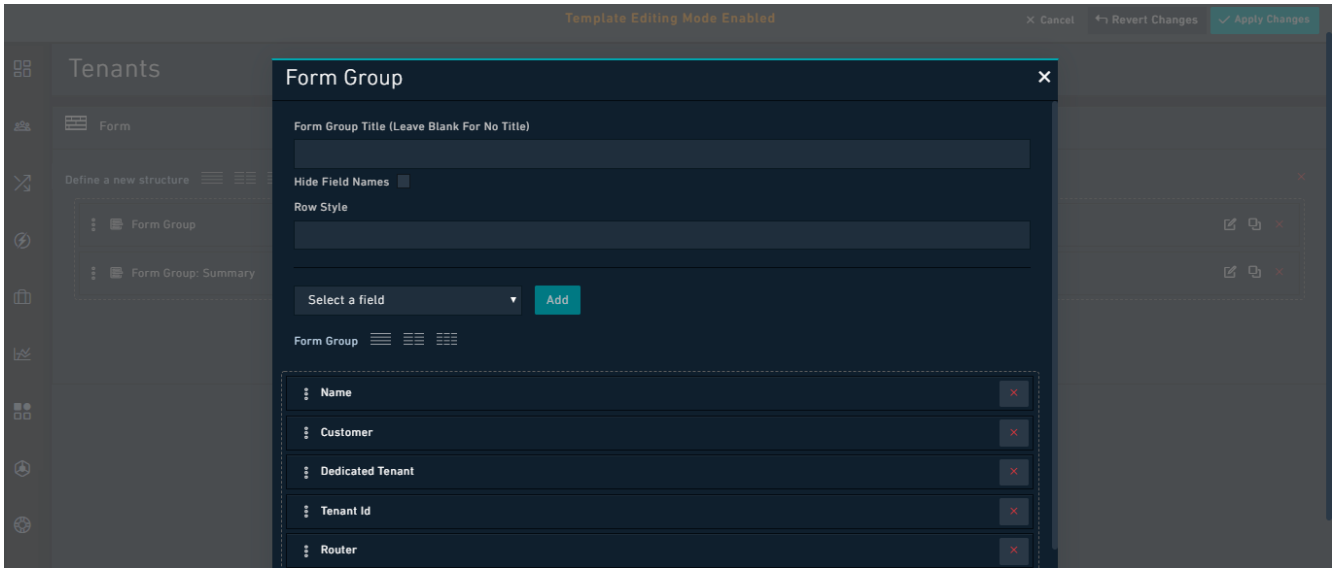
Once you have created the `Customers` module and set up the relationship between the Customers and Tenants modules, you must now edit the system view template (SVT) for the `Tenants` module. Edit the SVT for the `Tenants` module to include the **Customer** field:



The `Tenants` module will now appear as shown in the following image:

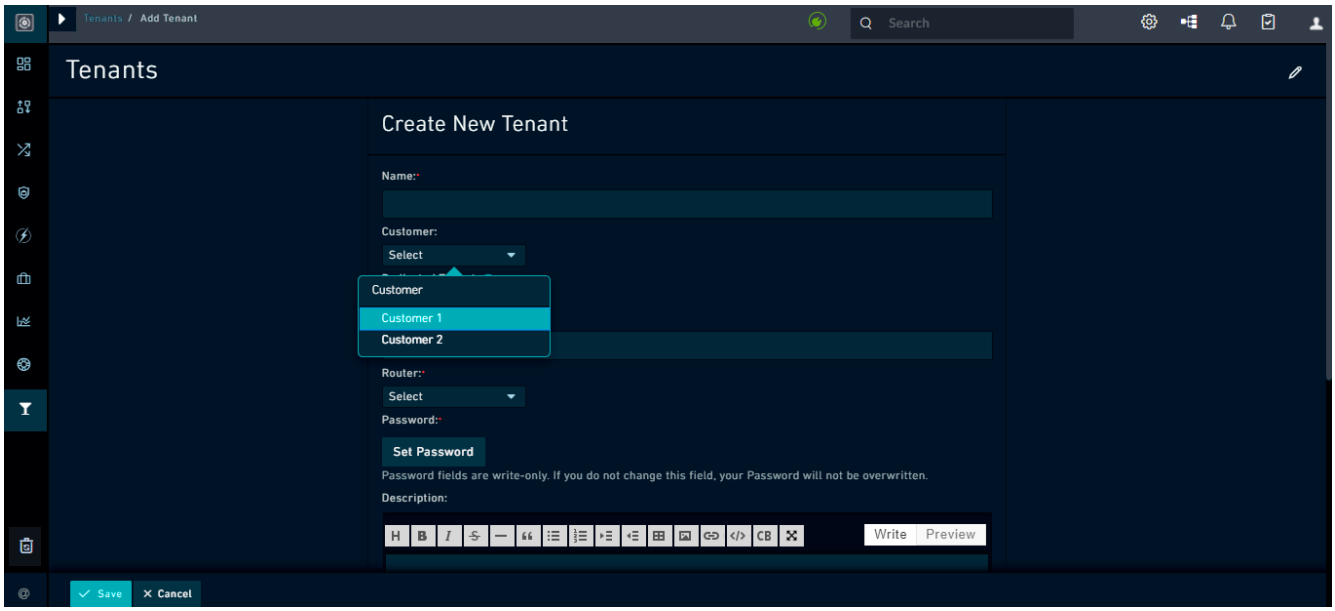


You should also edit the SVT of the *Add Tenant* form to include the **Customer** field, so that when you add a new tenant, you can select the **Customer** to whom the tenant belongs.



Step 3: Configuring the tenants on the Master Node

Now, you can configure the tenants on the master node (see Deploying tenant nodes). However, in this case you must ensure that you select the **Customer** to whom the tenant belongs to create the association between tenants and the customer.


Tenants / Add Tenant
Search
Tenants
Create New Tenant
Name:*
Customer:
Select
Customer
Customer 1
Customer 2
Router:*
Select
Password:*
Set Password
Password fields are write-only. If you do not change this field, your Password will not be overwritten.
Description:
Write
Preview
Save
Cancel

Best practices

Following are the best practices that you should follow while configuring the distributed managed service provider model for multi-tenancy in your environment

- On the master node, you should edit your template (SVT) in modules that are being replicated from the tenant nodes so that you include the Tenant field on `Add Record` Dialog.
  To edit the SVT, navigate to the module for which you want to update the SVT, for example, `Alerts`, open a record and click **Edit Template**. On the `Template Editing Mode Enabled` page, in the appropriate widget, such as `Form Group Details` click **Edit**, and add the `Tenant` field ensuring that this field has been marked as a **Required** field and then click **Add**. Click **Save** to save your changes to the SVT.
- SOC analysts at the master node must at the minimum have `Read` access for the `Tenants` module so that they can investigate any incidents that might occur at the tenant node.
- To assign a team to a tenant, on the master node, edit the tenant record on the tenant's page. In the detail view of the tenant record, add or link a team. Once a team is assigned, the records from that tenant will automatically be assigned to that team.
- If you are creating a record on the master node, ensure that you assign that record to the appropriate tenant.
  Important: Once a tenant is assigned to a record, the assignment of that record cannot be changed. If you require to change the assignment of that record, then you must delete the record and re-assign the record correctly.
- Once playbooks that contain On Update, On Create, or Custom API Endpoint triggers are pushed to the tenant they should be deactivated at the master node. Otherwise, the playbook will be triggered at both the nodes.

Troubleshooting

For troubleshooting the distributed managed service provider model for multi-tenancy issues that you might face, you can use the `postman.log` located at:

`/var/log/cyops/cyops-routing-agent/postman.log`

You can change the logging levels for the postman.log by using the following command:

`# vi /opt/cyops-routing-agent/postman/config.ini` and set the required logging level. By default, the logging level is set to **WARN**.

You can set the following logging levels in the log files:

- **DEBUG**: Low-level system information for debugging purposes.
- **INFO**: General system information.
- **WARN**: Information describing a minor problem that has occurred.
- **ERROR**: Information describing a major problem that has occurred.
- **CRITICAL**: Information describing a critical problem that has occurred.

Deployment Troubleshooting

While connecting to a secure message exchange, retries never stop if the DNS is not resolved

You can continue to retry to connect to the secure message exchange for 10 minutes. A long retry has been specified to ensure that you do not have to restart the service every time the network breaks since all consumers stop after a long network break.

Resolution

You can configure the time for which you will continue retrying to establish a connection with the secure message exchange using the `config` file, located at `/opt/cyops-routing-agent/postman/config.ini`:

```
heartbeat = 120
connection_attempts = 50
retry_delay = 10
```

After adding a tenant on the master, you see a Retry button on the tenant node

This issue can occur due to two reasons:

- Incorrect secure message exchange configuration.
- Incorrect selection of secure message exchange while configuring the tenant on the master node.

Resolution

Correct the secure message exchange configuration or selection and click the **Retry** button.

Configuration Troubleshooting

Failure while configuring master on a tenant node

This issue can occur if you have specified the wrong password or port while configuring the master on the tenant node.

Resolution

On the tenant node, click **Master Configuration** > **Edit Configuration**. On the `Configure Master` dialog and correct the password or TCP port number that you have specified.

If you yet see that the **Enabled** button is in the **NO** state, and you want to enable data replication, toggle the **Enabled** button to **YES**.

Records created at the tenant node do not replicate to the master node even when data replication is turned on

In this case, the FortiSOAR logs contain the following message: `Error message on master "Error: Bad Request for url: <URL> when a field is non-mandatory on tenant and mandatory on master`

This issue occurs if you have switched off replication of a required record field on a tenant node, or the field does not exist on the same module defined at the tenant node. This leads to a failure of record creation on the master node.

Resolution

You must ensure that replication is switched on for all required record fields, and you should also note that if you have done any schema changes in a module at the master node, then you must ensure that the required fields are marked as required across all tenants that are replicating that module.

A user is not able to view records and FortiSOAR displays a 500 error

Resolution

Any user who is created in a multi-tenant environment and who requires to view records must have a minimum of read-only access for the `Tenants` module in their respective role.

Therefore, ensure that you have assigned the user a role that has a minimum of read-only access for the `Tenants` module.

Tenant modules not getting displayed at the master node after initial configuration

You might not see any modules of a dedicated tenant node at the master node in **Tenant Manager** > **Manage Modules** after the initial configuration (addition) of the tenant node if there are errors while configuring the tenant.

Resolution

This issue occurs due to the tenant being in the "Verification Failed" state at the time of configuration. To make the MMD of the tenant visible at the master node, restart the "cyops-postman" service at the tenant node.

In case of clustering of secure message exchanges, FortiSOAR is unable to connect to primary secure message exchange even after the primary node has come back online after a failure

If you have your secure message exchanges setup as a cluster for high availability, and you face a failure, and the primary secure message exchange node comes back online and yet the FortiSOAR node is still not able to connect to the secure message exchange. You will see the following error in the postman logs:

```
2018-11-20 10:36:21,022 140437372753664 59b195ab94e35ef70e28ed129c58c804 ERROR pika.callback
callback process(): Calling <bound method BlockingChannel._on_channel_closed of <BlockingChannel
impl=<Channel number=1 CLOSED conn=<SelectConnection OPEN socket=('<xxx.xxx.xx.xxx', 53378)->
('xxx.xxx.xx.xxx', 52011) params=>>>> for "1:Channel.Close" failed
Traceback (most recent call last):
File "/opt/cyops-routing-agent/.env/lib/python3.4/site-packages/pika/callback.py", line 236, in
process
callback(*args, **keywords)
File "/opt/cyops-routing-agent/.env/lib/python3.4/site-packages/pika/adapters/blocking_
connection.py", line 1358, in _on_channel_closed
method.reply_text)
pika.exceptions.ChannelClosed: (404, "NOT_FOUND - home node 'rabbit@' of durable queue
'queue.postman.data.remoterequest.805553bfece2f6a5895c8db8c54b9ae0' in vhost 'vhost_
59b195ab94e35ef70e28ed129c58c804' is down or inaccessible")
```

Resolution

Stop and start the rabbitmq app on the secure message exchange node using the following commands:

```
# rabbitmqctl stop_app
```

```
# rabbitmqctl start_app
```

Shifting the Secure Message Exchange of tenants leads to MMDs not being pushed to the tenants that have been shifted to the new Secure Message Exchange

If you have shifted the Secure Message Exchange of any tenant in your multi-tenant configuration to a new Secure Message Exchange, then the remote mmd management does not work since the `uwsgi` service keeps the Secure Message Exchange details in memory and uses it to publish to the Secure Message Exchange. However, when the Secure Message Exchange is changed for any tenant the change gets notified using "rabbitmq publish" and the updated Secure Message Exchange setting is available only to the `postman` service; the `uwsgi` still has old Secure Message Exchange details.

Resolution

After you have shifted any tenant to a new Secure Message Exchange, then you must update the router settings in the `uwsgi` service by restarting the `postman` and `uwsgi` services on the master node using the following commands:

```
# systemctl restart uwsgi
# systemctl restart cyops-postman
```

You also need to restart the `postman` service on the tenant node.

Tenant with basic authentication is stuck in the "Awaiting Remote Node Connection" state when a user removes and re-adds certificates

If a user deletes the certificate or key file by mistake and then re-add the certificate on an 'Agent', the status of the agent changes to "Configuration Failed" and then to "Awaiting Remote Node Connection". In this case if the user configures multitenancy again by exporting the latest master configuration from master and importing the configuration on a tenant node, then the record of the tenant on master might remain stuck in the "Awaiting Remote Node Connection"; however, the master configuration on the tenant node shows that the remote node is connected.

Resolution

1. Open the Tenant node and navigate to **Setting** > **Master Configuration**.
2. Disable and again enable the master configuration. This updates the state of the tenant record on master and displays the correct "Remote Node Connected" state.

Post-upgrade to 7.3.0 the status of the Tenant displays "Remote Node Unreachable"

Once you have updated your MSSP setup to 7.3.0 or later from a release prior to 7.3.0, the status of the tenant displays "`Remote Node Unreachable`"

Resolution

Restart the `cyops-integrations-agent` service on the tenant node.

Known Issues and Workarounds

There are no significant known for MSSP environments in this release of FortiSOAR.

www.fortinet.com