# Cortex XSOAR Multi-Tenant Guide

*6.1*

paloaltonetworks.com/documentation

techDOCS paloalto NETWORKS

## Contact Information

Corporate Headquarters:
Palo Alto Networks
3000 Tannery Way
Santa Clara, CA 95054
www.paloaltonetworks.com/company/contact-support

## About the Documentation

- For the most recent version of this guide or for access to related documentation, visit the Technical Documentation portal www.paloaltonetworks.com/documentation.
- To search for a specific topic, go to our search page www.paloaltonetworks.com/documentation/document-search.html.
- Have feedback or questions for us? Leave a comment on any page in the portal, or write to us at documentation@paloaltonetworks.com.

## Copyright



## Last Revised

December 31, 2020

# Table of Contents

# *Multi-Tenant Deployments*

> Multi-Tenant Overview

# Multi-Tenant Overview

Cortex XSOAR multi-tenant deployments are designed for MSSPs that require strict data segregation between tenant accounts and the flexibility to easily share critical security data to those tenant accounts, such as known malicious/benign indicators.

Multi-tenancy also enables MSSPs to manage many tenants from a single console and to easily switch between tenant environments and from a tenant environment to the main environment, where the MSSP can get a bird's-eye, global view status, for example, all open incidents and indicators across all tenants.

## Who Should Use Multi-Tenancy?

Large enterprises that have multiple SOCs often also consider using multi-tenancy, however, in most cases we do not recommend it. The main reason is that enterprise customers expect some degree of collaboration between the SOCs, which, in many respects contradicts the product's architecture. We therefore suggest in most cases not to use XSOAR multi-tenancy for enterprises and instead use XSOAR Enterprise with proper RBAC implementation and where necessary add additional single servers (e.g., one server per SOC). There are exceptions to these recommendations and we encourage you to consult with XSOAR product managers and customer success to discuss your requirements.

## Multi-Tenancy Limitations

When using XSOAR multi-tenancy in an enterprise setting there are several limitations that can negatively impact the enterprise productivity.

- Complete isolation among tenants. This means that data is not easily shared between tenants. For example, collaborating on an incident requires extra steps (such as mirroring between tenants).
- Tenants cannot change definitions that were set by the main account (e.g., playbooks, etc.).
- Multi-tenancy architecture is more complex than XSOAR Enterprise server architecture. It requires heavier IT and computing resources. In general, server maintenance is more difficult and requires a strong IT team.
- For example, re-indexing a single tenant is a complex procedure and troubleshooting is often very complex.
- Backup and restore, as well as the DR mechanism are more complex than single server deployments.
- Since MT environments are more complex, some infrastructure features are introduced only after they are introduced in Enterprise environments.

## High Availability

For more information about High Availability, see high availability for multi-tenant deployments.

## Multi-Tenancy Architecture

Multi-tenancy architecture is based on the platform's ability to run multiple instances (processes and data) of the XSOAR server on a single server. Each deployment consists of a main server and tenant accounts. All tenant accounts can reside on the same (main) server or an MSSP can choose to run tenants on additional hosts. Tenants are deployed across hosts for several reasons.

- **Scalability**: full horizontal scalability - when a server or host is maxed out due to an excess of tenants on the same host, MSSPs can add another host on which to run tenants. MSSPs can easily move tenants between hosts.
- **Geography**: a global MSSP may have customers in multiple countries or even continents. To increase responsiveness the MSSP may choose to locate a host or multiple hosts in each location.

- **Compliance**: in some cases MSSPs are asked to run the tenant within the country where the customers' offices are located or even within the customer's internal network. For the latter XSOAR supports installing a single host running a single tenant that can be installed on the customer's premises.

## Engines

Direct access to remote networks might be blocked by network devices, such as proxies or firewalls. Cortex XSOAR Engines are installed in a remote network and act as proxies, which enable you to access those remote networks. You can install a single engine, or multiple engines, as a load-balancing group.In MSSP networks, engines are often used to enable the network connectivity between the MSSP's network and the customer's local network. Therefore, the engines are installed within the customer's networks (normally on a local VM situated either in the customer's DMZ or the security management network) and are programmed to communicate directly to the MSSP's main server. Once the communication starts a bi-directional tunnel is created between the MSSP and the customer's network allowing the MSSP to connect to the customer's relevant resources (e.g. AD, mail server, firewall management server, etc.).

# *Plan Your Multi-Tenant Deployment*

# Multi-Tenant Sizing Requirements

The exact specifications will vary depending on several factors, including the number of incidents ingested, the number of indicators in the system, playbook usage, and so on.

## Hardware Requirements

Each tenant is a standalone instance of Cortex XSOAR server and must meet the minimum sizing recommendations for a production environment.

**Table 1: Cortex XSOAR Server**

| Component | Dev Environment Minimum | Production Minimum |
|---|---|---|
| CPU | 8 CPU cores | 16 CPU cores |
| Memory | 16 GB RAM | 32 GB RAM |
| Storage | 500 GB SSD | 1 TB SSD with minimum 3k dedicated IOPS |

## Example Deployment

This example details the sizing requirements for a single host that has three tenants.

The host hardware requirements for this deployment are:

**Table 2: Example Deployment**

| Component | Calculation | Host Production Requirements |
|---|---|---|
| CPU | 3 tenants x 16 CPU cores | 48 CPU cores |
| Memory | 3 tenants x 32 GB RAM | 96 GB RAM |
| Storage | 3 tenants x 1 TB SSD with minimum 3k dedicated IOPS | 3 TB SSD with minimum 3k dedicated IOPS |

# Configure Security Settings for Multi-Tenant Deployments

These recommended security configurations are intended for deployments in which the main account and tenant accounts do not reside in the same DMZ.

**Table 3: Recommended Security Configurations**

| Key | Value | Description |
|---|---|---|
| `security.tenant.use.secret` | `true` | Generates a unique cookie session for the tenant account and main account.<br><br>*If you implement this key in a Multi-tenant high availability architecture, you must restart main host and all of the other hosts.* |
| `Tenant.AcceptAnyCertificate` | `false` | Validates the host certificate. Set to false if using a CA (certificate authority) signed certificate. Must be set to true if using a self signed certificate, or the main server cannot send requests to hosts. |
| `host.insecure` | `true` | Trusts any certificate (when host accounts exist). |

STEP 1 | In the main account, navigate to **Settings** > **About** > **Troubleshooting**.

STEP 2 | Create a separate server configuration for each of the recommended configurations.

1. Scroll to the bottom of the Server Configuration section.
2. Click **Add Server Configuration**.

# *Install Cortex XSOAR for a Multi-Tenant Deployment*

# Install Cortex XSOAR for a Multi-Tenant Deployment

*Multi-tenant deployments are only intended for MSSPs. If you are not an MSSP and want to deploy a multi-tenant environment, you must first consult with the Cortex XSOAR product management team. If you deploy a multi-tenant environment without approval from the product management team, Cortex XSOAR will not support the deployment.*

Ensure that you run all commands as the root user.

**Files and folders**

These are the files and folders created during the multi-tenant installation.

| File/Folder | Path |
|---|---|
| Binaries | **`/usr/local/demisto`** |
| Data | **`/var/lib/demisto`** |
| Logs | **`/var/log/demisto`** |
| Configuration | **`/etc/demisto.conf (this is not created if defaults are selected during installation).`** |

STEP 1 | Download the server package you received from Cortex XSOAR support.

STEP 2 | Run the **`chmod +x demistoserver-{version}.sh`** to make the server package executable.

STEP 3 | Run the **`./demistoserver-{version}.sh -- -multi-tenant`** command as root user.

## Installer Flags

**Flags that follow the -- separator**

| Flag | Type | Description |
|---|---|---|
| `-c` | N/A | (CentOS) Tells yum to run entirely from system cache, and does not download or update any headers unless it has to perform the requested action. |
| `-backup` | Boolean | Whether to back up server data when upgrading the Cortex XSOAR server. |
| `-backup-tenants` | Boolean | Whether to back up server data for tenants when upgrading the Cortex XSOAR server. Default is true. |

| Flag | Type | Description |
|---|---|---|
| `-conffile` | String | The server `.conf` file. The default location is `/etc/demisto.conf`. |
| `-db-address1` | String | The host name or IP address of the remote database. |
| `-db-any-certificate` | Boolean | Whether to trust any certificate when communicating with the database. Default is true. |
| `-db-conn-port` | String | The secure connection port for the remote database. Default is 50001. |
| `-db-only` | Boolean | Whether to set up only the Cortex XSOAR database server. Applicable for remote database deployment. Default is false. |
| `-db-port` | String | The port of the remote database. Default is 443. |
| `-db-secret` | String | The secret for the remote database. |
| `-db-use-proxy` | N/A | The proxy is used when communicating with the database. |
| `-demisto-no-gpg-check` | N/A | Tells yum to disable the GPG signature when checking for the Cortex XSOAR RPM. |
| `-distro` | String | Forces the Linux distro (such as CentOS, Debian, or Debian New). |
| `-do-not-start-server` | N/A | Prevents starting the server when the installation or upgrade is complete. |
| `-dr` | N/A | Sets up the disaster recovery server. |
| `-elasticsearch-url` | String | Elasticsearch URL addresses (comma-separated). For example, `http://test1:9200,http://test2:9200` |
| `-elasticsearch-api-key` | String | The Elasticsearch API key, which should be used in licensed versions. |
| `-elasticsearch-username` | String | The Elasticsearch username. |
| `-elasticsearch-password` | String | The Elasticsearch password. |
| `-elasticsearch-proxy` | Boolean | Whether to use a proxy when communicating with Elasticsearch. Can be **`true`** or **`false`**. Default is **`false`**. |
| `-elasticsearch-insecure` | Boolean | Whether to trust any certificate when communicating with Elasticsearch. Can be **`true`** or **`false`**. Default is **`false`**. |

| Flag | Type | Description |
|---|---|---|
| `-elasticsearch-timeout` | Integer | The amount of time (in seconds) before Elasticsearch times out. Default is 20 seconds. |
| `-external-address` | String | The external address of the instance during installation. |
| `-h` | N/A | Displays installation help. |
| `-index-entries` | Boolean | Whether to index entries. |
| `-otc` | String | The one-time configuration file. The default location is `/var/lib/demisto/otc.conf.json`. |
| `-prev-uninstall-script` | String | The path for the uninstall script. The default location is `/var/lib/dpkg/info/demistoserver.postrm`. |
| `-purge` | N/A | Removes the existing Cortex XSOAR installation. |
| `-restore-entries` | Boolean | Restores the entries index. If false, it prevents restoring the entries index. The default is **`true`**. |
| `-server-only` | N/A | (For remote database deployments) Sets up only the Cortex XSOAR server. |
| `-temp-folder` | String | Replaces the **`temp`** folder (located in **`/var/lib/demisto/temp`**) with **`temp-folder`** at installation. Useful when you cannot access the **`temp`** folder due to permission or storage issues. |
| `-tools` | Boolean | Installs the required tools. The default is **`true`**. |
| `-use-prev-uninstall-script` | N/A | (For Cortex XSOAR upgrades) The script that deletes the Cortex XSOAR user and group is not run. |
| `-y` | N/A | Answer all installer questions with y/yes, including the Cortex XSOAR EULA. |

**Flags that precede and include the -- separator**

Use the following flags to get help or information about the `./demistoserver-6.0-XXXXX.sh` file.

| Flag | Description |
|---|---|
| `--help` | Prints a message. |
| `--info` | Prints embedded information, including title, default target directory, or embedded script. |
| `--lsm` | Prints an embedded LSM entry, if one exists. |
| `--list` | Prints a list of files located in the archive. |

| Flag | Description |
| --- | --- |
| `--check` | Checks the integrity of the archive. |

Use the following flags to run the `./demistoserver-6.0-XXXXX.sh` file.

| Flag | Description |
| --- | --- |
| `--confirm` | Prompts you to confirm before running an embedded script. |
| `--quiet` | Prints only error messages. |
| `--accept` | Accepts the Cortex XSOAR license. |
| `--noexec` | Embedded scripts are not run. |
| `--keep` | The target directory is not deleted after running an embedded script. |
| `--noprogress` | Progress is hidden during decompression. |
| `--nox11` | An xterm is not spawned. |
| `--nochown` | Extracted files are not given to users. |
| `--nodiskspace` | Disk space is not checked for available space. |
| `--target dir` | Extracts directly to an absolute or relative target directory. |
| `--tar arg1[arg2...]` | Accesses the archive contents. |

# Install Cortex XSOAR for a Multi-Tenant Deployment with Elasticsearch

*Multi-tenant deployments are only intended for MSSPs. If you are not an MSSP and want to deploy a multi-tenant environment, you must first consult with the Cortex XSOAR product management team. If you deploy a multi-tenant environment without approval from the product management team, Cortex XSOAR will not support the deployment.*

Ensure you run all commands as root user.

**Files and folders**

These are the files and folders created during the multi-tenant installation.

| File/Folder | Path |
|---|---|
| Binaries | **`/usr/local/demisto`** |
| Data | **`/var/lib/demisto`** |
| Logs | **`/var/log/demisto`** |
| Configuration | **`/etc/demisto.conf (this is not created if defaults are selected during installation).`** |

STEP 1 | Download the server package you received from Cortex XSOAR support.

STEP 2 | Run the **`chmod +x demistoserver-{version}.sh`** to make the server package executable.

STEP 3 | To install the app server with Elasticsearch, run one of the following commands:

- If using username and password authentication: **`sudo ./demistoserver-X.sh -- -multi-tenant -elasticsearch-url=<elastic search url address> -elasticsearch-username=<the elasticsearch user name> -elasticsearch-password=<the elasticsearch password>`**
- If using API key authentication: **`sudo ./demistoserver-X.sh-- -multi-tenant -elasticsearch-url=<elastic search url address> -elasticsearch-api-key=<the elasticsearch API key>`**

| Flag | Type | Description |
|---|---|---|
| `-multi-tenant` | String | Indicates that the installation is for a Multi-tenant deployment. |
| `-elasticsearch-url` | String | Elasticsearch URL addresses (comma-separated).<br>For example, `http://test1:9200,http://test2:9200` |

| Flag | Type | Description |
|---|---|---|
| `-elasticsearch-api-key` | String | The Elasticsearch API key, which should be used in licensed versions.<br>**Note:** If you use this flag, you do not need to use the `-elasticsearch-username` and `-elasticsearch-password` flags. |
| `-elasticsearch-username` | String | The Elasticsearch username. This flag is used with the `-elasticsearch-password` flag.<br>**Note:** If you use this flag, you do not need to use the `-elasticsearch-url` flag. |
| `-elasticsearch-password` | String | The Elasticsearch password. This flag is used with the `-elasticsearch-username` flag.<br>**Note:** If you use this flag, you do not need to use the `-elasticsearch-url` flag. |
| `-elasticsearch-proxy` | Boolean | Whether to use a proxy when communicating with Elasticsearch. Can be **`true`** or **`false`**. Default is **`false`**. |
| `-elasticsearch-insecure` | Boolean | Whether to trust any certificate when communicating with Elasticsearch. Can be **`true`** or **`false`**. Default is **`true`**. |
| `-elasticsearch-timeout` | Integer | The amount of time (in seconds) before Elasticsearch times out. Default is 20 seconds. |

To continue with a high availability configuration, you must install an additional app server.

# Upgrade Your Multi-Tenant Deployment

You can use the same Cortex XSOAR installation file to upgrade each host machine and the main machine.

*While a host is upgrading, tenants on that host are not available. You might encounter connectivity issues and gaps in data collection.*

STEP 1 | On the host machine, run the Cortex XSOAR installation file with the **`-- -multi-tenant`** flag.

You can run the installer on hosts in parallel or sequentially.

STEP 2 | After all host machines finish the upgrade procedure, run the Cortex XSOAR installation file on the main machine with the **`---multi-tenant`** flag.

# Get the Host Installer Manually

In cases where you cannot access the Cortex XSOAR server or Cortex XSOAR platform, you can manually get the host installer.

- The installer file is located in `/var/lib/demisto/artifacts/demistohost-<version>-<build-num>.sh`.
- The installer file might be in a different location if the `libdir` or `artifacts` folder has a different path.

STEP 1 | Copy the installer file by running the following command.

```
scp <host-installer> <host-user>@<host-address>:/tmp
```

STEP 2 | Run the installation command from the host machine using SSH.

```
sudo tmp/<host-installer>
```

# Configure the Multi-Tenant Deployment

- Main Account to Tenant Communication Encryption
- Add a Host
- Add a Tenant to a Host
- Delete a Host
- Configure Live Backup
- Configure the HTTP Timeout
- Forward Server Configurations to Tenant Accounts
- Move a Tenant to a Different Host
- Stop and Start a Tenant
- Block and Unblock a Tenant
- Reindex a Tenant Database
- Install Engines on Tenants in a Multi-Tenant Deployment

# Main Account to Tenant Communication Encryption

In a multi-tenant deployment, communication is predominately from the main account to the host account, and then from the host to the tenants, unless when a host is first registered (communication from the host account to the main account).

Two-way communication should always be available between the main account and tenant account so that replies can be sent from the tenant to the main.

## Encryption

By default, requests are encrypted using TLS using a Cortex XSOAR self-signed certificate. You can replace the certificate by creating your own certificate and private key.

## Validation and authorization

Cortex XSOAR uses an internal API key so that the tenants or hosts can verify that the request originates from a main account and not from an unauthorized third party. An internal API key is used in all communications, which is kept on the main account, and is passed to the tenants or host when they created. The internal API key is created for hosts on installer creation, and for tenants when they are created.

Requests that require authorization (such as when a user wants to view incidents from the main account) the user details are passed down in requests, so the tenant can decipher and query them.

## Security

API keys are created by Cortex XSOAR. Requests are sent from an external source, which is received by Cortex XSOAR (usually a tenant) and interpreted as a request from an administrator. In multi-tenant environments, you need to consider where to create the API key.

- If created on a main account, it will propagate to all tenants, so anyone with that key can send requests to any tenant in the environment.
- If created on a tenant, you can only send requests to that tenant.

# Add a Host

It is recommended that you continually monitor your servers' CPU, memory, and storage usage. When capacity is reached, consider moving tenants to a new host server.

The host installation package installs and configures the new host, including the settings that will automatically connect the new host to your existing main server.

You can use the same package for all hosts that you want to add, unless you need to change the server's External Host Name or the default admin. In this case, you need to create a new host installer package for each host.

You access a new host with the multi-tenant default admin credentials that generates the host installation file.

The host machine must meet the Multi-Tenant Sizing Requirements.

*Run all installation commands as root user.*

STEP 1 | In Cortex XSOAR, go to **Settings** > **Account Management**.

STEP 2 | Scroll to the bottom of the page and click the **Add Host** button.

STEP 3 | When prompted, click the **Download** button to download and save the installation package.

STEP 4 | On the new host server machine, run the **`chmod +x demistohost-xxx.sh`** command to make the installer package an executable file.

STEP 5 | Run the **`sudo ./demistohost-xxx.sh`** command to install the host package.

STEP 6 | When prompted, enter the listening port used by the Cortex XSOAR server.

The default listening port is 443.

After you add a host, the host UI is accessible, but there is no content. It serves as a proxy to the tenant accounts from the main account.

**Troubleshooting**

If the new host does not appear in the Account Management page, try the following.

- Check the status of the host to make sure that the new service is up and running. Try restarting the host server, and if the issue persists, check the logs for potential errors. You can connect to the host using a browser with the Cortex XSOAR server default admin user to access logs and view status.
- Check that there are no firewalls or other network devices that might block access from the new host to the Cortex XSOAR server. You can try telnet or curl from host to the Cortex XSOAR server listening port (default is 443 default) to check if the server is accessible.
- If the host does not connect, log in to the host machine and make sure you can reach the external hostname from the main machine.

# Add a Tenant to a Host

A tenant, or account, is a single instance of Cortex XSOAR that you connect to a host. You populate tenants with users by specifying user roles. Any user who belongs to a selected role is automatically added to the account. All nested roles (both directions) must be explicitly assigned. When you create the tenant account you can specify the propagation labels, which determines which content items are eligible for syncing to the tenant.

*When adding a role, by default, the administrator with read and write permissions is added automatically.*

Each tenant machine must meet the mutli-tenant sizing requirements.

STEP 1 | In Cortex XSOAR, go to **Settings** > **Account Management** > **Accounts**.

STEP 2 | Click the **Add account** button.

STEP 3 | Enter a meaningful name for the account.

The account name cannot contain white space.

STEP 4 | Choose the host or high availability group.

STEP 5 | (Optional) Select existing propagation labels or type new propagation labels.

STEP 6 | Select at least one role to add to the account.

The administrator with read and write permissions is added to the account by default. You can change this if required.

STEP 7 | Click **Create account**.

# Delete a Host

You might want to delete a host if you have migrated the tenants to another host, or if you need to re-provision the host.

*If you delete a host, all tenants on that host are also deleted. Move tenants to another host, if needed, before proceeding.*

*For High Availability deployments, to delete an entire High Availability group, stop and purge all hosts in the group before deleting the group in Cortex XSOAR.*

STEP 1 | Stop the host:

```
service demisto stop
```

STEP 2 | Purge the installation on the host machine:

```
sudo ./demistohost-xxx.sh -- -purge
```

STEP 3 | In Cortex XSOAR, go to **Settings** > **Account Management** > **Hosts**.

STEP 4 | Select the host and click **Delete**.

STEP 5 | Enter the external host name to confirm deletion.

# Configure Live Backup

Make sure you have satisfied all prerequisites before you configure Live Backup.

- Install and configure a Cortex XSOAR multi-tenant deployment, a main server and at least one host server.
- Root access.
- Internet access.

You need to configure Live Backup for the main server and each host server.

*You must install the same Cortex XSOAR version and build on all servers.*

**Live Backup architecture**

The example provided assumes a multi-tenant architecture with four servers.

- Main server (already installed)
- Host server (already installed)
- Backup main server
- Backup host server

**Files and folders**

These are the files and folders you migrate from the main server and host server to the backup main server and backup host server, respectively.

| File/Folder | Location |
|---|---|
| Data | **`/var/lib/demisto/data`** |
| Artifacts | **`/var/lib/demisto/artifacts`** |
| Attachments | **`/var/lib/demisto/attachments`** |
| System Tools | **`/var/lib/demisto/systemTools`** |
| Tenants | **`/var/lib/demisto/tenants`** |
| Public Key | **`/usr/local/demisto/cert.key`** |
| PEM file | **`/usr/local/demisto/cert.pem`** |
| License | **`/usr/local/demisto/license`** |

STEP 1 | Install and configure Cortex XSOAR on the backup main server.

1. Run the **`./<demistoserver-xxxx.sh> -- -multi-tenant -dr -do-not-start-server`** command as root user to install Cortex XSOAR.
2. On the main server machine, set all necessary server configurations, for example, External host name, Archiving, Log Level, and so on.

3. On the main server machine go to **Settings** > **Account Management** > **+Add Host** to download a host installer file.
4. Get the host installer file by one of the following methods.
   - Download the file.
   - Retrieve the file from the `/artifacts` folder.

STEP 2 | Install and configure the host server and backup host server.

1. Copy the host installer file from the main server to the backup host server.
2. On the backup host server machine run the **`./<demistoserver-xxxx.sh> -- -dr -do-not-start-server`** command as root user to install Cortex XSOAR.
3. On the host server machine, set all necessary server configurations, for example, External host name, Archiving, Log Level, Log level, and so on.
4. On the main server machine, go to **Settings** > **Account Management** and verify that the main server recognizes the host server.

   All statuses should be green.

STEP 3 | Configure Live Backup for the main server.

1. On the main server machine, go to **Settings** > **Advanced** > **Backups** and enable the Live Backup option.
2. Enable the **Live Backup** option.
3. In the **Hostname/IP Address** field enter the hostname or IP address of the backup main server.
4. In the **Port** field enter the port of the backup main server.
5. Click **Save Live Backup configuration**.

STEP 4 | Configure Live Backup for the host server.

1. On the main server machine, go to **Settings** > **Account Management**.
2. Click **Edit** next to the host name.
3. Enable the **Live Backup** option.
4. In the **Hostname/IP Address** field enter the hostname or IP address of the backup host server.
5. In the **Port** field enter the port of the backup host server.
6. Click **Save Live Backup configuration**.

STEP 5 | When prompted, restart the servers.

STEP 6 | Shut down the main server and host server in the following order.

- Main server
- Host server

STEP 7 | Copy all relevant files and folders from the main server to the backup main server.

STEP 8 | Copy all relevant files and folders from the host server to the backup host server.

STEP 9 | Start the servers in the following order.

- Backup main server
- Backup host server
- Main server
- Host server

STEP 10 | When all servers are up, confirm that Live Backup is running successfully.

1. On the main server machine, go to **Settings** > **Advanced** > **Backups** and verify that there are no errors.
2. (Recommended Create an incident on each tenant account and verify that there are no errors.

# Configure the HTTP Timeout

You configure the HTTP timeout from the main account and the timeout that you configure is applied to all tenant accounts. If you add the server configuration on tenants, it will be ignored.

STEP 1 | In the main account, go to **Settings** > **About** > **Troubleshooting**.

STEP 2 | In the **Server Configuration** section click **Add Server Configuration**.

| Key | Value |
|---|---|
| `accounts.http.timeout` | The amount of time after which the HTTP request times out (in seconds). The default value is 30. |

# Forward Server Configurations to Tenant Accounts

Server configurations that you forward from the main account are made available on all tenant accounts. You cannot specify a subset of tenant accounts in which to make the server configurations available.

STEP 1 | In the main account, go to **Settings** > **About** > **Troubleshooting**.

STEP 2 | In the **Server Configuration** section click **Add Server Configuration**.

| Key | Value |
|---|---|
| `server.config.forward.tenants` | A comma-separated list of server configurations to forward to all tenants. For example, ***`server.config.1,server.config.2,server.config.3`*** |

STEP 3 | Click **Save**.

# Move a Tenant to a Different Host

Moving a tenant to a different host might risk data loss. Back up the tenant server before moving to a different host.

STEP 1 | In Cortex XSOAR, go to **Settings** > **Account Management**.

STEP 2 | Locate the tenant that you want to move to a different host and click the edit icon.

STEP 3 | Select the **Move to a different host** check box.

STEP 4 | Select the host to which to move the tenant.

# Stop and Start a Tenant

The following provides instructions for stopping and starting specific tenants. A use case would be if you want to reindex a single tenant.

- To stop and start a tenant via the API, use the following commands:
  - Stop: **`/account/stop/<acc_name>`**
  - Start: Use the Post command with the following body:

```
{
 "name": "<acc_name>"
 "additionalArgs": "<argsToCLI>"
 }
```

  To view the REST API documentation, select **Settings** > **INTEGRATIONS** > **API Keys** > **ViewCortex XSOAR API**.

- To start/stop a tenant via the UI:
  1. Click the cogwheel next to the tenant.
  2. To stop the tenant, click **Stop**.
  3. To start the tenant, click **Start**.

  Additional arguments can be passed to the tenant via the CLI upon start. This allows you to reindex the entire database (Reindex the Entire Database), or a specific tenant index ( Reindex a Specific Index Database), or to run other processes.

# Block and Unblock a Tenant

In the Accounts Management tab, when clicking on the cog wheel next to the tenant, you can block the tenant's account.

When you block an account, it blocks the account at proxy level (from the main server) but you can still access the tenant account via the port. You can take various actions such as edit, stop, delete, and unblock the account in the Account Management tab. The account process still runs.

A typical use case for blocking an account is when a tenant experiences an issue that affects the rest of the system. For example, if a tenant is operating at suboptimal speed and efficiency due to an indexing issue, you can block the tenant so that the rest of the system is unaffected and continues to run as expected. After you block a tenant, you can still access it via the port, fix the problem, and then unblock the tenant.

Alternatively, you can Stop and Start a Tenant the account without deleting it from the database. You cannot access the account via the port, when the account is stopped. You can take various actions such as edit, start and delete the account in the Account Management tab. If you no longer require the account, you can delete the account from the database.

# Reindex a Tenant Database

If an index is corrupted and the tenant service is not starting or if data is missing, you might need to reindex the database. Depending on the volume of data, it may take some time for the reindexing to complete. Follow these steps to reindex the entire database of a tenant.

*Reindexing a tenant requires machine resources and might have a temporary impact on the performance of other tenants.*

STEP 1 | Backup the tenant index directory, located at: **`/var/lib/demisto/tenants/acc_`*`TENANT_NAME`*`/data/demistoidx`** .

The backup of the index directory should not be stored under **`/var/lib/demisto/tenants/acc_`*`TENANT_NAME`*** .

STEP 2 | Delete the tenant index folder:

```
sudo rm -rf /var/lib/demisto/tenants/acc_ TENANT_NAME/data/demistoidx
```

STEP 3 | If the tenant is running, stop the tenant process:

Go to **Settings** > **Account Mangement** > **Accounts**, select the tenant account, and click **Stop**.

STEP 4 | Restart the tenant process:

Go to **Settings** > **Account Mangement** > **Accounts**, select the tenant account, and click **Start**.

The host will automatically reindex the tenant account.

STEP 5 | Log in to your Cortex XSOAR tenant instance and verify that the reindex process was successful.

All of your data should appear, for example, incidents, playbooks, automations, and so on. If there is a problem, contact the Cortex XSOAR support team.

# Install Engines on Tenants in a Multi-Tenant Deployment

Engines created on tenants use a different encryption handshake for each tenant and connect back to the tenant through the main host server.

STEP 1 | Configure the base URL.

1. On the tenant, go to **Settings** > **ABOUT** > **Troubleshooting**.
2. In the **Base URL (for D2 Agents and Engines)** field, enter the external tenant URL address in the following format:

   ***<main account external address>/<tenant account name>***

   For example, **demisto.com:443/acc_myaccount**

STEP 2 | Download and install the engine.

1. Go to **Settings** > **Integrations** > **Engines**.
2. Click **Create New Engine**.
3. Select and download the appropriate installer file.
4. Install the engine on the appropriate remote machine.

STEP 3 | Go to **Settings** > **Integrations** > **Engines** and verify that the engine is connected.

In many cases the Cortex XSOAR server has a firewall because the engine is probably installed in a different network. The firewall might stop any communication between the engine machine and the Cortex XSOAR server.

Ensure that the engine machine is able to communicate with the main host server. You can use Telnet, or any similar tool to check the engine has access to the main account before you install it. If there is a firewall you may need to allow access from the machine that hosts the engine, so that it can communicate back on port 443 (or any other port the main host may use) or set an ANY ANY rule.

# *Manage Content*

# Manage Content Overview

In order for a content item to be synced to a tenant account, both the content and tenant account must have the same propagation label.

For example, if you want Playbook ABC to sync to Tenant 123, they both need to have the same propagation label, such as **Premium**.

*As of version 6.0, if there is no relevant propagation tag on your content, for example, a script or playbook, but it is a dependency of a package that you do propagate to a tenant, the unlabeled content is still synced to the tenant.*

The default label for content items is **all**, which means that the content item will be synced to all tenant accounts.

## Propagation labels

There are several types of propagation labels that you can use for syncing content to tenant accounts.

- **All**: Content items with the label **all** will be synced to all tenants, whether or not the tenants have labels. This is the default label for content items.
- **Custom**: You can add custom labels by typing a label name in the **Propagation Label** field when adding or editing a content item or a tenant.
- **None**: If a content item does not have any labels, it will not be synced to any tenants. If a tenant does not have any labels, only content items with the **all** propagation label will sync to it.

*We recommend that you first apply propagation labels to your tenant accounts and then add the corresponding labels to the content items that you want to sync to the tenants.*

# Add Propagation Labels to an Existing Tenant Account

The propagation labels that you add to the tenant account determines which content items are eligible for syncing. This information is intended for existing tenant accounts. You can add propagation labels when you add a tenant to a host.

STEP 1 | In Cortex XSOAR, go to **Settings** > **Account Management**.

STEP 2 | Click the edit (pencil) icon for the tenant that you want to add propagation labels to.

STEP 3 | Select existing propagation labels or type new propagation labels.

# Add Propagation Labels to Content

You can add propagation labels when creating a new content item or when editing an existing item.

The default propagation label for all content items is **All**.

*We recommend that you first apply propagation labels to your tenant accounts and then add the corresponding labels to the content items that you want to sync to the tenants.*

The following content items support propagation labels:

- Playbooks
- Scripts
- Integrations
- Indicator fields
- Incident fields
- Evidence fields
- Indicator types
- Incident types
- Pre-process rules
- Widgets
- Dashboards
- Incident layouts
- Indicator layouts

*When installing a Content Pack from the Marketplace, the default propagation label is set to* ***all****. If you want to change the propagation label, after installation, go to the INSTALLED CONTENT PACKS tab and click the propagation button. If a content item is part of a Content Pack and it is not specifically labeled, it inherits the Content Pack's propagation labels. If labels are specified, it propagates according to those labels.*

*For a non-content item such as an integration instance, if you want to propagate the instance, you need to apply propagation labels both to the integration and to the instance. If a tenant does not have the integration installed, the instance will not be propagated even if the propagation label exists both on the main account and tenant account.*

STEP 1 | Go to the content item that you want to add a propagation label to.

STEP 2 | In the **Propagation Labels** field add the relevant labels.

- Select an existing label.
- Type a new label and press **Enter**. The label is immediately available to select in the platform.
- Keep the **All** label to sync this content item to all tenant accounts.

# Sync Content to Tenant Accounts

The content that you sync from the main account to the tenant accounts might add, override, or remove content from the tenant accounts. New content items, those which do not currently exist on the tenants will be added. When you sync content to tenant accounts, there can potentially be content items that will be added,

- **Add**: New content items, those which do not currently exist on the tenants, will be added.
- **Override**: For content items that are being pushed in the sync operation and which already exist on the tenants, the sync operation will override the existing content on the tenant accounts.
- **Remove**: For content items that were removed from the main account and which already exist on the tenant accounts, the sync operation will remove the existing content from the tenant accounts.

It is important that you review each content item and their dependencies before syncing the content. You have the option to remove items before executing the sync operation.

STEP 1 | In the main account, go to **Settings** > **Account Management**.

STEP 2 | Review all content affected by the sync operation in each of the three tabs.

STEP 3 | Select the tenant accounts you want to sync content to.

1. (Optional) To sync content to a single tenant account, click **Sync** for the tenant account.
2. (Optional) To sync content to all tenant accounts, click **Sync all**.

STEP 4 | (Optional) Remove items from the sync operation.

STEP 5 | Click **Sync**.

## Troubleshoot Errors When Syncing All Accounts

When trying to sync all accounts in a large scale environment, the sync fails. The log shows that there are too many open files and the sync fails to open any additional files.

To resolve this issue, update the limit of the number of files on the Cortex XSOAR service level as follows:

1. Open the **demisto.service** file on each host. It is located `/etc/systemd/system/demisto.service`.
2. Under **[Service]**, update the following parameters with a number that will be large enough to avoid the error. if these parameters do not exist, add them.
   - **`LimitNOFILE`**
   - **`LimitNOFILESoft`**

   For example:

```
[Unit]
Description=Demisto Server Service
After=network.target

[Service]
LimitNOFILE=100000000
LimitNOFILESoft=100000000
WorkingDirectory=/usr/local/demisto
Type=simple
User=demisto
ExecStart=/usr/local/demisto/server
EnvironmentFile=/etc/environment
```

```
Restart=always
RestartPreventExitStatus=29

[Install]
WantedBy=multi-user.target
```

3. Reload the daemon.

   **`systemctl daemon-reload`**

4. Restart the service.

   **`systemctl restart demisto`**

# Disable Propagation Labels

STEP 1 | In the main account, go to **Settings** > **About** > **Troubleshooting**.

STEP 2 | In the **Server Configuration** section click **+Add Server Configuration**.

STEP 3 | Set the **`selective.propagation.enabled`** key to **`false`**.

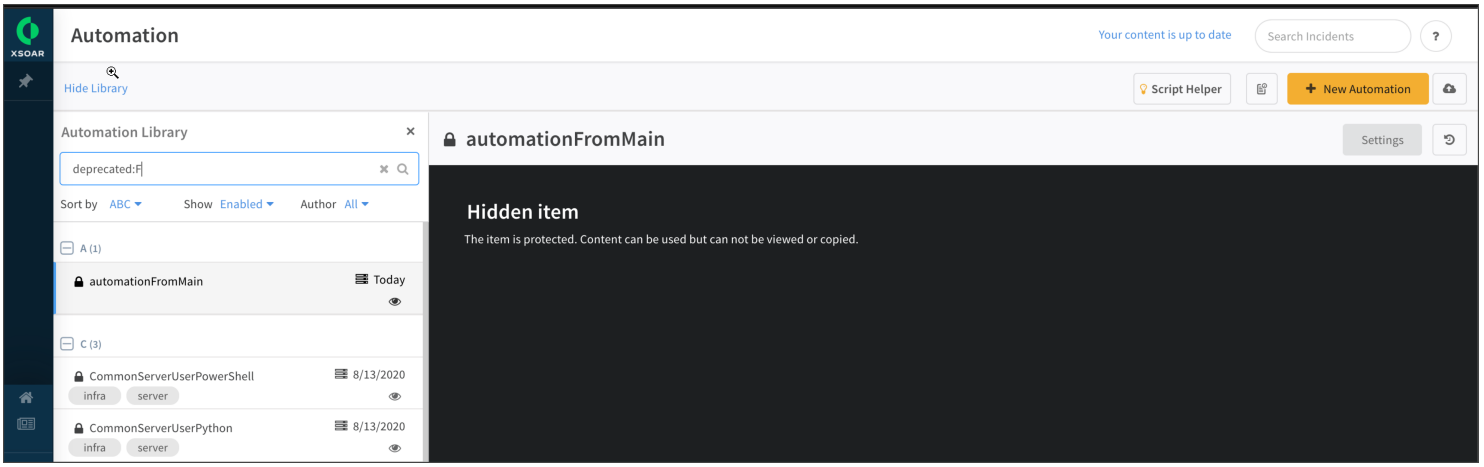
# Restrict Actions for Custom Locked Content Items

This feature is applicable in multi-tenant environments. The feature enables you to restrict actions such as downloading, cloning, and viewing code for custom locked content items on the tenant account. Users who are local to the tenant are not able to:

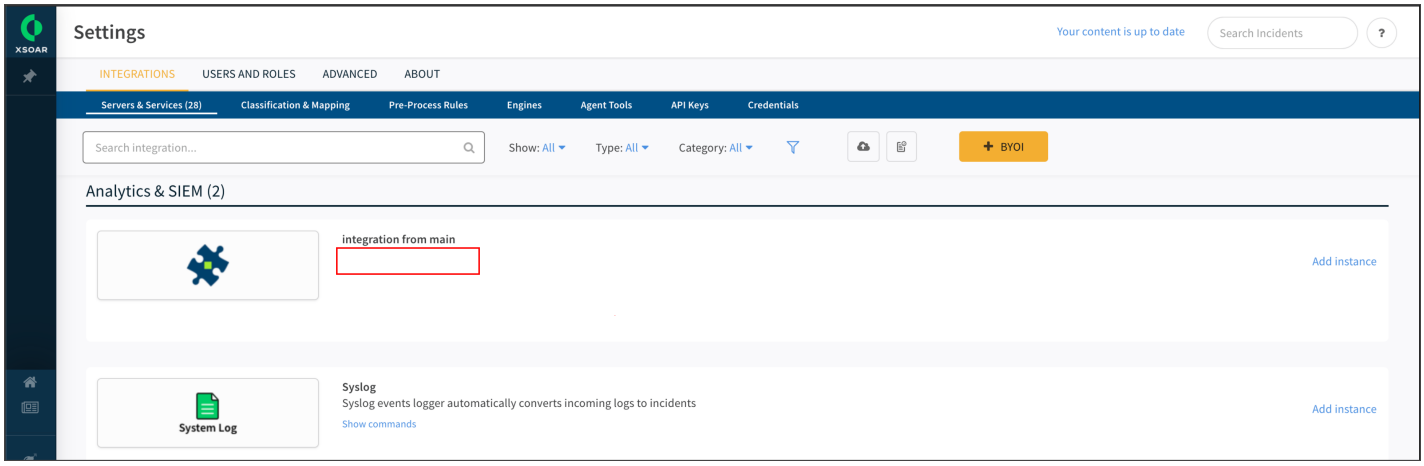
- Clone, detach, download, or view automation code.
- Clone or view integration codes.
- Download or clone playbooks.

When the feature is enabled, changes in the following pages are visible in tenant accounts:

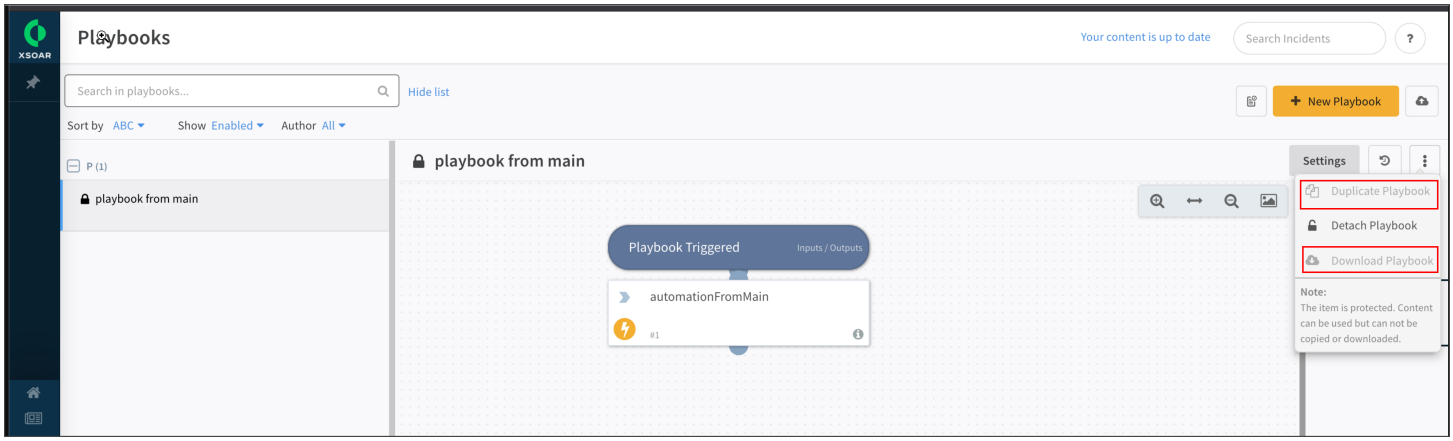
- In the Automations page, the script and all action buttons are hidden.




- In the Integrations page, the view and copy buttons are not visible for custom hidden integrations.




- In the Playbooks page, the duplicate and download action buttons are disabled for custom locked playbooks. Task scripts are not available for view if custom locked scripts are used.




*The restrictions in this feature do not apply to users of the main account who are accessing the tenant account.*

To restrict actions for custom locked content items, perform the following server configuration.

STEP 1 | Go to **Settings** > **About** > **Troubleshooting**.

STEP 2 | In the Server Configuration section, click **Add Server Configuration**.

STEP 3 | Enter the key **`server.restrict.custom.locked.content.actions`** with a value of **`true`**.

# Share Indicators

The share indicators feature requires a Cortex XSOAR Threat Intel Management license.

# Share Indicators Overview

*The share indicators feature requires a Cortex XSOAR Threat Intel Management license and that Cortex XSOAR runs using Elasticsearch.*

Each tenant account has a dedicated shared index in Elasticsearch. When you export a tenant's indicators, either manually or using the Share Indicators integration, the indicators are stored in the index. This is the index from which other tenants ingest the shared indicators.

There are two flows when sharing indicators. First, you export a tenant's local indicators to a shared index. Second, you configure the other tenants to ingest indicators from the shared indexes. There are several ways that tenant accounts ingest, or receive, indicators from the shared index.

The Share Indicators integration serves two functions in the share indicators flow.

- When configured on a tenant account, the Share Indicators integration defines which local indicators to export to the shared indicator index.
- When configured on the main account, the Share Indicators integration defines which indicators to push (share) to tenant accounts. Indicators are shared according to the propagation labels that you apply to the tenant accounts.

# Export Indicators to the Shared Index

Use the Share Indicators integration to define which indicators to export to the shared index.

STEP 1 | Access the tenant account for which to share the indicators.

STEP 2 | Go to **Settings** > **Integrations** > **Servers & Services**.

STEP 3 | Search for **`Share Indicators`**.

STEP 4 | Configure the integration instance.

| Parameter | Description | Example |
|---|---|---|
| Name | A meaningful name for the integration instance. | indicators-share_domains_ips |
| Fetch indicators | Make sure you select this option if you want this integration instance to export indicators to the shared index. | N/A |
| Fetch interval | How often to fetch indicators from this tenant and export them to the shared index. You can specify the interval in days, hours, or minutes. | 5 minutes |
| Indicators Query | The query that defines which indicators to fetch from the tenant and export to the shared index. The Query is in Elasticsearch syntax. | type:Domain or type:IP |

The tenant's indicators are exported to the shared index, at the specified interval, and are available for other tenants to ingest.

# Configure Tenants to Ingest Shared Indicators



## Ingest Indicators from the Shared Indicators Index

When you configure the Elasticsearch Feed integration to fetch indicators for a tenant, all indicators are fetched from the shared indexes. You cannot define a subset of indicators for the tenant to ingest.

STEP 1 | Access the tenant account for which to share the indicators.

STEP 2 | Go to **Settings** > **Integrations** > **Servers & Services**.

STEP 3 | Search for **`Elasticsearch Feed`**.

STEP 4 | Configure the integration instance.

| Parameter | Description | Example |
|---|---|---|
| Name | A meaningful name for the integration instance. | Elasticsearch_Feed_domains_ips |
| Fetch indicators | Make sure you select this option if you want this integration instance to export indicators to the shared index. | N/A |
| Feed Type | Predefined configuration of indexes to fetch from. For sharing indicators, it should be **`Cortex XSOAR MT Shared Feed`**. | Cortex XSOAR MT Shared Feed |
| Server URL | The URL of the Elasticsearch server. **Note**: If Elasticsearch is installed in the same machine as the Cortex XSOAR instance, the following system configuration should be added to the tenant configuration under **Settings** > **About** > **Troubleshooting**: key:**`python.pass.extra.keys`** and value: **`--network=host`**. | http://elasticsearch.<*companyA*>.com |
| Fetch interval | How often to fetch indicators from this tenant and export them to the shared index. You can specify the interval in days, hours, or minutes. | 5 minutes |
| Indicator Reputation | The reputation to apply to indicators ingested from this integration instance. | Suspicious |

| Parameter | Description | Example |
|---|---|---|
| Source Reliability | The reliability of the source providing the intelligence data, which affects how this indicator's fields and reputation are populated. | B - Usually reliable |
| Indicator Expiration Method | The method by which indicators from this instance are expired. | Never Expire |
| Bypass exclusion list | When selected, the exclusion list is ignored for indicators from this feed. This means that if an indicator from this feed is on the exclusion list, the indicator might still be added to the system. | N/A |

## Share Indicators with Tenants Using Propagation Labels

*The share indicators feature requires a Cortex XSOAR Threat Intel Management license and that Cortex XSOAR runs using Elasticsearch.*

The Share Indicators integration is a dedicated integration that you configure on the main account to share indicators with tenants. In order for a tenant to receive the shared indicators, you need to assign corresponding propagation labels to the integration instance, the integration, and the tenants.

For example, if you want the indicators from the Share Indicators integration instance A to three tenant accounts, you need to assign each the same propagation label to the following items:

- Share Indicators integration
- Share Indicators integration instance A
- Tenant 1
- Tenant 2
- Tenant 3

STEP 1 | Go to **Settings** > **Integrations** > **Servers & Services**.

STEP 2 | Search for **`Share Indicators`**.

STEP 3 | Configure the integration instance.

| Parameter | Description | Example |
|---|---|---|
| Name | A meaningful name for the integration instance. | indicators-share_domains_ips |
| Fetch indicators | Make sure you select this option if you want this integration instance to export indicators from the shared indexes to the tenant accounts with corresponding propagation labels | N/A |

| Parameter | Description | Example |
| --- | --- | --- |
| Fetch interval | How often to fetch indicators from the shared indexes and export them to the tenant accounts with corresponding propagation labels. You can specify the interval in days, hours, or minutes. | 5 minutes |
| Indicators Query | The query that defines which indicators to fetch from the tenant and export to the shared index. The Query is in Elasticsearch syntax. | type:Domain or type:IP |
| Propagation Labels | These labels define which tenants will receive the indicators fetched from this integration instance. Make sure whatever labels you apply here are also applied on the Elasticsearch Feed integration itself, and the relevant tenants. The default label is **all**, which will send indicators from this integration instance to all tenants, whether or not propagation labels are assigned to the tenant accounts. | Premium |

# *Remote Repositories for Multi-Tenant Deployments*

# Remote Repositories Overview

Cortex XSOAR supports the ability to work with separate repositories for a development environment and main account. This enables you to develop and test all of your content in one location, and when it is ready, you push the content to the main account. In your main account, you pull the content as you would all other content updates, and push content to your tenants using selective propagation.

In addition, Cortex XSOAR content updates are only delivered to the development environment. This enables you to determine which updates you want to push to the main account.

*Working with remote repositories is git-based. Any service that supports this protocol can be used, for example, GitHub, GitLab, Bitbucket, etc. In addition, on-premise repositories are also supported.*

**How it Works**

In the main account, the content appears as a content update, just like any other, and you pull the content from the remote repository into your working branch.

To work with remote repositories, you must have two separate Cortex XSOAR environments on two separate machines. The development environment is used to write the following content:

- Automations
- Playbooks
- Integrations
- Classification
- Agent tools
- Incident fields
- Indicator fields
- Evidence fields
- Incident layouts
- Incident types
- Pre-processing rules

  *If you have more than two pre-processing rules in your Local Changes queue, you must push all of those changes to the remote repository.*
- Indicator types
- Reports
- Dashboards
- Widgets

It is not possible to edit these elements on the main account.

You need to configure the remote repository feature both on your development machine and the main account. After you develop your content, if you want it to be available as part of a content update for the production environment, you must push the changes to the remote repository. If you experience issues, learn how to troubleshoot remote repositories.

# Configure a Remote Repository on a Development Machine

To work with the remote repositories, you must have at least two machines, one for development and one for your main account.

**Prerequisites**

Before configuring the remote repository, review the following list of prerequisites:

- Ensure that you have enabled the Selective Propagation feature on the development machine.

  Navigate to **Settings** > **About** > **Troubleshooting**.

  In the **Server Configuration** section click **+Add Server Configuration**.

  Set the **`selective.propagation.enabled`** key to **`True`**.
- Ensure that you have network connectivity from the Cortex XSOAR server to your repository. All communication goes through the Cortex XSOAR server so it must have access to the remote repository.

  *You cannot configure a Cortex XSOAR engine to manage communication to the remote repository.*
- When creating a repository in your remote Git platform, verify that the repository contains branches. Defining the repository in Cortex XSOAR does not create the branches.
- Before toggling the remote repository feature on or off, or changing your repository configuration, make sure to back up your existing content to your local computer:

  Navigate to **Settings** > **About** > **Troubleshooting** > **Custom Content**.

  Click **Export**.
- You need to install a standard, single-server deployment of Cortex XSOAR on your development environment.

STEP 1 | **Configure the environment**.

1. On the machine that is designated to be the development environment, navigate to **Settings** > **About** > **Troubleshooting** > **Custom Content** and click **Import**.
2. Verify that the content in both the development and production environments is synchronized before you proceed. If you do not synchronize the content, you might lose content in one of the environments.

STEP 2 | **Define the repository**.

1. Navigate to **Settings** > **Advanced** > **Content Repository**.
2. Click the **On/Off** toggle to enable the remote repositories.
3. Set the current machine as the development environment.
4. Enter the URL and connection credentials to the repository. Only SSH connections are supported.

   *If you are using a passphrase, only RSA private keys are supported.*

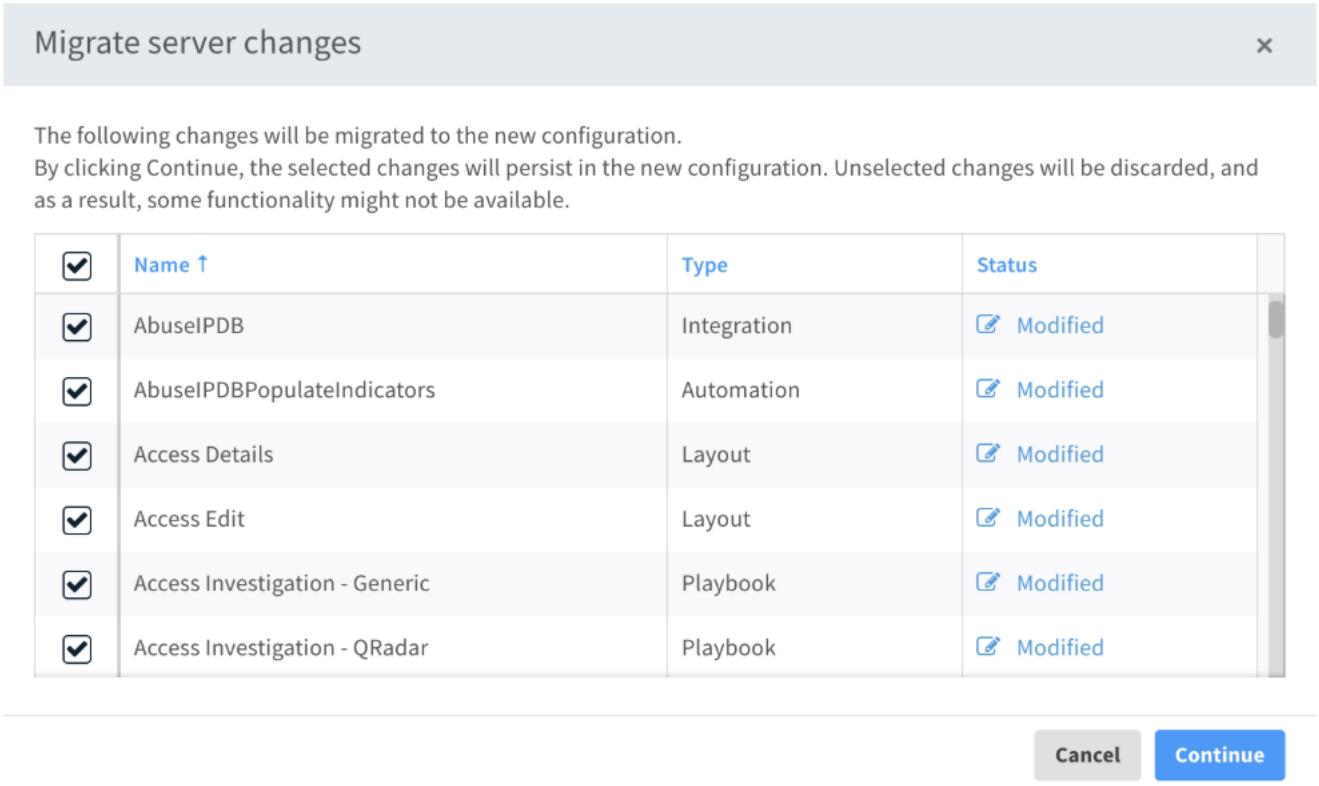
   If your SSH connection uses a port other than port 22 (the default SSH port), you must include the ssh string and port number in the url. In the following example, we use port 20017:

   **`ssh://git@content.demisto.com:20017/~/my-project.git`**

5. Select the active branch on which you will be working.
6. Click **Save**.
7. In the Migrate server changes screen, determine whether or not you want to keep the content that is currently on the development server, or discard the changes and synchronize completely with the remote repository.

   **Selected content**: the current content on your server will be maintained and presented in the Local Changes window.

   **Cleared content**: the current content on the server will be overwritten by the content in the remote repository, if it exists.




8. Click **Continue**.

Content from the remote repository is installed.

This can take several minutes depending on the amount of content in the remote repository and your hardware configuration. Your custom content is automatically backed up to the Cortex XSOAR server any time you change one of the remote repository settings. The backup is located under `/var/lib/demisto/backups/content-backup-*.tar.gz`.

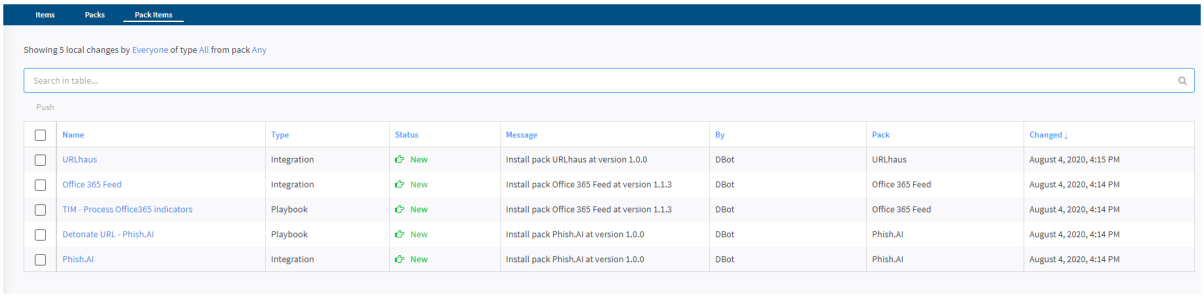
STEP 3 | **Manage Classifiers.**

1. Navigate to **Settings** > **Integrations** > **Classification and Mapping**
2. Select the classifier that you want to keep.
3. Make a small change and save the classifier.

Each integration instance has its own classification. You can only push one classification per integration. When switching the remote repository to dev mode, if you have multiple instances of the same integration, the last classification that was saved appears in the Local Changes. You should decide which classification you want to keep, and make a small change to that classification.

STEP 4 | Push Content to the Remote Repository.

1. Navigate to **Settings** > **Local Changes**
2. Select the changes that you want to push to the remote repository, and click **Push**




You should not manually push content to the remote repository. Use only the procedures outlined in the documentation to ensure that your content is properly updated in the production environment.

# Configure a Remote Repository on the Main Account

To work with the remote repositories, you must have one machine configured as your development machine, and the main account configured as your production environment.

**Prerequisites**

Before configuring the remote repository, review the following list of prerequisites:

- Ensure that the Selective Propagation feature is enabled. It should be enabled by default.
- Verify that the content that you want already exists in the remote repository. You cannot edit content in the main account. All of the content is fully synchronized with the remote repository.

*Any content that exists in the main account, but not on the remote repository, will be deleted*

STEP 1 | **Define the repository**.

1. Navigate to **Settings** > **Advanced** > **Content Repository**.
2. Click the On/Off slider to enable the remote repositories.

   The main account is automatically defined as the Production environment. This cannot be changed.
3. Enter the URL and connection credentials to the repository. Only SSH connections are supported.

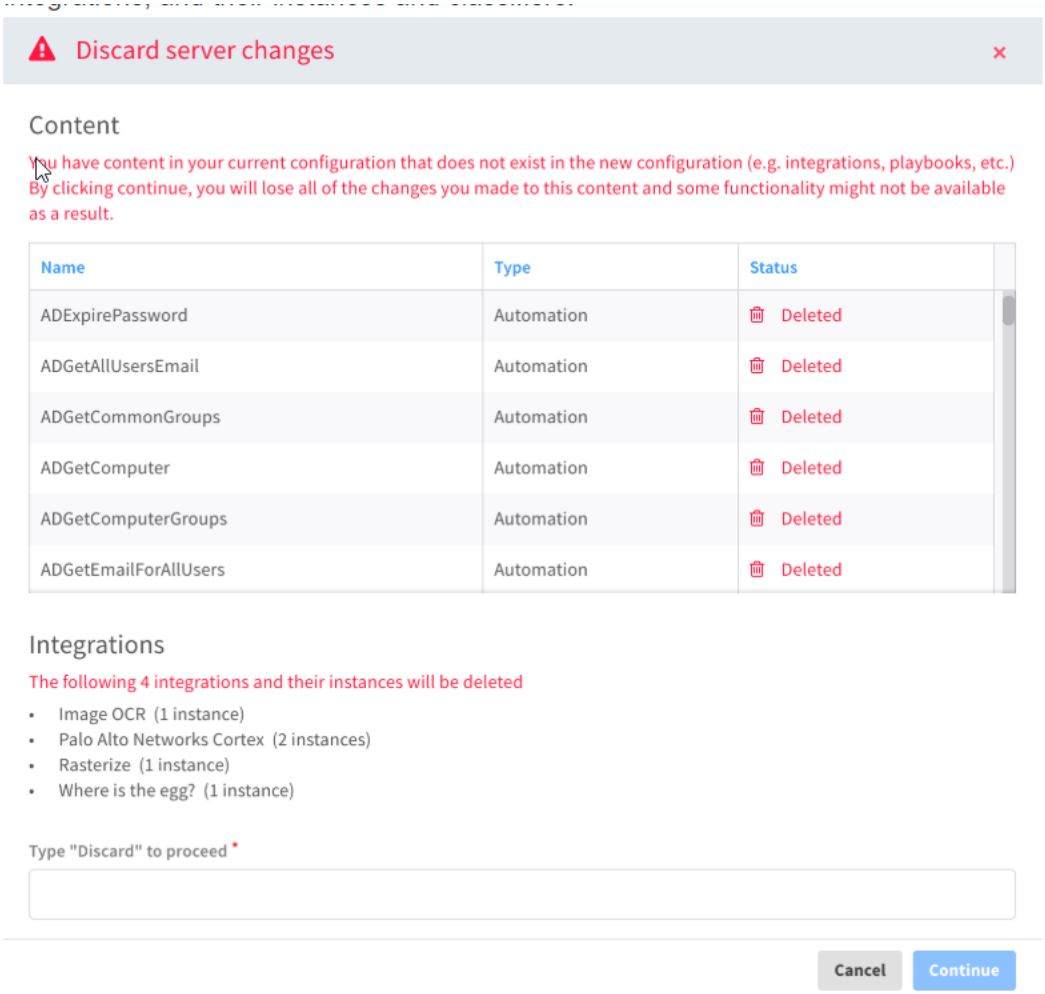
   

   *If you are using a passphrase, only RSA private keys are supported.*

   If your SSH connection uses a port other than port 22 (the default SSH port), you must include the ssh string and port number in the url. In the following example, we use port 20017:

   **`ssh://git@content.demisto.com:20017/~/my-project.git`**
4. Select the active branch from which you pull content.
5. Click **Save**.

   In the Discard server changes screen, you are presented with content that exists in your production environment, but does not exist on the remote repository. This includes integrations, and their instances and classifiers.




6. Type **`Discard`** in the relevant field and click **Continue**. All of the content that appears in this screen is discarded and permanently deleted.

   Content from the remote repository is installed. This can take several minutes depending on the amount of content in the remote repository and your hardware configuration. In addition, your custom content is automatically backed up to the Cortex XSOAR server any time you change one of the remote repository settings. The backup is located under `/var/lib/demisto/backups/content-backup-*.tar.gz`.

## STEP 2 | Pull Content from the Remote Repository

After you push content from the Development machine to the remote repository, it is available as an update for the main account.

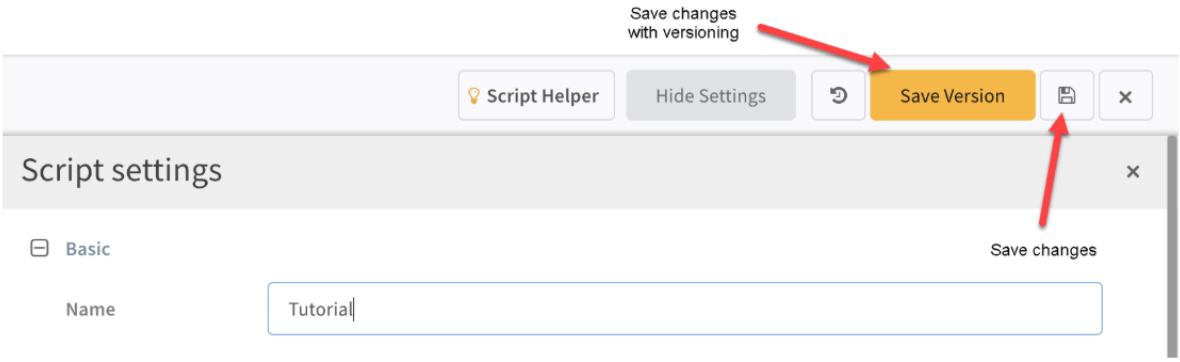
1. Check for updates.
2. Click **Install Content**.

# Edit and Push Content to a Remote Repository

Once you develop your content, for it to be available as part of a content update for the production environment, you must push the changes to the remote repository.

*You should not manually push content to the remote repository. Use only the procedures outlined in the documentation to ensure that your content is properly updated in the production environment*

You can save versions and manage revisions locally using the Save Version button. Alternatively, you can click and save the changes.




These options are only available for the following content types:

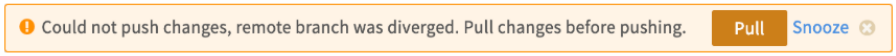
- Automations
- Playbooks
- Integrations
- Classifications
- Layouts
- Reports
- Dashboards

For all other content types, your changes are automatically saved to the local changes.

**Best Practices**

To avoid overwriting content on your local development server, Cortex XSOAR recommends that you work with a separate branch for each development environment.

If you do not work with separate branches, when trying to push content to an out-dated remote branch, you will receive a pull changes message.




Accepting this message overwrites the content on your local development server.

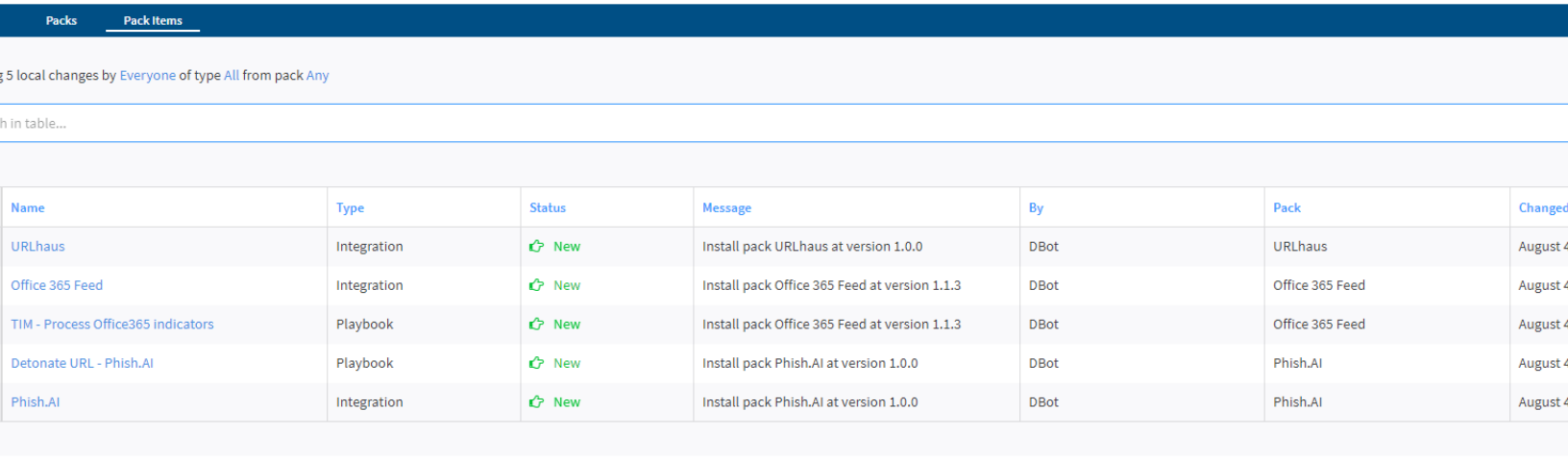
STEP 1 | Push Content to the Remote Repository.

1. Go to **Settings** > **Local Changes**. The content that you can push is shown in the tabs:

- **Items**: Content that is not related specifically to a Content Pack. For example, customized automations or playbooks.

  If you have already pushed a Content Pack and later edit one of its content items, the edited items will appear in the **Items** tab, not the **Packs** tab.
- **Packs**: All of the Content Packs that you have installed from Marketplace.

  **Note**: If you want to have more granular control over the items in the Content Packs that you push, add the **`ui.version.control.push.partial.packs`** server configuration and set the value to **`true`**.
- Pack Items - Content that is specific to the content packs you installed from Marketplace.




| Name | Type | Status | Message | By | Pack | Changed |
|---|---|---|---|---|---|---|
| URLhaus | Integration | New | Install pack URLhaus at version 1.0.0 | DBot | URLhaus | August 4 |
| Office 365 Feed | Integration | New | Install pack Office 365 Feed at version 1.1.3 | DBot | Office 365 Feed | August 4 |
| TIM - Process Office365 indicators | Playbook | New | Install pack Office 365 Feed at version 1.1.3 | DBot | Office 365 Feed | August 4 |
| Detonate URL - Phish.AI | Playbook | New | Install pack Phish.AI at version 1.0.0 | DBot | Phish.AI | August 4 |
| Phish.AI | Integration | New | Install pack Phish.AI at version 1.0.0 | DBot | Phish.AI | August 4 |

2. Select the changes that you want to push to the remote repository, and click **Push**.

   You should not manually push content to the remote repository. Use only the procedures outlined in the documentation to ensure that your content is properly updated in the production environment.

*When working with remote repositories and upgrading to version 6.0 and above, you must make sure to push your classifiers and mappers on the dev environment before upgrading the production environment.*

STEP 2 | In the dialog box add an optional message and click **Push & Exit**.

STEP 3 | Review the contents and click **Push**.

Sometimes you may not want to push all content, Content Pack dependencies, etc. For example, when a user makes a change in a playbook, which depends on an automation that another user is adding a feature, and the change does not require the new version of the automation, you can push the playbook, without the new automation.

STEP 4 | **Control Access for Pushing Content**.

1. Navigate to **Settings** > **About** > **Troubleshooting** > **Server Configuration**
2. Click **Add Configuration**.
3. Under key, enter **`UI.version.control.admin.only`**
4. Under value, enter **`true`**.
5. Click **Save**.

# Troubleshoot a Remote Repository Configuration

You can troubleshoot the following issues:

- Troubleshoot a Remote Repository Definition
- Troubleshoot Editing and Pushing Content
- Troubleshoot Content Issues

**FAQ**

**Question: Which services are supported?**

Answer: Working with remote repositories is git-based. Any service that supports this protocol can be used, for example, GitHub, GitLab, Bitbucket, etc. In addition, on-premise repositories are also supported.

**Question: Does Cortex XSOAR remote repository configuration support GPG signing?**

Answer: yes

1. Connect to the environment on which you are experiencing the issue using SSH.
2. Run the following commands:
   - **`git --global config user.signingKey KEYID`** where KEYID is your key ID.
   - **`git --global config commit.gpgSign true`**

**Question: Can I limit permissions for pushing content?**

Answer: Yes. See Edit and Push Content to a Remote Repository.

**Question: Can I edit content in my production environment?**

Answer: No. When working in a remote repository configuration, content can only be edited in the development environment.

## Troubleshoot a Remote Repository Definition

The following instructions describe how to troubleshoot when defining a remote repository.

**dial TCP Error Message**

If a **`dial tcp...`** error message appears, when defining your remote repository and loading the list of repository branches.

1. Contact your system administrator to enable connectivity. This may occur due to connectivity issues, such as a closed port or a proxy that is not enabling the connection.

**Invalid SSH URL Error**

An Invalid SSH URL error label appears, when defining your remote repository and entering the Repository URL.

1. Ensure the URL is not the HTTPS version. Currently only SSH connections are supported (and only from the server itself, no engine support).
2. Ensure the URL ends with `.git` such as `ssh://git@content.demisto.com:20017/~/my-project.git`.

**Permission Denied Error**

A permission denied error appears, when attempting to fetch the repository branches. This occurs because currently Cortex XSOAR supports only `RSA PRIVATE KEY`.

1. Ensure you have set the correct passphrase and are using the correct key type.
2. Navigate to the directory in which you saved the private key file.
3. Open the private key file in a text editor and verify that the file begins with **`-----BEGIN RSA PRIVATE KEY-----`**.

   If the file does not begin with this text, regenerate the private key and add the -m pem flag. For example,**`ssh-keygen -t rsa-b 4096 -C "your_email@example.com" -m pem`**.

**Internal Server Error**

The **`internal server error. Something went wrong.`** error message appears, when attempting to fetch the repository branches.

1. In the server log files, search for **`Host keyverification failed`** error message.
2. Connect to the server by SSH.
3. Run the following command:

   **`ssh-agent bash -c "ssh-add ~/key; git clone [url]"`**
4. Copy the file `~/.ssh/known_hosts to /var/lib/demisto/`.

# Troubleshoot Editing and Pushing Content

The following instructions describe how to troubleshoot when editing and pushing your content to the remote repository.

**Invalid Signatures When Committing**

When you push your content to the remote repository, you might receive the following error message: **`Error: Commits musthave valid signatures`**.

This occurs when your Git server configuration requires that every commit must have a signature.

1. Connect to the remote repository server using SSH.
2. Run the following commands:
   - **`git--global config user.signingKey KEYID`** where KEYID is your key ID.
   - **`git --global config commit.gpgSign true`**

**GIT Version Out-of-Date**

When you push your content to the remote repository, you might receive the following error message: **`error GIT: failedexecuting [rebase [--exec=git log --max-count=1 --pretty=format:%H;printf'>';grep -o "\w\{40\}" .git/rebase-merge/done | tail -n 1 --strategyrecursive --strategy-option theirs --onto stag --root stag_workspace]]`**

This occurs when you have an outdated Git version on your server. You must have at least version 2.21.0.

To update your Git version:

1. Export the custom content by going to **Settings** > **About** > **Troubleshooting** > **Custom content** > **Export**.

   This exports the state of the current content.
2. Turn off the remote repository feature on the dev server.
3. On your server, either download and install an updated RPM package of the Git client, or run the Demsito installer again.
4. Turn on the remote repository feature on the dev server.
5. Import the previously exported content.

   This restores the local content.

# Troubleshoot Content Issues

The following instructions describe how to troubleshoot issues with your content.

**Missing Content**

After enabling the remote repository feature or updating content, some content is missing. To resolve this issue, restore the content backup package that was created when you enabled the feature or updated content.

1. Copy the content package from the server to your local machine using the **`scp`** command.

   The backup is located at `/var/lib/demisto/backup/content-backup-*`
2. In Cortex XSOAR, navigate to **Settings** > **About** > **Troubleshooting** > **Custom content** > **Import** and upload the content package.

**Missing Content History**

When using characters that are not UTF-8, the version history is not available.

1. Connect to the environment on which you are experiencing the issue using SSH.
2. Run the following **`git config --global core.quotepath off`** command.

**Repository Connection Error**

When attempting to commit content to the remote repository, the following error appears:

```
Another git process seems to be running in this repository, e.g.an editor
 opened by 'git commit'. Please make sure all processesare terminated
 then try again. If it still fails, a git processmay have crashed in this
 repository earlier:remove the file manually to continue.) 2020-09-15
 15:34:15.9433 error VC: failed to add objects files 'automation-
GetMLModelEvaluation.yml, automation-DBotMLFetchData.yml, automation-
DBotBuildPhishingClassifier.yml, automation-DBotPreProcessTextData.yml,
 automation-SanePdfReports.yml, automation-CommonServerPowerShell.yml,
 automation-GetIndicatorsByQuery.yml, automation-GetIncidentsByQuery.yml,
 automation-SaneDocReports.yml, automation-DBotSuggestClassifierMapping.yml,
 automation-DBotTrainTextClassifierV2.yml' to index (source: /home/
circleci/.go_workspace/src/github.com/demisto/server/util/versioncontrol/
versioncontrol.go:1631)(error: exit status 128fatal: Unable to create '/var/
lib/demisto/versionControlRepo/.git/index.lock': File exists.
```

This can occur if the server was stopped during git execution.

Delete the **`index.lock`** file located in the **`/var/lib/demisto/versionControlRepo/.git/`** directory to solve the issue.