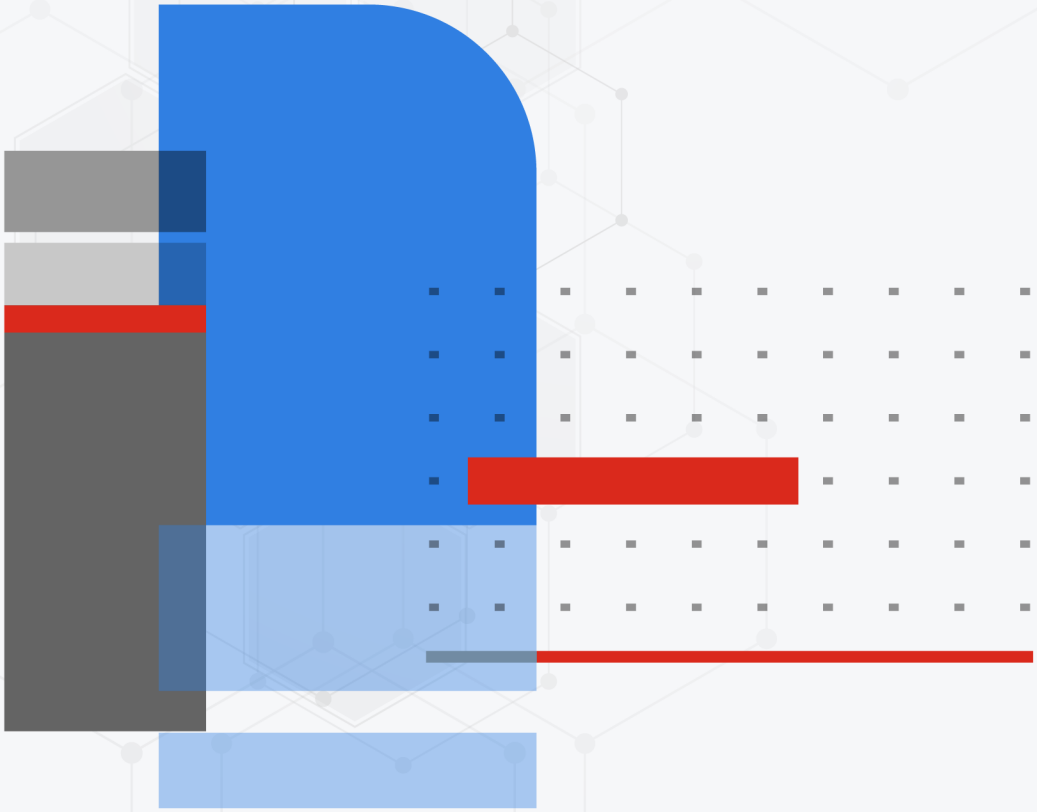
Install and Migration Guide

FortiClient EMS 7.4.6



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

March 18, 2026
FortiClient EMS 7.4.6 Install and Migration Guide
04-746-1016574-20260318

TABLE OF CONTENTS

Installing FortiClient EMS 7.4 ..... **4**
Migrating EMS 7.2.13 or 7.2.14 to 7.4.6 ..... **5**
Linux or VM 5
Docker 14
Kubernetes 20
Installation **27**
Installing EMS in standalone mode with a local DB 28
Configuring the IP address 29
Installing EMS with Postgres in Docker 32
Installing EMS with standalone remote DB without Docker 37
Deploying EMS with Docker Compose 42
Deploying EMS on Kubernetes 48
Deploying EMS as a VM image 51
VMware ESXi 52
KVM 54
Proxmox 55
Hyper-V 60
VirtualBox 61
Configuring the IP address 63
Configuring the search domain 64
Upgrading OS packages 64
Recovering FortiClient EMS VM password 65
Adding and expanding disks on EMS VMs 65
Deploying EMS in air-gapped environments 70
Air-gapped install or upgrade with EMS docker containers 70
Air-gapped install or upgrade with a dependencies bundle 71
Air-gapped install or upgrade using an HTTP proxy 73
Installation parameters **78**
Change log **80**

Installing FortiClient EMS 7.4

EMS 7.4 introduces a shift to a Linux-based model from the Windows Server-based model in earlier EMS versions. This change provides numerous benefits, including improved architecture and flexibility. This document provides instructions to migrate your EMS data from an existing Windows Server-based instance to the Linux-based model, as well as installation instructions for various use cases.

EMS 7.4 supports Ubuntu 22.04 and 24.04 Server and Desktop, Red Hat Enterprise Linux 9, and CentOS Stream 9. While EMS 7.4 supports both Ubuntu Desktop and Server, consider that Desktop uses more resources that could otherwise be available for EMS usage. You should consider proper planning for hardware resources.

Installing EMS requires an active internet connection. During installation, EMS also tries to download information about FortiClient signature updates from FortiGuard.

Because implementing or migrating to EMS 7.4 on Linux can be complex, Fortinet highly recommends the FortiClient Best Practices Service (BPS).

FortiClient BPS is an account-based annual subscription providing access to a specialized team that delivers remote guidance on deployment, upgrades, and operations. The service allows you to share information about your deployment, user requirements, resources, and other related items. Based on the information provided, BPS experts can provide recommended best practices, sample code, links to tools, and other materials or assistance to speed adoption and guide you towards best practice deployments. The team does not log into your devices to make changes. This is a consulting and guidance service which may include sample configurations or playbooks. This is not an on-site professional services offer.

You can also deploy EMS using a VM image. See Deploying EMS as a VM image on page 51.

Migrating EMS 7.2.13 or 7.2.14 to 7.4.6

As EMS 7.4 introduces a shift to a Linux-based model from the Windows Server-based model seen in earlier versions, EMS 7.4 does not support upgrade from earlier versions. You can only migrate EMS of specific 7.2 versions to 7.4. See FortiClient Upgrade Path for supported EMS migration path.

EMS 7.4.6 supports migration from 7.2.13 or 7.2.14 only. If you run a different EMS 7.2 version, upgrade your EMS to 7.2.13 or 7.2.14 first before the migration to 7.4.6.

The following provides instructions for migrating existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on different platforms:

- Linux or VM on page 5
- Docker on page 14
- Kubernetes on page 20

EMS 7.4 does not support the following legacy licenses:

- FC1-15-EMS01-297-01-DD
- FC2-15-EMS01-297-01-DD
- FC3-15-EMS01-297-01-DD
- FC4-15-EMS01-297-01-DD
- FC1-15-EMS03-297-01-DD
- FC2-15-EMS03-297-01-DD
- FC1-15-EMS03-298-01-DD
- FC2-15-EMS03-298-01-DD
- FC1-15-EMS01-299-01-DD
- FC2-15-EMS01-299-01-DD
- FC3-15-EMS01-299-01-DD

If you attempt to migrate EMS 7.2 using a legacy license to EMS 7.4.6, the migration process will be aborted with an error.

Avoid saving or deleting mobile device management (MDM) integrations while both the old and new EMS servers are running. This is necessary because both EMS servers share the same MDM service, and one server can accidentally overwrite changes made on the other.

Linux or VM

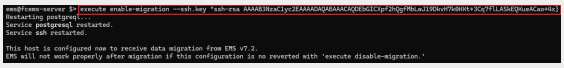
The following provides instructions for migrating existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on Linux or VM.

To enable migration and retrieve all the required information:

1. If you have not enabled or installed PowerShell on the Windows Server, follow the steps in Get started with OpenSSH for Windows to install OpenSSH.
2. Generate a public key pair in PowerShell by entering `ssh-keygen.exe -t rsa -b 4096`.
   For all subsequent prompts, press the `Enter` key. As the migration tool does not support encrypted private key files, leave the prompt empty and press `Enter` when you are prompted for a passphrase. A key pair is generated and saved to `C:\Users\Administrator\.ssh`.
3. Enable migration on the EMS Linux machine, which requires that the SSH public key transferred to the EMS Linux machine. You can transfer the key in either of the following ways:
   - **(Recommended)** Pass the RSA public key content as ASCI text directly in the command line:

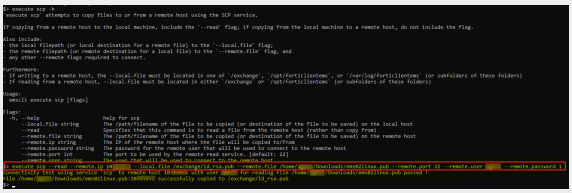
     This method is faster and more convenient but requires clipboard sharing (copying and pasting text strings) in the EMS Linux shell access.

     i. Copy the content of the public SSH key file (`id_rsa.pub` in this example).
     ii. Enable migration with the `--ssh.key` option which accepts passing the RSA public key content as ASCI text directly inside the command line. Specifically, run either of the following commands, depending on your EMS platform:

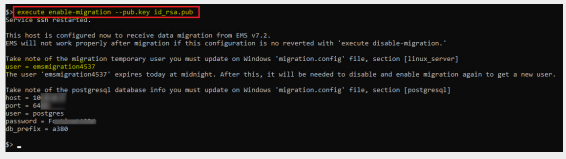
Linux	VM
`sudo emscli execute enable-migration --ssh.key="<public ssh key file content>"`	`execute enable-migration --ssh.key "<public ssh key file content>"` 

     The public SSH key file content must be wrapped in double quotations. See `execute enable-migration` for more information.
   - Copy the public key file to the EMS Linux environment using file path or filename:
     i. Transfer the public key file to the EMS Linux server home directory using either the following commands, depending on your EMS platform.

Linux	VM
Run the following commands from the Windows machine to transfer the SSH pubic key (`id_rsa.pub`) to the Linux machine. This file will be used in step 4 by a temporary ems migration user. `cat C:\Users\Administrator\.ssh/id_rsa.pub | ssh <LinuxUsername>@<ems linux ip address> 'cat >> ~/id_rsa.pub'`	Enable SSH server on the Windows Server and run the following command from the destination EMS VM appliance: `execute scp --read --remote.ip <IP> --local.file "/exchange/id_rsa.pub" --remote.file "<remote file path>" --remote.port <port number> (default is 22) --remote.user <username> --remote.password <password>`

Linux	VM
	Where the IP, remote file path, port number, username, and password should be of the remote machine where the public key file (e.g. `id_rsa.pub`) is stored. **Example:** `execute scp --read --remote.ip 10.0.0.3 --local.file "/exchange/id_rsa.pub" --remote.file "C:/Users/Administrator/.ssh/id_rsa.pub" --remote.port 22 --remote.user john --remote.password 1` The remote path can also be formatted as `C:\\Users\\Administrator\\.ssh\\id_rsa.pub\\`. If the remote machine is not Windows, the remote file path should be formatted differently. For example, `/home/user/Downloads/id_rsa.pub`. See `execute scp` for more information. 

ii. Enable migration with the `--pub.key` option to specify the public key file path or filename. Specifically, run either of the following commands, depending on your EMS platform:

Linux	VM
`sudo emscli execute enable-migration --pub.key=/home/<LinuxUsername>/id_rsa.pub` For example, if the Linux username is john, the path would be `/home/john/id_rsa.pub`. See `emscli execute enable-migration` for more information.	`execute enable-migration --pub.key <public key filename>` For *<public key filename>*, specify the filename of the public key file you copied from the remote machine in previous step (`id_rsa.pub` in this example). See `execute enable-migration` for more information. 

Migration will be automatically disabled 24 hours after `enable-migration` is executed.

4. From the CLI output, take note of the following information and save it in a handy location as you will need it when specifying migration configuration on the Windows machine in later steps:
   - Temporary user name for this machine
   - postgres information like port number, host IP, and password
   - db_prefix, if any

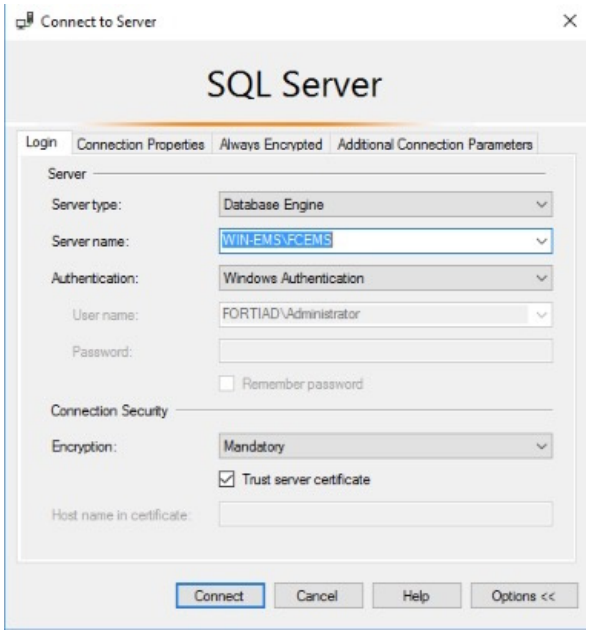
```
Take note of the migration temporary user you must update on Windows 'migration.config' file, section [linux_server]
user = emsmigration7228
The user 'emsmigration7228' expires today at midnight. After this, it will be needed to disable and enable migration again to get a new user.

Take note of the postgresql database info you must update on Windows 'migration.config' file, section [postgresql]
host = 192.168.180.131
port = 5432
user = postgres
password = ruLOJG46lkgcBU9hQVZzhCZ1WYvxWIBO
db_prefix =
```

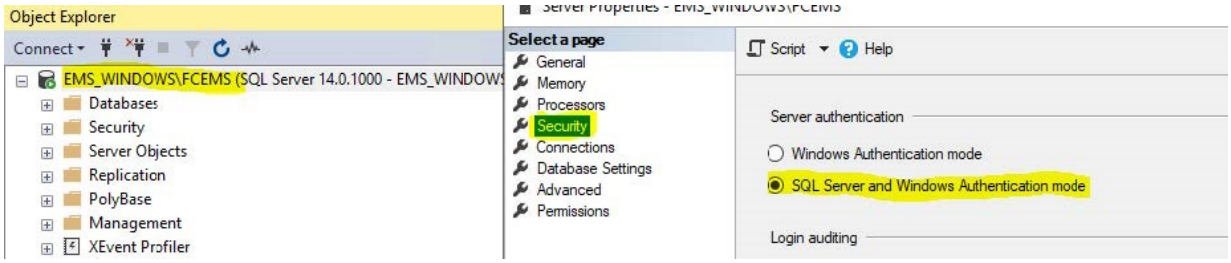
To configure the Windows Server machine with the EMS instance to migrate:

The Windows Server machine must have TLS 1.2 enabled for Client. In Registry Editor, confirm that the registry key [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client] is set to 1 or does not exist at all. Being enabled is the default behavior.

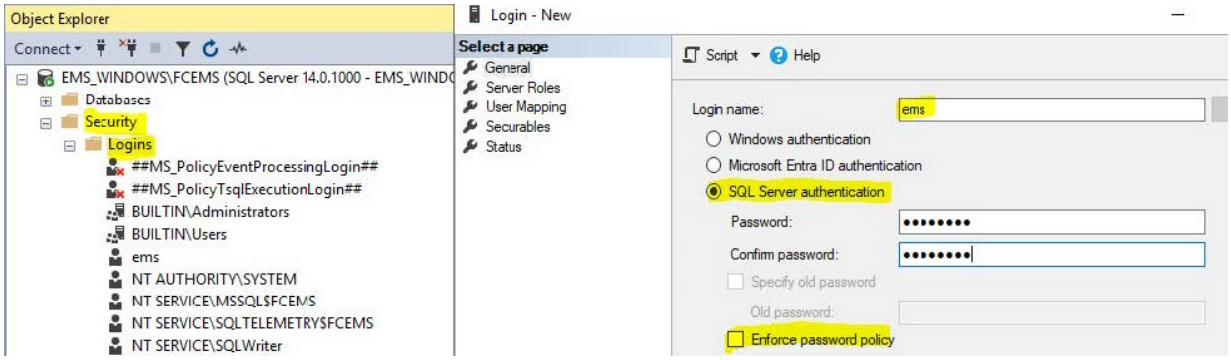
1. If the EMS has a local DB, which means the SQL server runs locally on the server, skip to step 2. Otherwise, you must create a user in SQL Server with the sysadmin role, which is required to export the tables that contain EMS data. To create the user:
   a. Log in to SQL Server Management Studio using Windows authentication. You may need to enable *Trust server certificate*.



   b. In *Object Explorer*, right-click *FCEMS* and select *Properties*. Click *Security* and select *SQL Server and Windows Authentication mode*.



c. In *Object Explorer*, go to *Security* > *Logins*. Right-click and select *New login* > *General*.
d. In the *Login name* field, enter the desired login name. In this example, the login name is ems.
e. Select *SQL Server authentication*.
f. In the *Password* and *Confirm password* fields, enter the desired password.
g. Disable *Enforce password policy*. Save.



h. Go to *Server Roles*. Select *sysadmin*. Save.
i. Restart the SQL Server (FCEMS) service. The service name may differ. Check the given name during your remote SQL install.

2. If you are migrating an EMS HA cluster, check the HA status by running `emscli ha get nodes` on any EMS node and note down the list of EMS HA nodes and the name of the primary DB node (which you will need in later steps). Go to each secondary EMS node and run `emscli service stop --all` to stop all services.
3. Download the migration tool from the Fortinet Support site and extract the files. The migration tool consists of an executable and a config file.
4. Open the config file in a text editor and specify the following parameters:

Parameter	Value to configure
[sqlserver]	
host	SQL Server IP address. For an EMS with a local DB, enter 127.0.0.1.
port	Microsoft SQL Server port.
user	User in SQL Server with sysadmin role. For an EMS with a local DB, leave this field blank.
password	Password for SQL user. For an EMS with a local DB, Leave this field blank.
[postgresql]	

Parameter	Value to configure
host	EMS Linux server IP address, which you should have retrieved in step 4 in To enable migration and retrieve all the required information: on page 6. In this example the DB and EMS will be on the same Linux server. For EMS HA clusters, enter the IP address of the primary DB node. You can see the primary DB name by running `emscli ha get nodes`) on any EMS node (see step 2).
port	Postgres port, which you should have retrieved in step 4 in To enable migration and retrieve all the required information: on page 6. For EMS HA clusters, enter the port of the primary DB node. You can see the primary DB name by running `emscli ha get nodes`) on any EMS node (see step 2).
user	Postgres default username. You should have retrieved the value in step 4 in To enable migration and retrieve all the required information: on page 6.
password	Postgres user password. You should have retrieved the value in step 4 in To enable migration and retrieve all the required information: on page 6.
db_prefix	Target database prefix. This is used when the target database has any prefix. For example, if the target database includes db1001FCM and db1001FCM_Default databases, you could enter db1001 as the `db_prefix` value. You must leave this field blank for migration to VM images downloaded from the Fortinet support portal and for default standalone installations.
[linux_server]	
host	EMS Linux server IP address. For EMS HA clusters, enter the IP address of the primary EMS node.
ssh_port	SSH port open in EMS Linux. For EMS HA clusters, enter the port of the primary EMS node.
user	EMS Linux Server user (member of sudo, forticlientems, and www-data group). You should have retrieved the value in step 4 in To enable migration and retrieve all the required information: on page 6.
key_file	Key file location in EMS Windows Server.

The following shows an example:

```
[sqlserver]
host =172.16.1.3
port =1433
user =ems
password =Test123!
[postgresql]
```

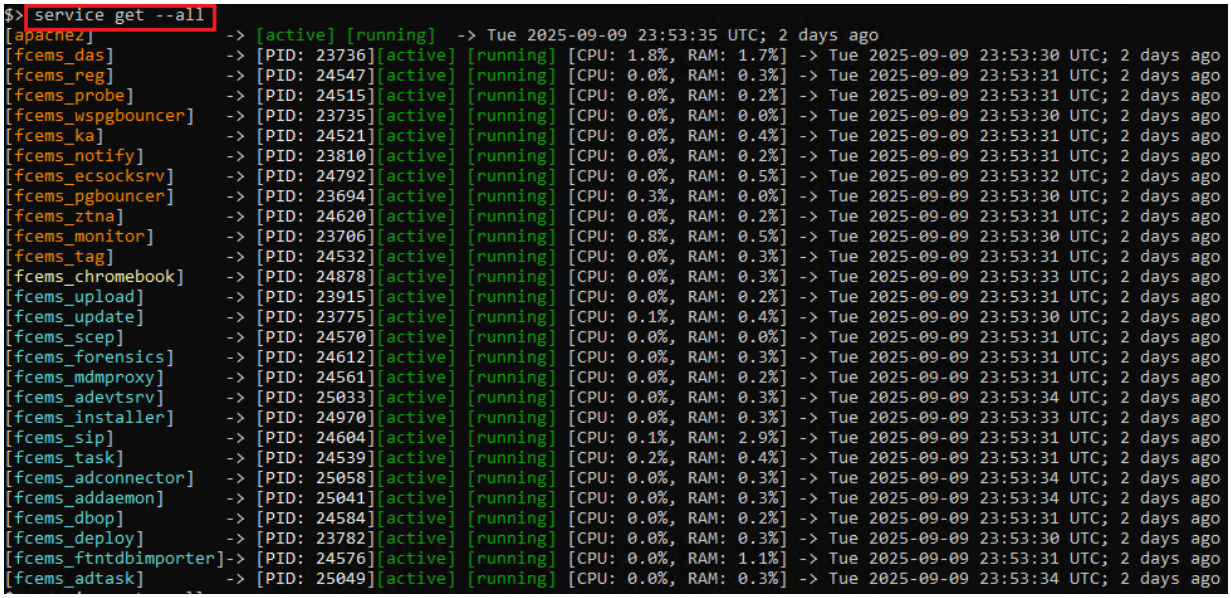
```
host =172.16.1.22
port =5432
user =postgres
password =postgres#Password from the output of the "execute enable-migration" command
db_prefix =
[linux_server]
host =172.16.1.22
ssh_port =22
user =test#Username from the output of the "execute enable-migration" command
key_file =C:\Users\Administrator\.ssh\id_rsa#location of key file in EMS windows
[files]
# Copy a single file or a directory recursively to the remote server
# follow the pattern: file_or_folder_key = {'source' : '<file_souce>', 'target' : <file_
     target>'}
# multiple entries are allowed, file_or_folder_key is just a placeholder
# Examples:
# 1 - copying the installer directory recursively:
# installer_dir =
# {'source' : 'C:\\Program Files (x86)\\Fortinet\\FortiClientEMS\\Installers',
# 'target' : '/opt/forticlientems/data'}
# 2 - copying a specific file:
# signatures_file =
# {'source' : ''C:\\Program Files (x86)\\Fortinet
     \\FortiClientEMS\\signatures\\emsaval\\emsaval.dll',
# 'target' : '/opt/forticlientems/data/signatures/emsaval/emsaval.dll'}
```

5. Open an elevated PowerShell prompt inside the EMS Windows server and go to the directory where you extracted the migration tool. Run migration.exe:

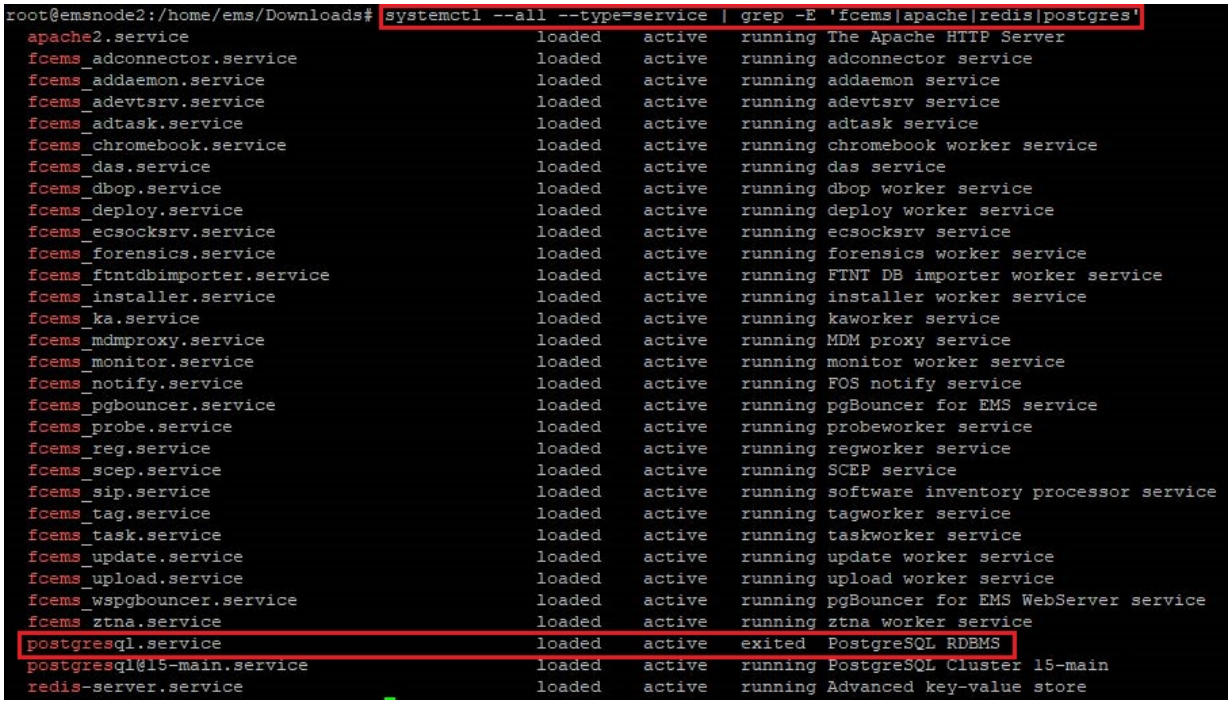
   ```
   .\migration.exe
   ```

   Wait for the migration to complete. If there are issues, check the migration log in the same folder as the migration tool.
6. For EMS HA clusters, run `emscli service start --all` on each secondary node to start all services again. Wait for a few minutes and verify the HA status by running `emscli ha get nodes` on any EMS node.
7. Check that all EMS services are running by entering either of the following commands, depending on your EMS platform:

- **VM appliance:** `service get --all`



- **Linux:** `systemctl --all --type=service | grep -E 'fcems|apache|redis|postgres'`



  The output shows that postgresql.service status displays as `exited`. This is the expected status. EMS does not create this service, which only exists to pass commands to version-specific Postgres services. It displays as part of the output as the command filters for all services that contain "postgres" in the name.

8. Access the EMS GUI on the Linux server and verify that all configurations and data from the EMS Windows Server were migrated.
9. After verifying that all configurations and data from the EMS Windows Server were migrated, disable migration using either of the following commands, depending on your EMS platform:

- **VM:** `execute disable-migration`

```
$> execute disable-migration
Service ssh restarted.
This host is no longer configured to receive data migration from EMS v7.2.
$> _
```

  See `execute disable-migration` for more information.
- **Linux:** `sudo emscli execute disable-migration`

  See `emscli execute disable-migration` for more information.

To migrate FortiClient endpoints to Linux EMS:

After EMS migration completes, do one of the following to migrate FortiClient endpoints to Linux EMS:

Method	Description
Update DNS record with Linux EMS IP address (recommended)	On the DNS server, update the DNS record for EMS with the Linux EMS IP address. FortiClient endpoints that were previously connected to Windows Server EMS resolve the new IP address to the EMS FQDN and connect to the Linux EMS. This is the recommended method to migrate FortiClient endpoints to Linux EMS. The endpoint automatically connects to Linux EMS. Even if user verification is enabled, the migration is seamless and the user does not notice any changes on the endpoint.
Switch EMS	1. On the Windows Server EMS, go to *Endpoints > All Endpoints*. 2. Select the desired endpoints. 3. Select *Action > Switch EMS > Switch by IP/Switch by Invitation*. 4. Enter the Linux EMS IP address, FQDN, or invitation. Selected endpoints connect to the Linux EMS. 5. If user verification is enabled, the user onboarding popup for the new invitation displays on the endpoint. The user must verify their identity to connect to Linux EMS.
Configure EMS server list on Windows Server EMS	This method only works if user verification is not enforced and FortiClient connects to EMS using an FQDN or IP address. 1. On the Windows Server EMS, go to *System Settings > EMS Settings*. 2. Enable *Configure EMS server list*. 3. Add the Linux EMS IP address or FQDN and port number. 4. Click *Save*. 5. After the endpoint receives the configuration changes, disconnect or shut down the EMS Windows Server. In the next keepalive interval, FortiClient connects to Linux EMS.

After migration, the license remains active on Windows Server EMS.

Docker

You can migrate existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on Docker (see Deploying EMS with Docker Compose on page 42).

To generate a public key and retrieve all the required DB information for migration:

1. If you have not enabled or installed PowerShell on the Windows Server, follow the steps in Get started with OpenSSH for Windows to install OpenSSH.
2. Generate a public key pair in PowerShell by entering `ssh-keygen.exe -t rsa -b 4096`.
   For all subsequent prompts, press the `Enter` key. As the migration tool does not support encrypted private key files, leave the prompt empty and press `Enter` when you are prompted for a passphrase. A key pair is generated and saved to `C:\Users\Administrator\.ssh`.
3. On the Linux environment where EMS docker containers are running, create a user with access to the container tools.
4. Run the following commands to grant the user membership in the required groups:

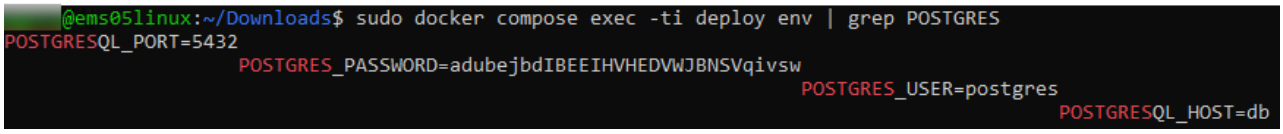
   ```
   Sudo usermod -aG docker <username>
   Sudo usermod -aG podman <username>
   Sudo usermod -aG rootlesskit <username>
   ```

5. From the Windows machine, use the following SSH commands to create the required directory and add the public key:

   ```
   ssh <linux_user>@<linux_machine_IP> mkdir -p .ssh
   cat C:\Users\administrator\.ssh\PUBLIC_KEY.pub | <linux_user>@<linux_machine_IP>  'cat >>
   ~/.ssh/authorized_keys'
   ```

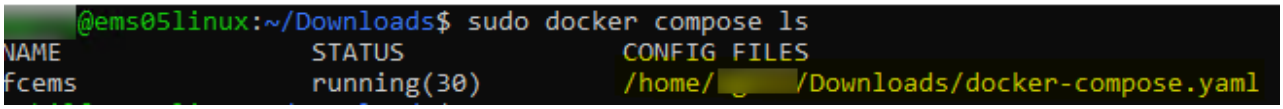
6. Run the following command to retrieve the DB information for the EMS Docker deployment:

   ```
   docker compose exec -ti deploy env | grep POSTGRES
   ```

   Save the DB information (port, host, user, password) in a handy location as you will need it in later steps.
7. Find the full path for the docker compose file by running the following command:

   ```
   Sudo docker compose ls
   ```

   Save the path in a handy location as you will need it in later steps.
8. Open the docker compose yaml file from the path you located and add ports under *services > db*.

```
services:
  db:
    image: ${REGISTRY_PATH:-}ems_postgresql15
    restart: always
    command: -c 'max_connections=1092'
    shm_size: '2gb'
    environment:
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD:-adubejbdIBEEIHVHEDVWJBNSVqivsw}
    ports:
      - 5432:5432
```

9. Recreate the container using the following command:

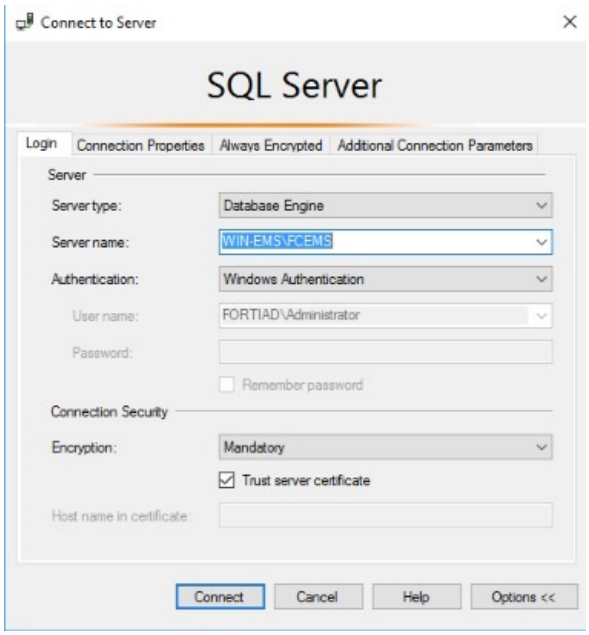
```
sudo docker compose up –d db –force-recreate
```

```
     @ems05linux:~/Downloads$ sudo docker compose up -d db --force-recreate
WARN[0000] No services to build
[+] up 3/3
 ✔Container fcems-loki-1     Running
 ✔Container fcems-promtail-1 Healthy
 ✔Container fcems-db-1       Recreated
```

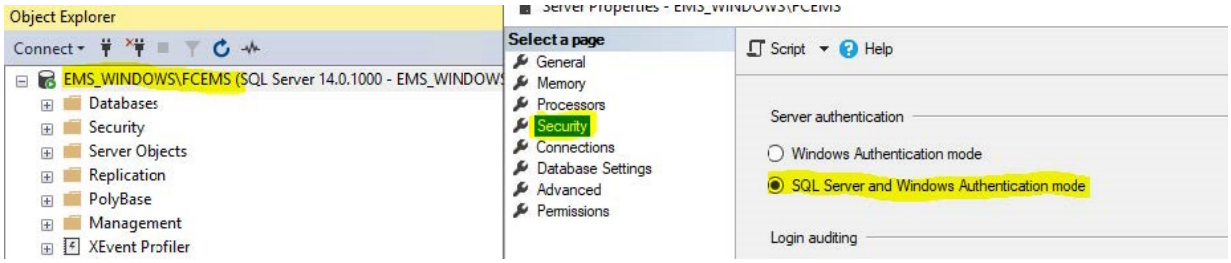
To configure the Windows Server machine with the EMS instance to migrate:

The Windows Server machine must have TLS 1.2 enabled for Client. In Registry Editor, confirm that the registry key `[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client]` is set to 1 or does not exist at all. Being enabled is the default behavior.

1. Create a user in SQL Server with the sysadmin role. You use this user to export the tables that contain EMS data:
   a. Log in to SQL Server Management Studio using Windows authentication. You may need to enable *Trust server certificate*.



b. In *Object Explorer*, right-click *FCEMS* and select *Properties*. Click *Security* and select *SQL Server and Windows Authentication mode*.



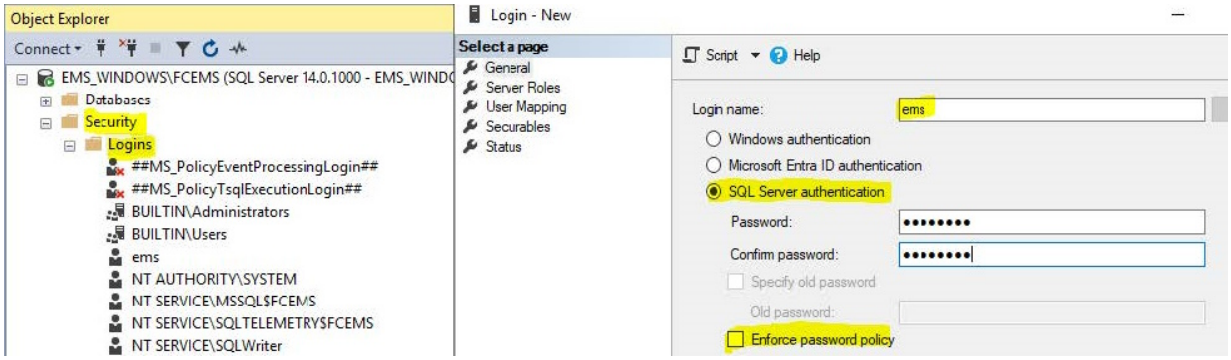
c. In *Object Explorer*, go to *Security* > *Logins*. Right-click and select *New login* > *General*.

d. In the *Login name* field, enter the desired login name. In this example, the login name is ems.

e. Select *SQL Server authentication*.

f. In the *Password* and *Confirm password* fields, enter the desired password.

g. Disable *Enforce password policy*. Save.



h. Go to *Server Roles*. Select *sysadmin*. Save.

i. Restart the SQL Server (FCEMS) service. The service name may differ. Check the given name during your remote SQL install.

2. Download the migration tool from the Fortinet Support site and extract the files. The migration tool consists of an executable and a config file.
3. Open the config file in a text editor and specify the following parameters:

Parameter	Value to configure
[sqlserver]	
host	SQL Server IP address. If you are using a local database (DB), enter 127.0.0.1.
port	Microsoft SQL Server port.
user	User in SQL Server with sysadmin role. For an EMS with a local DB, you can leave this field blank.
password	Password for SQL user. For an EMS with a local DB, you can leave this field blank.
[postgresql]	
host	EMS Linux server IP address, which you should have retrieved in step 6 in To generate a public key and retrieve all the required DB information for migration: on page 14). In this example the DB and EMS will be on the same Linux server.
port	Postgres port, which you should have retrieved in step 6 in To generate a public key and retrieve all the required DB information for migration: on page 14.
user	Postgres username, which you should have retrieved in step 6 in To generate a public key and retrieve all the required DB information for migration: on page 14
password	Postgres user password. You should have retrieved the value in step 6 in To generate a public key and retrieve all the required DB information for migration: on page 14
db_prefix	Target database prefix. This is used when the target database has any prefix. For example, if the target database includes db1001FCM and db1001FCM_Default databases, you could enter db1001 as the db_prefix value. Leave this field blank.
[linux_server]	
host	IP address of the host where docker is running.
ssh_port	SSH port open in EMS Linux, which is 22.
user	EMS Linux Server user, which you created in step 3 in To generate a public key and retrieve all the required DB information for migration: on page 14.
key_file	Key file location in EMS Windows Server. For example, C:\Users\Administrator\.ssh\id_rsa.

Parameter	Value to configure
docker_compose	Full path to the docker_compose yaml file, which you created in step 7 in To generate a public key and retrieve all the required DB information for migration: on page 14.

The following shows an example:

```
[sqlserver]
host =10.0.0.7
port =1433
user =
password =fct  @123
[postgresql]
host =10.0.0.20
port =5432
user =postgres
password =
db_prefix =
[linux_server]
host =10.0.0.20
ssh_port =22
user =
key_file = C:\Users\Administrator\.ssh\rsa_key
docker_compose = '/home/aghil/Downloads/docker-compose.yaml'
[site_migration]
# Update this section only if multiple source single-tenant EMS instances need to be migrated into a single EMS Linux instance.
# To enable site migration, set 'enable = true' and specify the target site name on the destination EMS where the site should be migrated.
enable = false
target_site =
# if running EMS on docker compose, use the `docker_compose` key below to specify the full qualified path to the docker compose file
# the location of the file can be obtained by running `docker compose ls` on the host where the docker compose is running
# if instead of pure docker, you're using podman, `docker_compose` is still required but the podman flag below needs to be set to 1
# podman = 0
# if running EMS on K8S, use the `k8s_namespace` and `k8s_pod` below to specify the k8s namespace and the pod where the `deploy` service
# is running with EMS. For single pod deployments, the pod name will always start with `ems-aio`. For multi pod deployments, the pod name
# starts with `daemons-low`.
# k8s_namespace = ems
# k8s_pod = ems-aio-86495ff465-dffxc
[files]
# Copy a single file or a directory recursively to the remote server
# follow the pattern: file_or_folder_key = {'source' : '<file_souce>', 'target' : <file_target>'}
# multiple entries are allowed, file_or_folder_key is just a placeholder
```

4. Open an elevated PowerShell prompt inside the EMS Windows server and go to the directory where you extracted the migration tool. Run `migration.exe`:

```
.\migration.exe
```

Wait for the migration to complete. If there are issues, check the migration log in the same folder as the migration tool.

5. Check that all EMS services are running using the following command:

```
docker compose exec -ti deploy emscli service get –all
```

```
root@ems:/home/ems/Downloads# docker compose exec -ti deploy emscli service get --all
[fcems_ztna]          -> [PID: 000000][active] [running] [CPU: 0.5%, RAM: 0.3%] -> 2025-12-11 20:09:35 +0000 UTC -> 1293h11m58.874753938s ago
[fcems_pgbouncer]     -> [PID: 000000][active] [running] [CPU: 3.9%, RAM: 0.1%] -> 2025-12-11 20:08:50 +0000 UTC -> 1293h12m43.874863939s ago
[fcems_das]           -> [PID: 000000][active] [running] [CPU: 18.0%, RAM: 2.0%] -> 2025-12-11 20:09:33 +0000 UTC -> 1293h12m0.875008839s ago
[fcems_probe]         -> [PID: 000000][active] [running] [CPU: 0.5%, RAM: 48.5%] -> 2025-12-11 20:09:41 +0000 UTC -> 1293h11m52.875120139s ago
[fcems_reg]           -> [PID: 000000][active] [running] [CPU: 0.8%, RAM: 0.4%] -> 2025-12-11 20:09:39 +0000 UTC -> 1293h11m54.87522924s ago
[fcems_monitor]       -> [PID: 000000][active] [running] [CPU: 5.5%, RAM: 1.1%] -> 2025-12-11 20:09:35 +0000 UTC -> 1293h11m58.87536984s ago
[fcems_wspgbouncer]   -> [PID: 000000][active] [running] [CPU: 0.4%, RAM: 0.1%] -> 2025-12-11 20:08:50 +0000 UTC -> 1293h12m43.87548264s ago
[fcems_ecsocksrv]     -> [PID: 000000][active] [running] [CPU: 0.5%, RAM: 0.5%] -> 2026-02-02 22:53:37 +0000 UTC -> 18h27m56.875715441s ago
[fcems_ka]            -> [PID: 000000][active] [running] [CPU: 0.5%, RAM: 0.4%] -> 2025-12-11 20:09:39 +0000 UTC -> 1293h11m54.875869941s ago
[apache2]             -> [PID: 000000][active] [running] [CPU: 1.7%, RAM: 4.6%] -> 2026-02-02 22:53:37 +0000 UTC -> 18h27m56.876043942s ago
[fcems_notify]        -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 0.6%] -> 2025-12-11 20:09:35 +0000 UTC -> 1293h11m58.876222342s ago
[fcems_tag]           -> [PID: 000000][active] [running] [CPU: 0.6%, RAM: 0.3%] -> 2025-12-11 20:09:40 +0000 UTC -> 1293h11m53.876383643s ago
[fcems_chromebook]    -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 0.3%] -> 2026-02-02 22:53:37 +0000 UTC -> 18h27m56.876509943s ago
[fcems_mdmproxy]      -> [PID: 000000][active] [running] [CPU: 0.4%, RAM: 43.5%] -> 2025-12-11 20:09:40 +0000 UTC -> 1293h11m53.876693144s ago
[fcems_dbop]          -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 8.0%] -> 2025-12-11 20:09:37 +0000 UTC -> 1293h11m56.876858144s ago
[fcems_event]         -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 0.1%] -> 2025-12-11 20:09:37 +0000 UTC -> 1293h11m56.876981544s ago
[fcems_scep]          -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 15.6%] -> 2025-12-11 20:09:41 +0000 UTC -> 1293h11m52.877139645s ago
[fcems_upload]        -> [PID: 000000][active] [running] [CPU: 0.4%, RAM: 0.3%] -> 2025-12-11 20:09:41 +0000 UTC -> 1293h11m52.877234745s ago
[fcems_task]          -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.877389745s ago
[fcems_adconnector]   -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 0.7%] -> 2026-02-02 22:53:37 +0000 UTC -> 18h27m56.877537746s ago
[fcems_installer]     -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.877760746s ago
[fcems_sip]           -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.877938047s ago
[fcems_addaemon]      -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.878034947s ago
[fcems_forensics]     -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 25.1%] -> 2025-12-11 20:09:38 +0000 UTC -> 1293h11m55.878164147s ago
[fcems_deploy]        -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.878249848s ago
[fcems_adevtsrv]      -> [PID: 000000][active] [running] [CPU: 0.3%, RAM: 0.3%] -> 2026-02-02 22:53:36 +0000 UTC -> 18h27m57.878335748s ago
[fcems_update]        -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.878417048s ago
[fcems_adtask]        -> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.878580849s ago
[fcems_ftntdbimporter]-> [PID: 000000][active] [running] [CPU: 0.7%, RAM: 24.5%] -> 2026-02-02 22:53:35 +0000 UTC -> 18h27m58.878701349s ago
```

6. Access the EMS GUI on the Linux server and verify that all configurations and data from the EMS Windows Server were migrated.

To migrate FortiClient endpoints to Linux EMS:

After EMS migration completes, do one of the following to migrate FortiClient endpoints to Linux EMS:

Method	Description
Update DNS record with Linux EMS IP address (recommended)	On the DNS server, update the DNS record for EMS with the Linux EMS IP address. FortiClient endpoints that were previously connected to Windows Server EMS resolve the new IP address to the EMS FQDN and connect to the Linux EMS. This is the recommended method to migrate FortiClient endpoints to Linux EMS. The endpoint automatically connects to Linux EMS. Even if user verification is enabled, the migration is seamless and the user does not notice any changes on the endpoint.
Switch EMS	1. On the Windows Server EMS, go to *Endpoints* > *All Endpoints*. 2. Select the desired endpoints. 3. Select *Action* > *Switch EMS* > *Switch by IP/Switch by Invitation*. 4. Enter the Linux EMS IP address, FQDN, or invitation. Selected endpoints connect to the Linux EMS. 5. If user verification is enabled, the user onboarding popup for the new invitation displays on the endpoint. The user must verify their identity to connect to Linux EMS.
Configure EMS server list on Windows Server EMS	This method only works if user verification is not enforced and FortiClient connects to EMS using an FQDN or IP address. 1. On the Windows Server EMS, go to *System Settings* > *EMS Settings*. 2. Enable *Configure EMS server list*. 3. Add the Linux EMS IP address or FQDN and port number. 4. Click *Save*. 5. After the endpoint receives the configuration changes, disconnect or

Method	Description
	shut down the EMS Windows Server. In the next keepalive interval, FortiClient connects to Linux EMS.

After migration, the license remains active on Windows Server EMS.

Kubernetes

You can migrate existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on Kubernetes (see Deploying EMS on Kubernetes on page 48).

To generate a public key and retrieve all the required DB information for migration:

1. If you have not enabled or installed PowerShell on the Windows Server, follow the steps in Get started with OpenSSH for Windows to install OpenSSH.
2. Generate a public key pair in PowerShell by entering `ssh-keygen.exe -t rsa -b 4096`.
   For all subsequent prompts, press the `Enter` key. As the migration tool does not support encrypted private key files, leave the prompt empty and press `Enter` when you are prompted for a passphrase. A key pair is generated and saved to `C:\Users\Administrator\.ssh`.
3. Copy the contents of the public SSH key file (`C:\Users\Administrator\.ssh\id_rsa.pub` in this example) and save it in a handy location as it will be used in later steps.
4. On the Linux environment where EMS containers are running, create a user with access to the container tools:

```
sudo useradd -m -s /bin/bash <username>
sudo usermod -aG microk8s <username>
```

5. Create script to map `kubectl` to `microk8s.kubectl`:

```
sudo cat <<'EOF' | sudo tee /usr/local/bin/kubectl > /dev/null
#!/bin/bash
exec microk8s kubectl "$@"
EOF
```

6. 4. Add execution permission to the script file:

```
sudo chmod +x /usr/local/bin/kubectl
```

7. Paste the EMS windows SSH public key (that you copied in step 3) to the authorized keys of the user in `/home/<username>/.ssh/authorized_keys`.

If the `/home/<username>/.ssh` folder does not exist yet, create it using the following commands:

```
sudo mkdir /home/<username>/.ssh && sudo chown <username>:<username>
/home/<username>/.ssh && sudo chmod 700 /home/<username>/.ssh
```

8. Create a new yaml file (eg: `db_access.yaml`) with following content:

```
apiVersion: v1
kind: Service
metadata:
name: db-mig
annotations:
metallb.universe.tf/address-pool: ""
metallb.universe.tf/loadBalancerIPs: ""
spec:
type: LoadBalancer
selector:
app: db
ports:
- name: db
protocol: TCP
port: 5432
targetPort: 5432
```

9. Run the following command to allow remote access for EMS DB during migration process:

```
kubectl apply -f <path to yaml file> -n <your namespace>
```

   To retrieve the namespace, run `kubectl get namespace`.

10. Run the following command to retrieve the DB password and save it in a handy location as you will need it when specifying migration configuration on the Windows machine in later steps:

```
kmicrok8s.kubectl exec -ti <pod> -n <namespace> -- env | grep POSTGRES
```

    To retrieve the pod name, run `kubectl get pods -n <namespace>`.

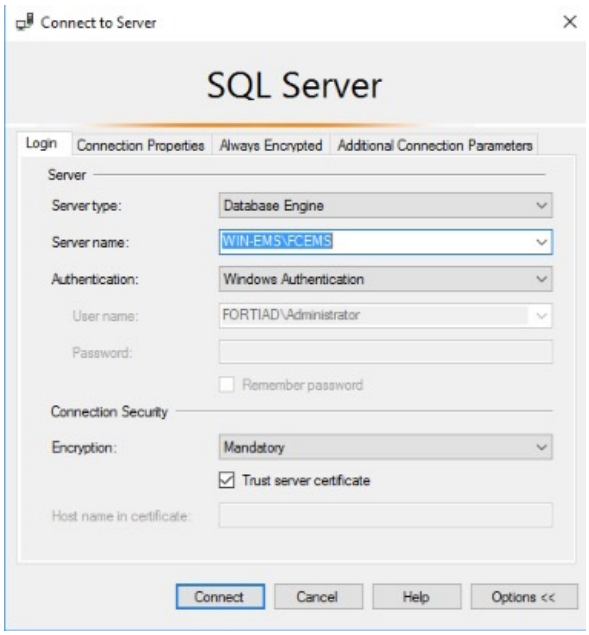
11. Run the following command to retrieve the DB IP address and save it in a handy location as you will need it when specifying migration configuration on the Windows machine in later steps:

```
kmicrok8s.kubectl get svc -n <namespace>
```

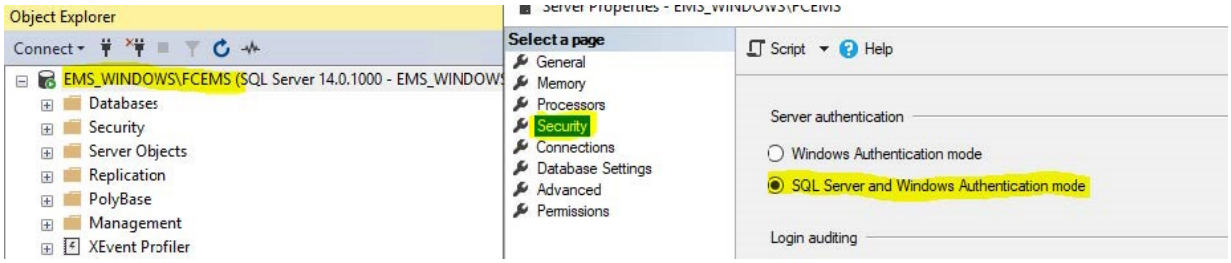
To configure the Windows Server machine with the EMS instance to migrate:

The Windows Server machine must have TLS 1.2 enabled for Client. In Registry Editor, confirm that the registry key `[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client]` is set to 1 or does not exist at all. Being enabled is the default behavior.

1. Create a user in SQL Server with the sysadmin role. You use this user to export the tables that contain EMS data:
   a. Log in to SQL Server Management Studio using Windows authentication. You may need to enable *Trust server certificate*.



b. In *Object Explorer*, right-click *FCEMS* and select *Properties*. Click *Security* and select *SQL Server and Windows Authentication mode*.



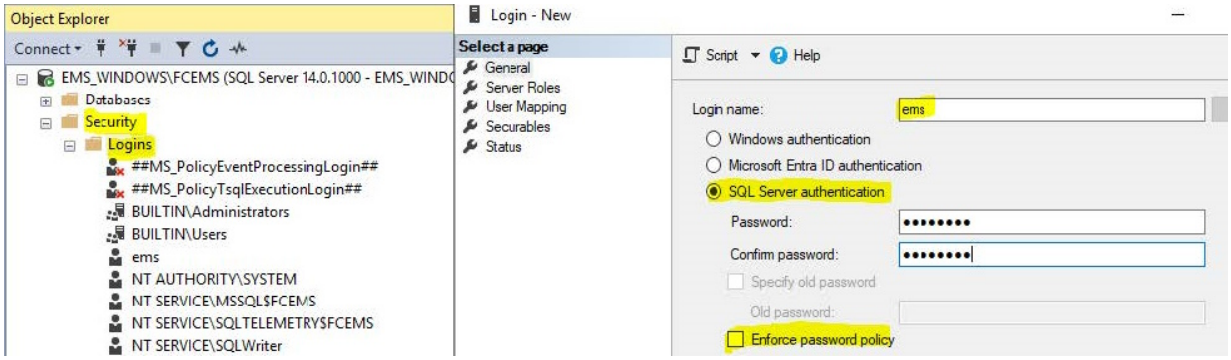
c. In *Object Explorer*, go to *Security* > *Logins*. Right-click and select *New login* > *General*.

d. In the *Login name* field, enter the desired login name. In this example, the login name is ems.

e. Select *SQL Server authentication*.

f. In the *Password* and *Confirm password* fields, enter the desired password.

g. Disable *Enforce password policy*. Save.



h. Go to *Server Roles*. Select *sysadmin*. Save.

i. Restart the SQL Server (FCEMS) service. The service name may differ. Check the given name during your remote SQL install.

2. Download the migration tool from the Fortinet Support site and extract the files. The migration tool consists of an executable and a config file.
3. Open the config file in a text editor and specify the following parameters:

Parameter	Value to configure
[sqlserver]	
host	SQL Server IP address. If you are using a local database (DB), enter 127.0.0.1.
port	Microsoft SQL Server port.
user	User in SQL Server with sysadmin role. For an EMS with a local DB, you can leave this field blank.
password	Password for SQL user. For an EMS with a local DB, you can leave this field blank.
[postgresql]	
host	EMS Linux server IP address, which you should have retrieved in step 11 in To generate a public key and retrieve all the required DB information for migration: on page 20. In this example the DB and EMS will be on the same Linux server.
port	Postgres port, which is 5432 as defined in the yaml file.
user	Postgres default username, which is postgres.
password	Postgres user password, which you should have retrieved in step 10 in To generate a public key and retrieve all the required DB information for migration: on page 20.
db_prefix	Target database prefix. This is used when the target database has any prefix. For example, if the target database includes db1001FCM and db1001FCM_Default databases, you could enter db1001 as the db_prefix value. Leave this field blank.
[linux_server]	
host	IP address of the host where docker is running.
ssh_port	SSH port open in EMS Linux, which is 22.
user	EMS Linux Server user, which you created in step 4 in To generate a public key and retrieve all the required DB information for migration: on page 20.
key_file	Key file location in EMS Windows Server. For example, C:\Users\Administrator\.ssh\id_rsa.
[site_migration]	
k8s_namespace	Kubernetes namespace where EMS has been deployed.
k8s_pod	Name of the pod that runs the deploy container:

Parameter	Value to configure
	• When EMS is deployed on Kubernetes in singlepod mode (default), the pod with the prefix `ems-aio` is used. • When deployed in multipod mode, the pod with the prefix `daemons-low` is used.

The following shows an example:

```
[sqlserver]
host =172.16.1.3
port =1433
user =ems
password =Test123!
[postgresql]
host =172.16.1.22
port =5432
user =postgres
password =Test123!
db_prefix =
[linux_server]
host =172.16.1.22
ssh_port =22
user =test
key_file =C:\Users\Administrator\.ssh\id_rsa
k8s_namespace = ems
k8s_pod = ems-aio-86495ff465-dffxc
[site_migration]
# Update this section only if multiple source single-tenant EMS instances need to be migrated
      into a single EMS Linux instance.
# To enable site migration, set 'enable = true' and specify the target site name on the
      destination EMS where the site should be migrated.
enable = false
target_site =
# if running EMS on docker compose, use the `docker_compose` key below to specify the full
      qualified path to the docker compose file
# the location of the file can be obtained by running `docker compose ls` on the host where the
      docker compose is running
# docker_compose = '/home/user/docker-compose.yaml'
# if instead of pure docker, you're using podman, `docker_compose` is still required but the
      podman flag below needs to be set to 1
# podman = 0
# if running EMS on K8S, use the `k8s_namespace` and `k8s_pod` below to specify the k8s
      namespace and the pod where the `deploy` service
# is running with EMS. For single pod deployments, the pod name will always start with `ems-
      aio`. For multi pod deployments, the pod name
# starts with `daemons-low`.
# k8s_namespace = ems
# k8s_pod = ems-aio-86495ff465-dffxc [files]
# Copy a single file or a directory recursively to the remote server
# follow the pattern: file_or_folder_key = {'source' : '<file_souce>', 'target' : <file_
      target>'}
# multiple entries are allowed, file_or_folder_key is just a placeholder
# Examples:
# 1 - copying the installer directory recursively:
# installer_dir =
# {'source' : 'C:\\Program Files (x86)\\Fortinet\\FortiClientEMS\\Installers',
```

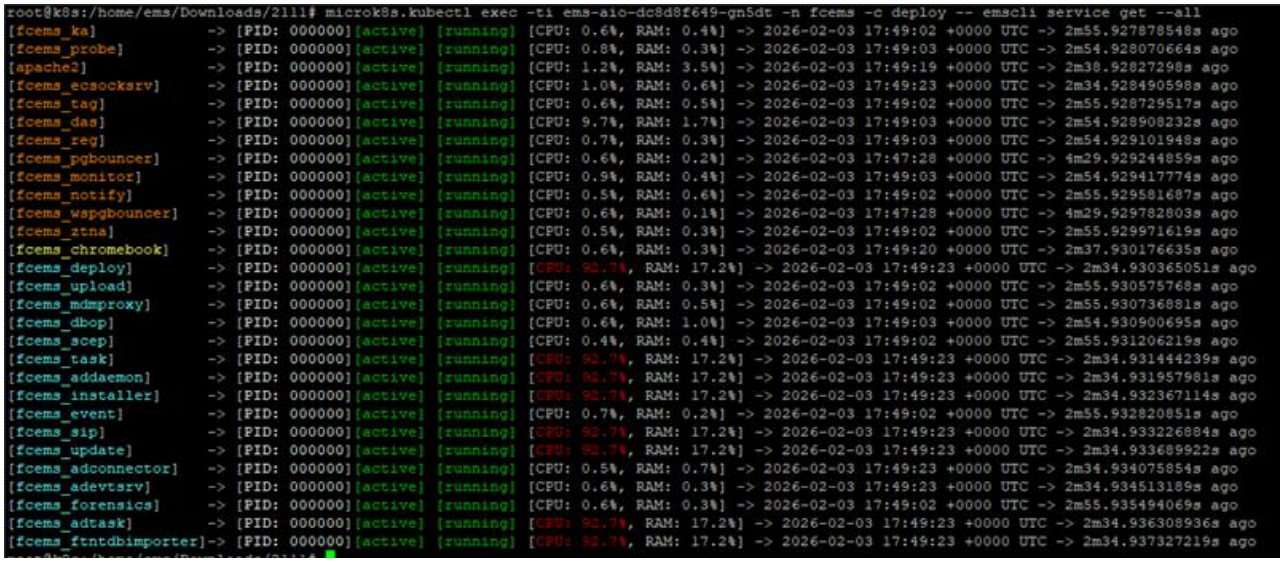
```
# 'target' : '/opt/forticlientems/data'}
# 2 - copying a specific file:
# signatures_file =
# {'source' : 'C:\\Program Files
      (x86)\\Fortinet\\FortiClientEMS\\signatures\\emsaval\\emsaval.dll',
# 'target' : '/opt/forticlientems/data/signatures/emsaval/emsaval.dll'}
```

4. Open an elevated PowerShell prompt inside the EMS Windows server and go to the directory where you extracted the migration tool. Run `migration.exe`:

   ```
   .\migration.exe
   ```

   Wait for the migration to complete. If there are issues, check the migration log in the same folder as the migration tool.
5. Check that all EMS services are running using the following command:

   ```
   microk8s.kubectl exec -ti <pod name> -n <namespace> -c deploy -- emscli service get –all
   ```



6. Access the EMS GUI on the Linux server and verify that all configurations and data from the EMS Windows Server were migrated.

To migrate FortiClient endpoints to Linux EMS:

After EMS migration completes, do one of the following to migrate FortiClient endpoints to Linux EMS:

Method	Description
Update DNS record with Linux EMS IP address (recommended)	On the DNS server, update the DNS record for EMS with the Linux EMS IP address. FortiClient endpoints that were previously connected to Windows Server EMS resolve the new IP address to the EMS FQDN and connect to the Linux EMS. This is the recommended method to migrate FortiClient endpoints to Linux EMS. The endpoint automatically connects to Linux EMS. Even if user verification is enabled, the migration is seamless and the user does not notice any changes on the endpoint.

Method	Description
Switch EMS	1. On the Windows Server EMS, go to *Endpoints > All Endpoints*. 2. Select the desired endpoints. 3. Select *Action > Switch EMS > Switch by IP/Switch by Invitation*. 4. Enter the Linux EMS IP address, FQDN, or invitation. Selected endpoints connect to the Linux EMS. 5. If user verification is enabled, the user onboarding popup for the new invitation displays on the endpoint. The user must verify their identity to connect to Linux EMS.
Configure EMS server list on Windows Server EMS	This method only works if user verification is not enforced and FortiClient connects to EMS using an FQDN or IP address. 1. On the Windows Server EMS, go to *System Settings > EMS Settings*. 2. Enable *Configure EMS server list*. 3. Add the Linux EMS IP address or FQDN and port number. 4. Click *Save*. 5. After the endpoint receives the configuration changes, disconnect or shut down the EMS Windows Server. In the next keepalive interval, FortiClient connects to Linux EMS.

After migration, the license remains active on Windows Server EMS.

Installation

The following topics provide instructions for installing EMS 7.4.6. You can install EMS in various scenarios, such as high availability (HA), with a remote database (DB), and so on. Select the appropriate installation scenario for your environment:

Install scenario	Description
Standalone install with a local DB or PostgreSQL (Postgres) DB	
Installing EMS in standalone mode with a local DB on page 28	Install EMS in standalone mode with a local DB on a Linux machine.
Installing EMS with Postgres in Docker on page 32	Install EMS to use a Postgres database in Docker. You can install the Postgres DB on the same machine as EMS or on a remote machine.
Installing EMS with standalone remote DB without Docker on page 37	Install EMS to use a Postgres DB that is installed on a remote Linux machine. This install does not use Docker for the Postgres DB.
Deploying EMS with Docker Compose on page 42	Deploy EMS on Docker using Docker Compose by defining which containers to start and how to orchestrate them using a yaml file.
Deploying EMS on Kubernetes on page 48	Install EMS on Kubernetes using containers and container orchestration.
Deployment as a virtual machine (VM) image	
Deploying EMS as a VM image on page 51	Deploy EMS as a VM image like many other Fortinet products. EMS supports various hypervisors, including VMware ESXi, KVM, Microsoft Hyper-V, and Oracle VirtualBox hypervisors.
Air-gapped deployment	
Deploying EMS in air-gapped environments on page 70	Installing or upgrading EMS in an environment without direct connectivity to the Internet with the following options: • Air-gapped install or upgrade with EMS Docker containers using Docker Compose or Kubernetes • Air-gapped install or upgrade with a dependencies bundle • Air-gapped install or upgrade using an HTTP proxy EMS 7.4.6 only supports x64 architecture environments for air-gapped deployment. ARM is not currently supported.
HA	
FortiClient EMS supports EMS application HA combined with or without PostgreSQL DB HA. See the EMS HA Deployment Guide for more information.	

For instructions about upgrading your EMS (including VMs), see Upgrading from an earlier FortiClient EMS version.

Installing EMS in standalone mode with a local DB

The following provides instructions for installing EMS in standalone mode with a local database and assumes that you have a machine with Linux installed. You can install EMS in other scenarios, such as high availability, with a remote database, and so on. See Installation on page 27.

Installing EMS on Red Hat Enterprise Linux (RHEL) requires an active Red Hat subscription.

To install standalone EMS:

1. Download the `forticlientems_7.4.6.2170.M.arm64.bin` or `forticlientems_7.4.6.2170.M.amd64.bin` file from the Fortinet Support site.
2. Run `sudo -i` to log in to the shell with root privileges.
3. Change permissions and add execute permissions to the installation file:
   `chmod +x forticlientems_7.4.6.2170.M.XXX64.bin`
   ```
   root@emsnode2:/home/ems/Downloads# chmod +x forticlientems_7.4.0.1745.bin
   ```
4. Set `umask` to 022 if the existing `umask` setting is more restrictive.
5. If you are installing EMS on Red Hat Enterprise Linux (RHEL) 9, do one of the following. EMS 7.4.4 and later versions support install on RHEL 9.:
   - If you are installing EMS on RHEL on Azure, run the following:

     ```
     sudo dnf repolist enabled
     ```

     Verify if `codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms` is in the list. If it is in the list, it is enabled. If it is not in the list, then run the following:

     ```
     sudo subscription-manager repos --enable codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms
     ```

     Run the following commands:

     ```
     sudo dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-$(rpm -E
     %rhel)-$(uname -m)/pgdg-redhat-repo-latest.noarch.rpm
     sudo dnf config-manager --disable pgdg17 pgdg16
     sudo dnf install -y https://rpms.remirepo.net/enterprise/remi-release-$(rpm -q --qf "%
     {VERSION}\n" redhat-release).rpm
     sudo curl -o /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 https://rpms.remirepo.net/RPM-GPG-KEY-
     remi2021
     sudo rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021
     sudo ln -sf /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 /etc/pki/rpm-gpg/RPM-GPG-KEY-
     remi.el$(rpm -q --qf "%{VERSION}\n" redhat-release)
     ```

- If you are installing EMS on RHEL on AWS, run the following command:

```
sudo dnf config-manager --set-enabled codeready-builder-for-rhel-9-rhui-rpms
```

6. Run the following command to install EMS:
`./forticlientems_7.4.6.2170.M.XXX64.bin -- --allowed_hosts '*' --enable_remote_https`
Run the installer to and from any directory other than `/tmp`. Running the installer to or from `/tmp` causes issues.
7. After installation completes, verify that `/etc/timezone` and `/etc/localtime` are configured with the same time zone on the Linux system. Check that all EMS services are running by entering the following command:

```
systemctl --all --type=service | grep -E 'fcems|apache|redis|postgres'
```

```
root@emsnode2:/home/ems/Downloads# systemctl --all --type=service | grep -E 'fcems|apache|redis|postgres'
  apache2.service                          loaded    active   running The Apache HTTP Server
  fcems_adconnector.service                loaded    active   running adconnector service
  fcems_addaemon.service                   loaded    active   running addaemon service
  fcems_adevtsrv.service                   loaded    active   running adevtsrv service
  fcems_adtask.service                     loaded    active   running adtask service
  fcems_chromebook.service                 loaded    active   running chromebook worker service
  fcems_das.service                        loaded    active   running das service
  fcems_dbop.service                       loaded    active   running dbop worker service
  fcems_deploy.service                     loaded    active   running deploy worker service
  fcems_ecsocksrv.service                  loaded    active   running ecsocksrv service
  fcems_forensics.service                  loaded    active   running forensics worker service
  fcems_ftntdbimporter.service             loaded    active   running FTNT DB importer worker service
  fcems_installer.service                  loaded    active   running installer worker service
  fcems_ka.service                         loaded    active   running kaworker service
  fcems_mdmproxy.service                   loaded    active   running MDM proxy service
  fcems_monitor.service                    loaded    active   running monitor worker service
  fcems_notify.service                     loaded    active   running FOS notify service
  fcems_pgbouncer.service                  loaded    active   running pgBouncer for EMS service
  fcems_probe.service                      loaded    active   running probeworker service
  fcems_reg.service                        loaded    active   running regworker service
  fcems_scep.service                       loaded    active   running SCEP service
  fcems_sip.service                        loaded    active   running software inventory processor service
  fcems_tag.service                        loaded    active   running tagworker service
  fcems_task.service                       loaded    active   running taskworker service
  fcems_update.service                     loaded    active   running update worker service
  fcems_upload.service                     loaded    active   running upload worker service
  fcems_wspgbouncer.service                loaded    active   running pgBouncer for EMS WebServer service
  fcems_ztna.service                       loaded    active   running ztna worker service
  postgresql.service                       loaded    active   exited  PostgreSQL RDBMS
  postgresql@15-main.service               loaded    active   running PostgreSQL Cluster 15-main
  redis-server.service                     loaded    active   running Advanced key-value store
```

The output shows that postgresql.service status displays as `exited`. This is the expected status. EMS does not create this service, which only exists to pass commands to version-specific Postgres services. It displays as part of the output as the command filters for all services that contain "postgres" in the name.

8. Access the EMS GUI and log in.
9. If after initially installing EMS 7.4.6 you need to upgrade to a newer build, repeat the process with the new installation file.

Configuring the IP address

After deploying EMS in standalone mode, you may want to configure the IP address. Refer to one of the following procedures, depending on your platform.

An alternative way to configure the IP address is using the `emscli` tool:

- `emscli system set network ip`
- `emscli system set network domain`

Ubuntu:

On Ubuntu, you configure the IP address by modifying the Netplan configuration files.

1. On the Ubuntu machine, locate the Netplan configuration files. Ubuntu stores Netplan configuration files in `/etc/netplan`. The files typically have a .yaml extension, such as 01-netcfg.yaml or 50-cloud-init.yaml. Run the following to list the files:

   ```
   ls /etc/netplan/
   ```

2. Use a text editor such as nano or vim to open the yaml file for editing:

   ```
   sudo nano /etc/netplan/01-netcfg.yaml
   ```

3. In the yaml file, find the section for your desired network interface. Do one of the following:
   - If you are using a static IP address, modify the file, setting `addresses` with your desired static IP address and subnet mask. Update the IP address under `routes: - to: default via:` with your desired gateway, and modify `nameservers` with search domains as needed. The following provides an example configuration where the static IP address is 192.168.1.100/24:

     ```
     network:
         version: 2
         renderer: networkd
         ethernets:
             enp3s0:
                 addresses:
                     - 192.168.1.100/24
                 nameservers:
                     addresses: [8.8.8.8, 1.1.1.1]
                     search: [mydomain1.local, mydomain2.local]
                 routes:
                     - to: default
                       via: 192.168.1.1
     ```

   - If you are using DHCP, ensure that `dhcp4` is set to `yes`. The following provides an example configuration:

     ```
     network:
       version: 2
       renderer: networkd
       ethernets:
         enp0s3:
           dhcp4: yes
     ```

4. Before applying the changes permanently, run the following to test the configuration. This command temporarily applies the configuration and gives you 120 seconds to confirm the changes. If the configuration does not work, it rolls back automatically.

```
sudo netplan try
```

5. If the test succeeds, apply the changes permanently:

```
sudo netplan apply
```

6. To verify the configuration, check that the IP address is updated as you configured:

```
ip addr show
```

RHEL:

You can utilize several methods to configure a network interface with a static IP address on Red Hat Enterprise Linux (RHEL) 9. The following approach uses the nmcli command-line tool, which allows you to manage network connections from the command line.

1. Run the following command to list all network interfaces and identify the one you want to configure:

```
nmcli device status
```

2. Modify the connection using the following command (this example modifies the interface *enp0s3*):

```
sudo nmcli con mod 'enp0s3' ipv4.method manual ipv4.addresses 192.168.1.100/24 ipv4.gateway
192.168.1.1 ipv4.dns "8.8.8.8 8.8.4.4"
```

   Replace *enp0s3* with your actual interface name and adjust the IP address, gateway, and DNS servers as per your network configuration.
3. Apply the network interface changes by restarting the connection using the following command:

```
sudo nmcli con down 'enp0s3' && sudo nmcli con up 'enp0s3'
```

4. Verify the IP configuration using the following commands:

```
ip addr show enp0s3
ip route show
```

CentOS:

To configure the IP address on CentOS:

1. Run the following command to list all network interfaces and identify the one you want to configure:

```
nmcli connection show
```

2. Modify the connection to set a static IP address using the following commands (this example modifies the interface *enp0s3*):

```
sudo nmcli connection modify enp0s3 \
ipv4.method manual \
ipv4.addresses 192.168.1.100/24 \
```

```
ipv4.gateway 192.168.1.1 \
ipv4.dns "8.8.8.8 8.8.4.4" \
connection.autoconnect yes
```

Replace *enp0s3* with your actual interface name and adjust the IP address, gateway, and DNS servers as per your network configuration.

3. Apply the network interface changes by restarting the connection using the following command:

```
sudo nmcli connection down enp0s3 && sudo nmcli connection up enp0s3
```

4. Verify the IP configuration using the following commands:

```
ip addr show enp0s3
ip route show
```

Installing EMS with Postgres in Docker

You can install PostgreSQL (Postgres) in Docker on the same machine as EMS or on a remote machine.

The following guide gives instructions on performing a fresh install of EMS with Postgres in Docker and for upgrading a previously installed EMS deployment with Postgres in Docker to 7.4.6:

- To install EMS with Postgres in Docker: on page 32
- To upgrade a previously installed EMS deployment with Postgres in Docker to 7.4.6: on page 36

To install EMS with Postgres in Docker:

1. Prepare the desired Linux machine(s). If using two machines, you install Postgres on one machine and EMS on the other machine. The following instructions designate some steps for the Postgres machine and others for the EMS machine. If you are using one machine, simply perform all configuration on that machine.
2. On the Postgres machine, do the following:
   a. Run `sudo -i` to log in to the shell with root privileges. Perform all following steps with root privileges.
   b. Install Docker:

```
apt install docker.io
```

   c. Download the Postgres Docker image `forticlientems_7.4.6.2170.M_postgresql15.tar.gz` file from Fortinet Support site.
   d. Load the image:

```
docker load -i forticlientems_postgresql15.tar.gz
```

```
root@sqlserver:/home/ems/Downloads# docker load -i forticlientems_postgresql15.tar.gz
ceb365432eec: Loading layer [==================================================>]  77.83MB/77.83MB
26dc91746b2c: Loading layer [==================================================>]  12.29kB/12.29kB
ec67632ef300: Loading layer [==================================================>]  10.15MB/10.15MB
2c79ad6f81d5: Loading layer [==================================================>]   4.18MB/4.18MB
002a8393fac7: Loading layer [==================================================>]  25.77MB/25.77MB
9b09624652a6: Loading layer [==================================================>]  3.283MB/3.283MB
8aed92e2de60: Loading layer [==================================================>]  1.536kB/1.536kB
670996de4c3e: Loading layer [==================================================>]   7.68kB/7.68kB
961d73daa5ab: Loading layer [==================================================>]  311.6MB/311.6MB
d2920b6da2df: Loading layer [==================================================>]  67.58kB/67.58kB
9e3bd0ff62e8: Loading layer [==================================================>]  2.048kB/2.048kB
f70f12458f07: Loading layer [==================================================>]  3.072kB/3.072kB
0e59cf63101c: Loading layer [==================================================>]  18.94kB/18.94kB
a3c128b62e91: Loading layer [==================================================>]  3.072kB/3.072kB
f8d6beb0af73: Loading layer [==================================================>]  20.48kB/20.48kB
ba4de6a805db: Loading layer [==================================================>]  19.46kB/19.46kB
d7b0b96f3747: Loading layer [==================================================>]  19.46kB/19.46kB
69448970b7ff: Loading layer [==================================================>]  19.46kB/19.46kB
4da37bb63387: Loading layer [==================================================>]  22.02kB/22.02kB
27d2164c0deb: Loading layer [==================================================>]  4.608kB/4.608kB
c72660a042fb: Loading layer [==================================================>]  32.26kB/32.26kB
8935552f48f8: Loading layer [==================================================>]  4.608kB/4.608kB
006b5352c0f1: Loading layer [==================================================>]  4.608kB/4.608kB
8f9bbfba2396: Loading layer [==================================================>]  3.584kB/3.584kB
b870106d5359: Loading layer [==================================================>]  3.584kB/3.584kB
Loaded image: ems_postgresql15:latest
```

e. List the images on Docker to verify the image has been created or loaded:

```
docker image ls
```

```
root@sqlserver:/home/ems/Downloads# docker image ls
REPOSITORY         TAG       IMAGE ID       CREATED       SIZE
ems_postgresql15   latest    6ce48c81f364   3 weeks ago   425MB
```

f. Run the Docker container. The following shows the command to run a container:

```
docker run --restart always --name <container name> -e POSTGRES_PASSWORD=<password> -p
<local port number>:<PostgreSQL port number:5432> -d <container instance name><default
username> -N <number>
```

The following details the options for the command:

Option	Description
`-e`	Set environment variables.
`-p`	Publish all exposed ports to random ports.
`-d`	Run container in the background and print container ID.
`-N`	Maximum number of concurrent connections allowed to the containerized Postgres database.
`--restart always`	Ensures that if the host restarts, it starts the container automatically.

The following shows an example command with example values:

```
docker run --restart always --name ems_docker -e POSTGRES_PASSWORD=Fortinet123# -p
6434:5432 -d ems_postgresql15 postgres
```

The example command configures Fortinet123# as the password. Ensure that you configure your own unique password as to not compromise the security of your installation.

You can use any container instance name and password. In this example, the container's Postgres port, 5432, is exposed to port 6434 on the machine where Docker is running.

This allows you to have several instances of Postgres containers running and isolated from each other as long as they use different local host ports. You can use any port number as a local port for a Postgres container.

3. On the EMS machine, install EMS and connect to the database:
   a. Download the `forticlientems_7.4.6.2170.M.XXX64.bin file` from the Fortinet Support site.
   b. Change permissions and add execute permissions to the installation file:

```
chmod +x forticlientems_7.4.6.2170.M.XXX64.bin
```

   c. Set `umask` to 022 if the existing `umask` setting is more restrictive.
   d. If you are installing EMS on Red Hat Enterprise Linux (RHEL) 9, do one of the following. EMS 7.4.4 and later versions support install on RHEL 9.:
      - If you are installing EMS on RHEL on Azure, run the following:

```
sudo dnf repolist enabled
```

        Verify if `codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms` is in the list. If it is in the list, it is enabled. If it is not in the list, then run the following:

```
sudo subscription-manager repos --enable codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms
```

        Run the following commands:

```
sudo dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-$(rpm -E %rhel)-$(uname -m)/pgdg-redhat-repo-latest.noarch.rpm
sudo dnf config-manager --disable pgdg17 pgdg16
sudo dnf install -y https://rpms.remirepo.net/enterprise/remi-release-$(rpm -q --qf "%{VERSION}\n" redhat-release).rpm
sudo curl -o /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 https://rpms.remirepo.net/RPM-GPG-KEY-remi2021
sudo rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021
sudo ln -sf /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 /etc/pki/rpm-gpg/RPM-GPG-KEY-remi.el$(rpm -q --qf "%{VERSION}\n" redhat-release)
```

      - If you are installing EMS on RHEL on AWS, run the following command:

```
sudo dnf config-manager --set-enabled codeready-builder-for-rhel-9-rhui-rpms
```

   e. Start the EMS installation and connect to the Postgres database on the Docker container. The following shows the command to do so:

```
sudo ./<ems installation script file> -- --db_host <IP address or FQDN> --db_port <local port> --db_user <username> --db_pass <password> --skip_db_install --allowed_hosts '*' --enable_remote_https
```

The following shows an example command with example values:

```
./forticlientems_7.4.6.2170.M.bin -- --db_host 192.168.1.20 --db_port 6434 --db_user postgres --db_pass Fortinet123# --skip_db_install --allowed_hosts '*' --enable_remote_https
```

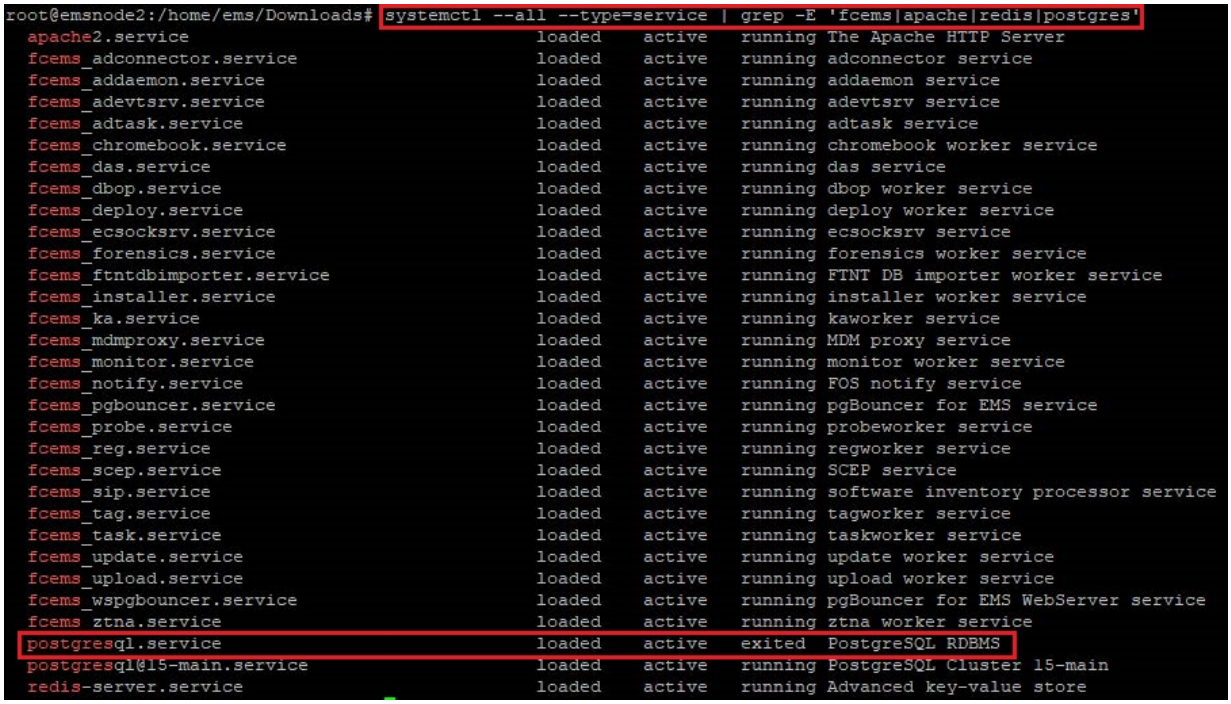
The example command configures Fortinet123# as the password. Ensure that you configure your own unique password as to not compromise the security of your installation.

Run the installer to and from any directory other than `/tmp`. Running the installer to or from `/tmp` causes issues.

`db_host` is the Postgres Docker machine IP address or FQDN.

f. After installation completes, verify that `/etc/timezone` and `/etc/localtime` are configured with the same time zone on the Linux system. Check that all EMS services are running by entering the following command:

```
systemctl --all --type=service | grep -E 'fcems|apache|redis|postgres'
```



The output shows that postgresql.service status displays as `exited`. This is the expected status. EMS does not create this service, which only exists to pass commands to version-specific Postgres

services. It displays as part of the output as the command filters for all services that contain "postgres" in the name.

g. Access the EMS GUI and log in.

h. If after initially installing EMS 7.4.6 you need to upgrade to a newer build, repeat the steps with the new installation file.

To upgrade a previously installed EMS deployment with Postgres in Docker to 7.4.6:

If you are upgrading a previously installed EMS deployment with Postgres in Docker from 7.4.3 or an earlier 7.4 version to 7.4.6, you must use the following procedure to ensure that your Postgres instance is upgraded without incurring data loss.

1. Stop EMS services:

```
sudo emscli service stop web probe notify ztna ka monitor ec wspgbouncer das pgbouncer reg tag chromebook deploy task installer upload adevtsrv dbop adconnector mdmproxy scep sip update addaemon forensics ftntdbimporter adtask
```

2. Tag the old Docker image to keep it:

```
docker tag ems_postgresql15:latest ems_postgresql15:15.6
```

3. Download and load the new Postgres Docker image from the Fortinet Support portal:

```
docker load -i forticlientems_7.4.6.2170.M_postgresql15.tar.gz
```

4. Stop the old Postgres container:

```
docker stop ems_docker
```

5. Obtain the name of the old data volume from the old Postgres container. Save the returned value:

```
docker inspect --format='{{range .Mounts}}{{if eq .Destination "/var/lib/postgresql/data"}}
{{.Name}}{{end}}{{end}}' ems_docker
```

6. Configure a new Postgres container:

a. Create a new named Docker volume:

```
docker volume create ems-postgres-data
```

b. Copy the data from the old volume to the newly created one:

```
docker run --rm -v <value from step 5>:/from -v ems-postgres-data:/to alpine ash -c "cd /from ; cp -av . /to"
```

The following shows an example of this command:

```
docker run --rm -v 4af4f59229ffc....aea1381b329bceeefbc9:/from -v ems-postgres-data:/to alpine ash -c "cd /from ; cp -av . /to"
```

c. Create a new Postgres container using the new image and the new volume:

```
docker run --name ems_docker_new --restart unless-stopped -v ems-postgres-
data:/var/lib/postgresql/data -e POSTGRES_PASSWORD=<password> -p 6434:5432 -d ems_
postgresql15 postgres
```

d. Ensure the container is running by checking its status with the following command:

```
docker ps
```

7. Start EMS services:

```
sudo emscli service start web probe notify ztna ka monitor ec wspgbouncer das pgbouncer reg
tag chromebook deploy task installer upload adevtsrv dbop adconnector mdmproxy scep sip update
addaemon forensics ftntdbimporter adtask
```

8. If EMS runs as expected, delete the old container:

```
docker rm ems_docker
```

Installing EMS with standalone remote DB without Docker

In this installation scenario, EMS and the Postgres DB are hosted on two Linux machines. The following uses PostgreSQL (Postgres) for the remote database (DB). You can choose to install the Postgres DB server on Ubuntu or RHEL.

To configure the Postgres DB server:

1. Install Postgres 18 on the Postgres Linux machine:
   - **Ubuntu**:

```
sudo apt install -y --no-install-recommends curl ca-certificates
sudo install -d /usr/share/postgresql-common/pgdg
sudo curl -o /usr/share/postgresql-common/pgdg/apt.postgresql.org.asc --fail
https://www.postgresql.org/media/keys/ACCC4CF8.asc
sudo sh -c 'echo "deb [signed-by=/usr/share/postgresql-common/pgdg/apt.postgresql.org.asc]
https://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" >
/etc/apt/sources.list.d/pgdg.list'
sudo apt update
sudo apt install -y postgresql-18
```

   - **RHEL**:

```
sudo dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-$(rpm -E
%rhel)-$(uname -m)/pgdg-redhat-repo-latest.noarch.rpm
sudo dnf -qy module disable postgreSQL
sudo dnf install -y postgresql18
```

```
sudo dnf install -y postgresql18-server postgresql18-contrib
sudo /usr/pgsql-18/bin/postgresql-18-setup initdb
sudo systemctl enable --now postgresql-18
sudo systemctl status postgresql-18
```

2. Install the EMS custom extension in either of the following ways:
   - Download the EMS custom extension (`forticlientems_7.4.6.2170_pg_extension_ubuntu.tar.gz` or `forticlientems_7.4.6.2170_pg_extension_rhel.tar.gz`) from the Fortinet Support site and install it using either of the following commands:
     - **Ubuntu**:

       ```
       sudo tar zxvf forticlientems_7.4.6.2170_pg_extension_ubuntu.tar.gz -C /
       sudo systemctl restart postgresql
       ```

     - **RHEL**:

       ```
       sudo tar zxvf forticlientems_7.4.6.2170_pg_extension_rhel.tar.gz -C /
       sudo systemctl restart postgresql
       ```

   - Install pgAgent which will create the necessary extension required by EMS:
     - **Ubuntu**:

       ```
       sudo apt install -y pgagent
       ```

     - **RHEL**:

       ```
       sudo dnf install -y pgagent_18
       ```

3. You can tune Postgres based on the host server specs by applying the recommended configuration. While there are various tools you can use to find the recommended configuration, these instructions use PGTune. Generate and copy the recommended configuration:
   a. Go to PGTune and enter the following information:

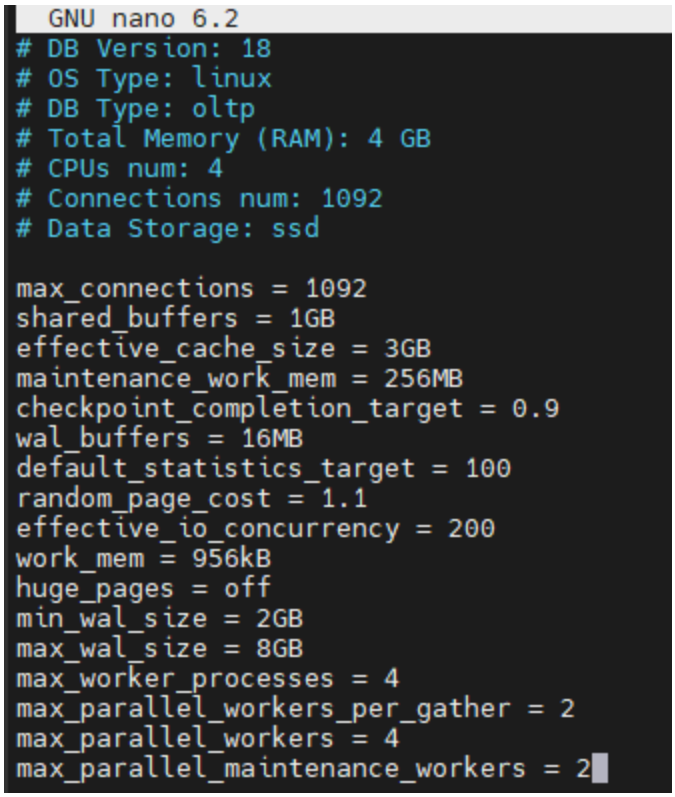
Field	Value
DB version	18
OS Type	Linux
DB Type	Online transaction processing system
Total Memory (RAM)	Enter the total memory for your Postgres server. In this example, it is 4 GB.
Number of CPUs	Enter the total number of CPUs for your Postgres server. In this example, it is 4.
Number of Connections	1092
Data Storage	Enter the data storage type as per your device. In this example, it is SSD storage.

   b. Click *Generate*.

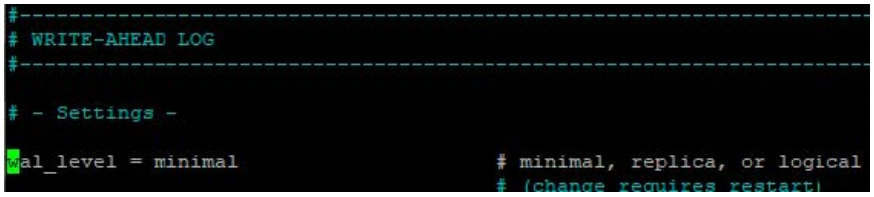
c. Click *Copy configuration*.

4. Update `/etc/postgresql/18/main/postgresql.conf` (Ubuntu) or `/var/lib/pgsql/18/data/postgresql.conf` (RHEL):

a. Add or update the configuration with the content that you copied in step 3.



b. Uncomment and change `wal_level` to `minimal`, e.g. `wal_level = minimal`. This removes all logging except the information required to recover from a crash or immediate shutdown.



c. Uncomment and change `max_wal_senders` to `0`, e.g. `max_wal_senders = 0`. This disables replication.

d. Uncomment and change `listen_addresses` to `'*'`, e.g. `'localhost' >> listen_addresses = '*'`.

5. Give passwordless permission to the Postgres user on the same machine by changing the configuration file by running either of the following commands:

- **Ubuntu**:

```
sed -i 's/# Database administrative login by Unix domain socket/\nhost all postgres
127.0.0.1\/32 trust\nhost all postgres ::1\/128 trust/' /etc/postgresql/18/main/pg_
hba.conf
```

- **RHEL**:

```
sed -i 's/# Database administrative login by Unix domain socket/\nhost all postgres
127.0.0.1\/32 trust\nhost all postgres ::1\/128 trust/' /var/lib/pgsql/18/data/pg_hba.conf
```

6. Allow connection from the remote machine by updating the following in `pg_hba.conf` (located in `/etc/postgresql/18/main/` for Ubuntu or `/etc/postgresql/18/main/` for RHEL):

```
# IPv4 local connections:
host all all 127.0.0.1/32 scram-sha-256 >> host all all 0.0.0.0/0 scram-sha-256
```

7. Restart the postgreSQL service:
   - **Ubuntu**:

     ```
     systemctl restart postgresql
     ```

   - **RHEL**:

     ```
     systemctl restart postgresql-18
     ```

8. Change the postgres user password:

   ```
   sudo -u postgres psql
   ALTER USER postgres PASSWORD '<password>';
   ```

To install EMS:

1. On the second Linux machine, download the `forticlientems_7.4.6.2170.M.bin` file from https://support.fortinet.com.
2. Set `umask` to 022 if the existing `umask` setting is more restrictive.
3. If you are installing EMS on Red Hat Enterprise Linux (RHEL) 9, do one of the following. EMS 7.4.4 and later versions support install on RHEL 9.:
   - If you are installing EMS on RHEL on Azure, run the following:

     ```
     sudo dnf repolist enabled
     ```

     Verify if `codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms` is in the list. If it is in the list, it is enabled. If it is not in the list, then run the following:

     ```
     sudo subscription-manager repos --enable codeready-builder-for-rhel-9-x86_64-eus-rhui-rpms
     ```

     Run the following commands:

     ```
     sudo dnf install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-$(rpm -E
     %rhel)-$(uname -m)/pgdg-redhat-repo-latest.noarch.rpm
     sudo dnf config-manager --disable pgdg17 pgdg16
     sudo dnf install -y https://rpms.remirepo.net/enterprise/remi-release-$(rpm -q --qf "%
     {VERSION}\n" redhat-release).rpm
     sudo curl -o /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 https://rpms.remirepo.net/RPM-GPG-KEY-
     remi2021
     sudo rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021
     sudo ln -sf /etc/pki/rpm-gpg/RPM-GPG-KEY-remi2021 /etc/pki/rpm-gpg/RPM-GPG-KEY-
     remi.el$(rpm -q --qf "%{VERSION}\n" redhat-release)
     ```

   - If you are installing EMS on RHEL on AWS, run the following command:

     ```
     sudo dnf config-manager --set-enabled codeready-builder-for-rhel-9-rhui-rpms
     ```

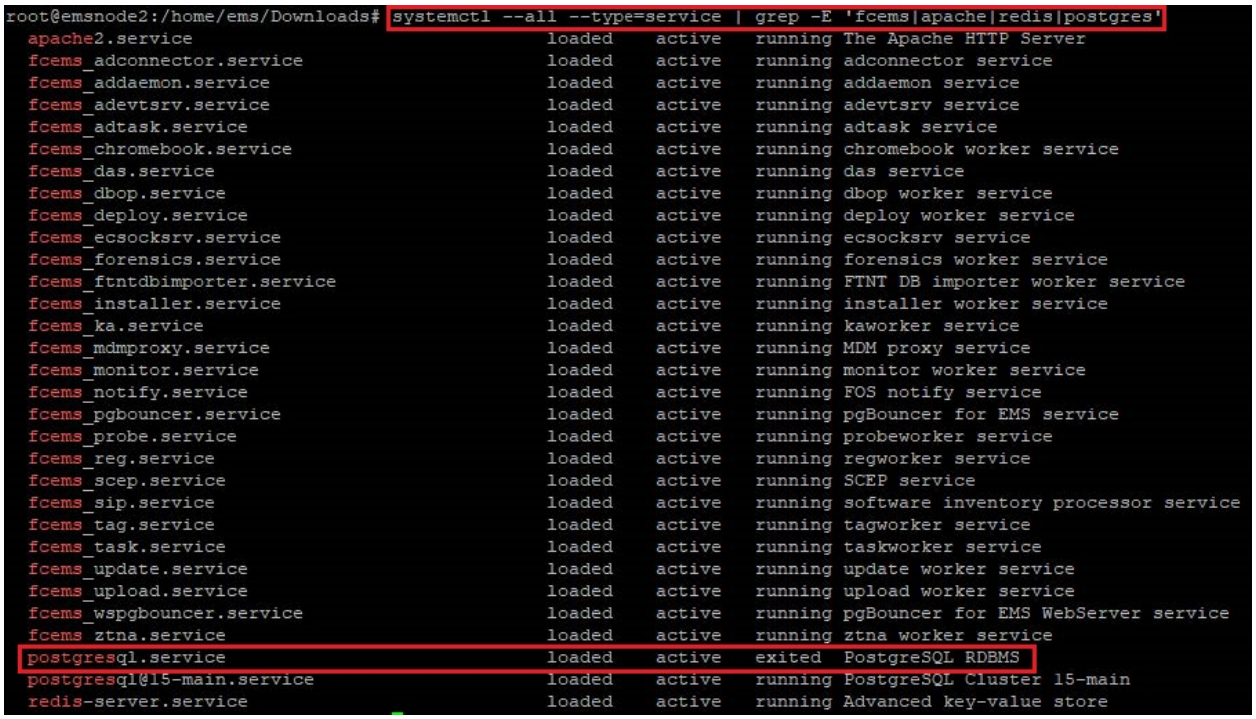
4. Install EMS. db_host is the remote Postgres server:

```
sudo chmod +x forticlientems_..bin
sudo ./forticlientems_..bin -- --db_host "172.16.1.26" --db_user postgres --db_pass <password>
--skip_db_install --allowed_hosts '*' --enable_remote_https
```

   Run the installer to and from any directory other than /tmp. Running the installer to or from /tmp causes issues.

5. After installation completes, verify that /etc/timezone and /etc/localtime are configured with the same time zone on the Linux system. Check that all EMS services are running by entering the following command:

```
systemctl --all --type=service | grep -E 'fcems|apache|redis|postgres'
```



   The output shows that postgresql.service status displays as exited. This is the expected status. EMS does not create this service, which only exists to pass commands to version-specific Postgres services. It displays as part of the output as the command filters for all services that contain "postgres" in the name.

6. Access the EMS GUI and log in.
7. If you need to upgrade to a newer build after the initial installation of EMS 7.4.6, repeat the steps with the new installation file.

Deploying EMS with Docker Compose

You can deploy EMS on Docker using Docker Compose by defining which containers to start and how to orchestrate them using a yaml file.

After deployment, you can migrate existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on Docker on page 14.

To deploy EMS with Docker Compose:

1. Download the EMS Docker image files from the Fortinet support site:
   - `forticlientems_7.4.6.2170.M_docker.tar.gz`
   - `forticlientems_7.4.6.2170.M_docker_compose.zip`
2. Load the Docker image:

```
docker load -i forticlientems_7.4.6.2170.M_docker.tar.gz
```

```
root@ems:/home/ems/Downloads# docker load -i forticlientems_7.4.5.2111.M_docker.tar
Loaded image: ems_workers:7.4.5.2111
Loaded image: ems_adconnector:7.4.5.2111
Loaded image: ems_deploy:7.4.5.2111
Loaded image: ems_dbop:7.4.5.2111
Loaded image: ems_fos:7.4.5.2111
Loaded image: ems_das:7.4.5.2111
Loaded image: ems_webserver:7.4.5.2111
Loaded image: ems_grafana/grafana:7.4.5.2111
Loaded image: ems_grafana/promtail:7.4.5.2111
Loaded image: ems_grafana/loki:7.4.5.2111
Loaded image: ems_mdm:7.4.5.2111
Loaded image: ems_scep:7.4.5.2111
Loaded image: ems_nginx:7.4.5.2111
Loaded image: ems_pgbouncer:7.4.5.2111
Loaded image: ems_daemons:7.4.5.2111
Loaded image: ems_postgresql15:latest
Loaded image: ems_consul:7.4.5.2111
Loaded image: ems_redis:7.4.5.2111
Loaded image: ems_haproxy:7.4.5.2111
root@ems:/home/ems/Downloads#
```

3. Unzip `forticlientems_7.4.6.2170.M_docker_compose.zip`, which includes two yaml files and one readme file.
4. Rename `docker-compose-remote-db.yaml` or `docker-compose-with-db.yaml` as `docker-compose.yaml`, depending on whether you use a remote or local EMS database for your EMS installation.
5. Create an environment file named `.env` with the following variables and put it in the same location as the Docker Compose file:

Variable	Description
Required	
`EMS_VERSION`	Version of EMS containers to deploy. For example, `7.4.6.2170`.
`POSTGRESQL_HOST`	IP address or host name of the PostgreSQL server that EMS must connect to. Leave this field blank if you use a local EMS DB.

Variable	Description
	When using a postgreSQL cluster with multiple nodes, this variable can support multiple nodes, comma separated with the current primary node at the beginning. For example, `POSTGRESQL_HOST=node1,node2,node3,node4`. Alternatively, specify a DC name for each of the hosts in the list. For example, `POSTGRESQL_HOST=node1@dc1,node2@dc1,node3@dc2,node4@dc3`. EMS can use the DC information in conjunction with `EMS_PREFERRED_DC` to make failover decisions when in HA.
`POSTGRESQL_PORT`	Port of the PostgreSQL server that EMS must connect to. Leave this field blank if you use a local EMS DB. When using a PostgreSQL cluster with multiple nodes, this variable support multiple nodes (comma separated) which must match the number and sequence of nodes provided in `POSTGRESQL_HOST`. For example, if `POSTGRESQL_HOST` is pg1, pg2 and pg3 with pg1 and pg3 listening on 5432 while pg2 listening on 6432, `POSTGRESQL_PORT` must be `5432,6432,5432`. If only a single port is provided, that single port will be used for all nodes. Default port is 5432.
`POSTGRESQL_USER`	The user that EMS will use to connect to the database. Leave this field blank if you use a local EMS DB. Default user is postgres.
`POSTGRESQL_PASSWORD`	• For a remote EMS DB, specify the password of the user that EMS will use to connect to the remote DB. • For local EMS DB, this will be the password for the PostgreSQL database. The default password is `adubejbdIBEEIHVHEDVWJBNSVqivsw`.
`EMS_DB_PREFIX`	Prefix to add to the database name. The default is empty, in which case EMS will create the `fcm` and `fcm_default` databases. If a prefix value is provided, such as "uat_", EMS will append it to the database names: `uat_fcm` and `uat_fcm_default`. This is useful to segregate the data for each EMS instance when multiple EMS instances connect to the same DB server.
Optional	
`EMS_AIRGAP`	Specifies whether it is an air-gapped environment. Acceptable values are `true` or `false` (default). When set to true, you will be able to upload your license files during initial setup for air-gapped environments without access to the Internet.
`ENABLE_EVENT_FEATURE`	Specifies whether to enable the *Consolidated Events* feature on EMS, which sends events to an elastic search database. Acceptable values are `true` or `false` (default).

Variable	Description
	When set to true, configure the following options for elastic search: `ES_HOSTS` — List of elastic search hosts for EMS to connect to. `ES_USER` — User account to use for the elastic search connection. This variable is ignored if an API key is configured in `ES_API_KEY`. `ES_PASSWORD` — Password for the account to use for the elastic search connection. This variable is ignored if an API key is configured in `ES_API_KEY`. `ES_API_KEY` — API key to use for the elastic search connection. EMS can use either API key or user/password to connect to elastic search. If both are set, API key will be used and user/password will be ignored. `CA_CERT_FOR_ES` — Full qualified path of the CA certificate for the ES cluster located on the host computer.
`INSTANCE_NAME`	If you want to run multiple EMS Docker on the same host, specify the instance name so that components created as part of the Docker compose, such as volumes, networks, do not conflict with one another.
`EXTERNAL_IP`	If you want to run multiple EMS Docker on the same host, you must specify the external IP of each network interface. Otherwise, 0.0.0.0 will be used for all interfaces. For example, for a VM with two network interfaces with IP 192.168.122.217 and 192.168.122.12, you can run ems1 with EXTERNAL_IP=192.168.122.217 and ems2 with EXTERNAL_IP=192.168.122.12 so you can access each on those specific IPs.
`REGISTRY_PATH`	Define the registry path to pull EMS Docker images from. For example, if your registry runs on `mycomp.docker.reg.io` and images are pushed to group `` `fortinet/ems` ``, set `REGISTRY_PATH` to `` `mycomp.docker.reg.io/fortinet/ems/` `` (Note that it must end with a slash) If no registry path is specified, the local docker cache is used.
`EMS_FIPS_ENABLED`	Specifies whether to initialize and operate in OpenSSL FIPS mode across all EMS containers. Acceptable values are `true` or `false` (default).

Variable	Description
SCEP_PUBLIC_HOSTNAME	Public hostname or FQDN accessible by mobile endpoints when using MDM integration. Define this value so that those endpoints can pull their ZTNA certificates from the SCEP service on EMS.
EMS_NODE_ALIAS	Alias used to identify the EMS in the list of EMS nodes in HA. If undefined, the host name will be used.
EMS_PREFERRED_DCS	Preferred data centers for the EMS node to follow in HA. Separate multiple values with a comma. When failover happens, EMS verifes the DC of the current DB primary node and gives promotion preference to EMS nodes that use a preferred DC.

6. Deploy EMS with the defined configuration using Docker Compose:

```
docker compose up -d
```

```
root@ems:/home/ems/Downloads# docker compose up -d
[+] up 31/31
 ✔ Network fcems_forticlientems    Created
 ✔ Container fcems-grafana-1       Created
 ✔ Container fcems-loki-1          Created
 ✔ Container fcems-promtail-1      Healthy
 ✔ Container fcems-db-1            Healthy
 ✔ Container fcems-deploy-1        Created
 ✔ Container fcems-redis-1         Healthy
 ✔ Container fcems-consul-1        Healthy
 ✔ Container fcems-ws_pgbouncer-1  Created
 ✔ Container fcems-pgbouncer-1     Created
 ✔ Container fcems-webserver-1     Created
 ✔ Container fcems-das-1           Healthy
 ✔ Container fcems-fos-1           Created
 ✔ Container fcems-ztna-1          Created
 ✔ Container fcems-chromebook-1    Created
 ✔ Container fcems-adconnector-1   Created
 ✔ Container fcems-monitor-1       Created
 ✔ Container fcems-upload-1        Created
 ✔ Container fcems-tag-1           Created
 ✔ Container fcems-reg-1           Created
 ✔ Container fcems-ecsocksrv-1     Created
 ✔ Container fcems-adevtsrv-1      Created
 ✔ Container fcems-probe-1         Created
 ✔ Container fcems-dbop-1          Created
 ✔ Container fcems-daemons-1       Created
 ✔ Container fcems-ka-1            Created
 ✔ Container fcems-event-1         Created
 ✔ Container fcems-forensics-1     Created
 ✔ Container fcems-nginx-1         Created
 ✔ Container fcems-mdmproxy-1      Created
 ✔ Container fcems-scep-1          Created
root@ems:/home/ems/Downloads#
```

7. Check the health of the EMS services by running `docker compose ps`.
   - To view logs for a specific service: `docker compose logs <service>`
   - To view logs for multiple services: `docker compose logs --tail 100 -f <service1> <service2>`
   - To view logs for all services: `docker compose logs`
8. After verifying that all EMS services are running, access EMS using `ip/fqdn` in the browser.

To upgrade an existing EMS Docker deployment to 7.4.6:

1. Stop the current EMS services:

```
docker compose down
```

2. If you are not using jfrog, load the new EMS images to the docker cache:

```
docker load -i forticlientems_7.4.6.2170.M_docker.tar.gz
```

3. In the `.env` file, change the value of the `EMS_VERSION` variable to 7.4.6.2170.

4. Restart the service and force recreate to ensure that the new version of containers are used:

```
docker compose up -d --force-recreate
```

Deploying EMS on Kubernetes

You can deploy EMS on Kubernetes using containers and container orchestration.

After deployment, you can migrate existing EMS 7.2.13 or 7.2.14 configurations to EMS 7.4.6 on Kubernetes on page 20.

To deploy EMS on Kubernetes:

1. Download the EMS Docker image and helm chart files from the Fortinet support site:
   - `forticlientems_7.4.5.2111.M_docker.tar.gz`
   - `forticlientems_7.4.5.2111.M_helm.zip`
2. Unzip `forticlientems_7.4.5.2111.M_docker.tar.gz` and load the Docker image:

```
microk8s ctr image import forticlientems_7.4.5.2111.M_docker.tar
```

3. Unzip `forticlientems_7.4.5.2111.M_helm.zip`, which includes two yaml files.
4. Open `values.yaml` and update the following variables as needed:

For complex values, we recommend creating a copy of the `values.yaml` file to update the variables.

Variable	Description
`namespace`	Custom namespace to create on your k8s cluster and deploy EMS.

Variable	Description
	No validation is performed on whether this namespace already exists or whether it already has EMS installed.
`db.host`	Specify the remote DB host to install EMS. If not specified, the helm chart will deploy a database pod.
`db.port`	Port of the remote DB that EMS will connect to.
`db.user`	The user that EMS will use to connect to the remote DB.
`db.password`	Password of the user that EMS will use to connect to the remote DB.
`db.prefix`	Prefix to add to the database name. This is optional and only recommended when installing multiple EMS instances that connect to the same DB server.
`image.regPrefix`	Image registry to pull EMS images from. For example, `xxxx.corp.fortinet.com/forticlient/ems`.
`image.tag`	Version of EMS images to deploy. For example, `7.4.5.2111.M`.
`elasticsearch.host`	List of elastic search hosts for EMS to connect to, separated with comma. For example, `"host1:9200,host2:9200"`.
`elasticsearch.user`	User account to use for the elastic search connection. This variable is ignored if an API key is configured in `elasticsearch.apiKey`.
`elasticsearch.password`	Password for the account to use for the elastic search connection. This variable is ignored if an API key is configured in `elasticsearch.apiKey`.
`elasticsearch.apiKey`	API key to use for the elastic search connection. EMS can use either API key or user/password to connect to elastic search. If both are set, API key will be used and user/password will be ignored.
`elasticsearch.ca_cert`	Contents of the CA certificate that EMS uses to connect to the elastic search cluster.
`multiPod`	Specifies whether EMS will create multiple pods for its various containers. Acceptable values are `true` or `false`. When set to false, all containers are deployed in a single pod.
`data.root`	Root to create directories for the EMS volumes in local storage. The default root is `/data/ems` and the volumes are created under `/data/ems/<namespace>`.
`fips`	Specifies whether to initialize and operate in OpenSSL FIPS mode across all EMS containers. Acceptable values are `true` or `false` (default).

Variable	Description
scepPublicHostname	Public hostname or FQDN accessible by mobile endpoints when using MDM integration. Define this value so that those endpoints can pull their ZTNA certificates from the SCEP service on EMS.
nodeAlias	Alias used to identify the EMS in the list of EMS nodes in HA. If undefined, the host name will be used.
preferredDCs	Preferred data centers for the EMS node to follow in HA. Separate multiple values with a comma. When failover happens, EMS verifes the DC of the current DB primary node and gives promotion preference to EMS nodes that use a preferred DC.

5. Run the following command to install the helm chart:

```
microk8s.helm install ems .
```

```
root@k8s:/home/ems/Downloads/2111# microk8s.helm install ems .
NAME: ems
LAST DEPLOYED: Fri Dec 12 14:28:55 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
FortiClient EMS 7.4.5.2111
Namespace: fcemsnew
Running in single-pod mode
```

If you use a copy file of `values.yaml` with a different name or location (i.e, not in the same folder as the rest of the chart), install the helm chart from the charts directory by running `helm install <name><params> -f /home/user/<yaml filename>`.

6. Locate the external IP for EMS:

```
microk8s kubectl get svc -A
```

You will now be able to access EMS using `ip/fqdn` in the browser.

7. To upgrade EMS on Kubernetes, repeat the steps above with new EMS Docker image and helm chart files but use the following command instead in step 5:

```
microk8s.helm upgrade ems .
```

If you use a copy file of `values.yaml` with a different name or location (i.e, not in the same folder as the rest of the chart), upgrade the helm chart from the charts directory by running:

`helm upgrade <name> <params> -f /home/user/<yaml filename>.`

Deploying EMS as a VM image

You can deploy EMS as a virtual machine (VM) image on VMware ESXi, KVM, Microsoft Hyper-V, and Oracle VirtualBox hypervisors. EMS provides Ubuntu-based VM images for x86_64 and ARM architectures. However, regular OS maintenance is still required.

The VM image include some OS hardening modifications as follows:

- Unneeded users are removed:
  - games
  - man
  - news
  - uucp
  - proxy
  - backup
  - list
  - irc
  - gnats
  - uuidd
  - mail
  - lp
  - nobody
  - tss
  - landscape
  - fwupd-efresh
  - usbmux
  - lxd
- The forticlientems user, which runs EMS processes, has no login.
- Only the ems user has SSH access.
- Firewall is enabled and only the following ports are enabled by default:

TCP port	Usage
22	SSH access to EMS VM or server

TCP port	Usage
4001	Send zero trust network access certificates to mobile device management endpoints
8013	Telemetry
8015	Send updates to FortiOS
8443	Provision profiles to Chromebooks
8871	Connection to remote Active Directory connector
• 80 • 443 • 10443 • 9443	EMS GUI and APIs

- On first login, EMS requires changing the password for the ems user.

This topic contains instructions for deploying EMS as a VM as follows:

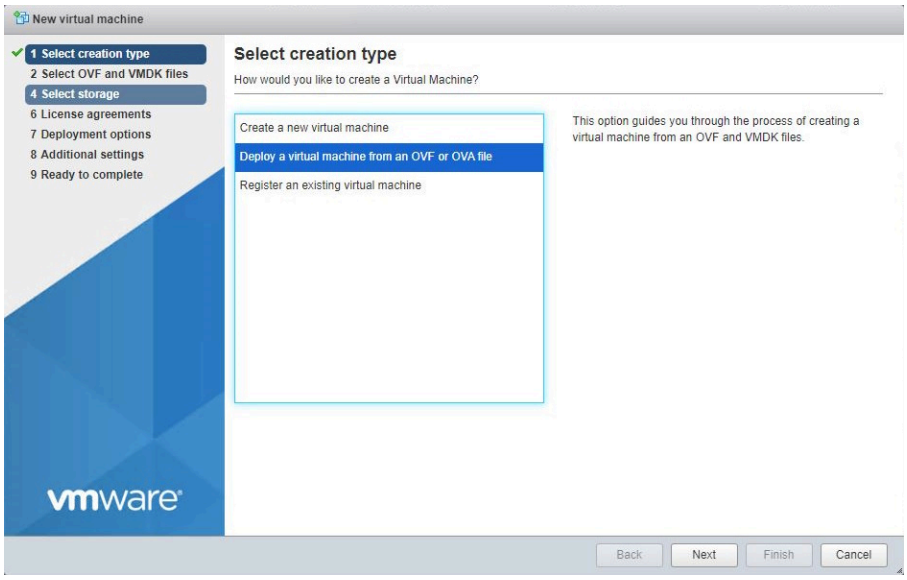
- VMware ESXi on page 52
- KVM on page 54
- Proxmox on page 55
- Hyper-V on page 60
- VirtualBox on page 61
- Configuring the IP address on page 63
- Configuring the search domain on page 64
- Upgrading OS packages on page 64
- Recovering FortiClient EMS VM password on page 65
- Adding and expanding disks on EMS VMs on page 65

- For instructions about upgrading your EMS VM to 7.4.6, see Upgrading from an earlier FortiClient EMS version.
- For instructions about HA deployment for your EMS VMs, see Setting up HA for EMS VM appliances.

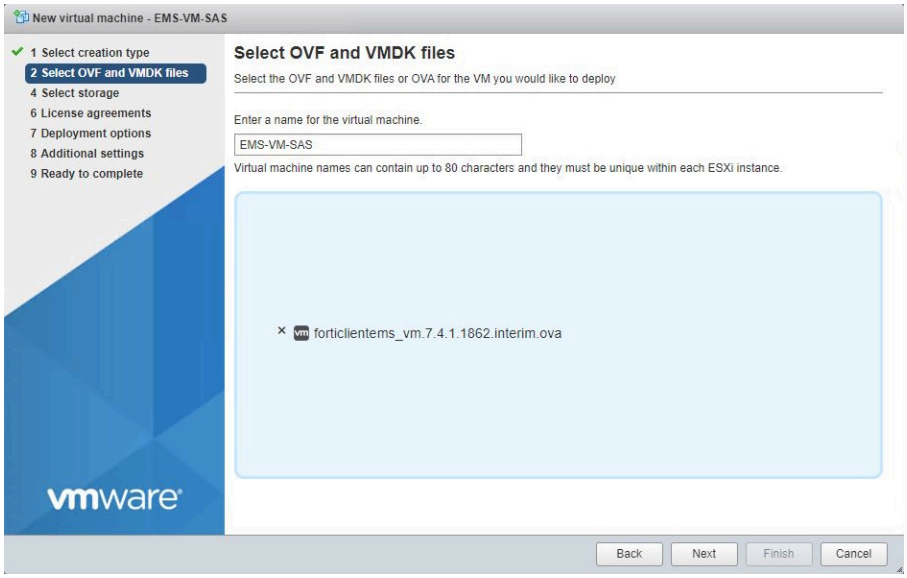
VMware ESXi

To deploy EMS on VMware ESXi:

1. Click *Create/Register VM*.
2. Select *Deploy a virtual machine from an OVF or OVA file*. Click *Next*.



3. Enter the VM name and upload the OVA file. Click *Next*.



4. Configure the VM.
5. Click *Finish*.
6. Review the configuration and start the VM. When the VM boot completes, the OS logon page displays.
7. Log in to the VM. The default credentials are:
   - Username: `ems`
   - Password: `ems`

   EMS requires you to change these credentials upon first login.
8. Change the default password when prompted.
9. Access the EMS GUI using the VM IP address or FQDN. See Starting FortiClient EMS and logging in.

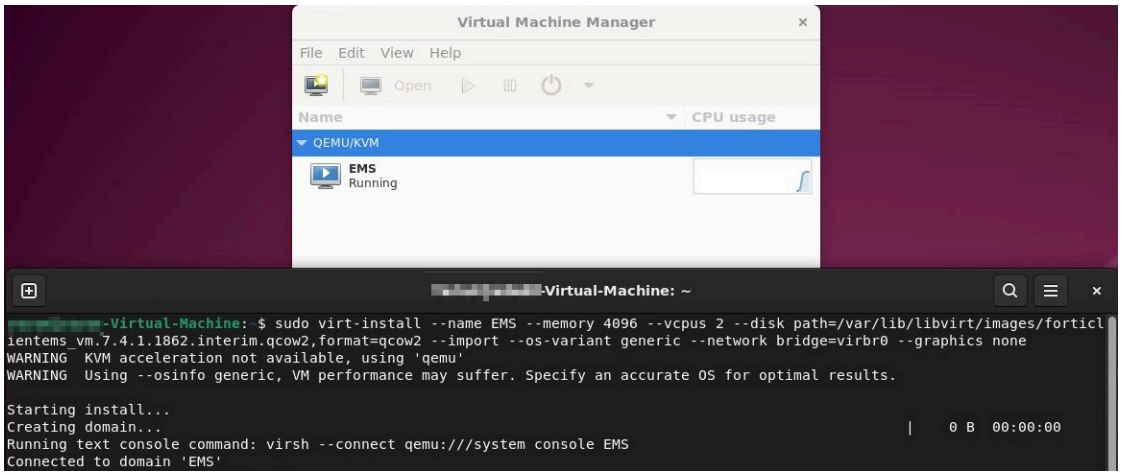
KVM

To deploy EMS on KVM:

1. Set up QEMU or KVM on a Linux host.
2. Copy the `forticlientems_vm qcow2` image under `/var/lib/libvirt/images/`.
3. Run the following command to initialize the virtual machine with the FortiClient EMS image:

```
sudo virt-install --name EMS_VM --memory 4096 --vcpus 2 --disk
path=/var/lib/libvirt/images/forticlientems_vm.7.4.6.2170.M.qcow2,format=qcow2 --import --os-
variant generic --network bridge=virbr0 --graphics none
```

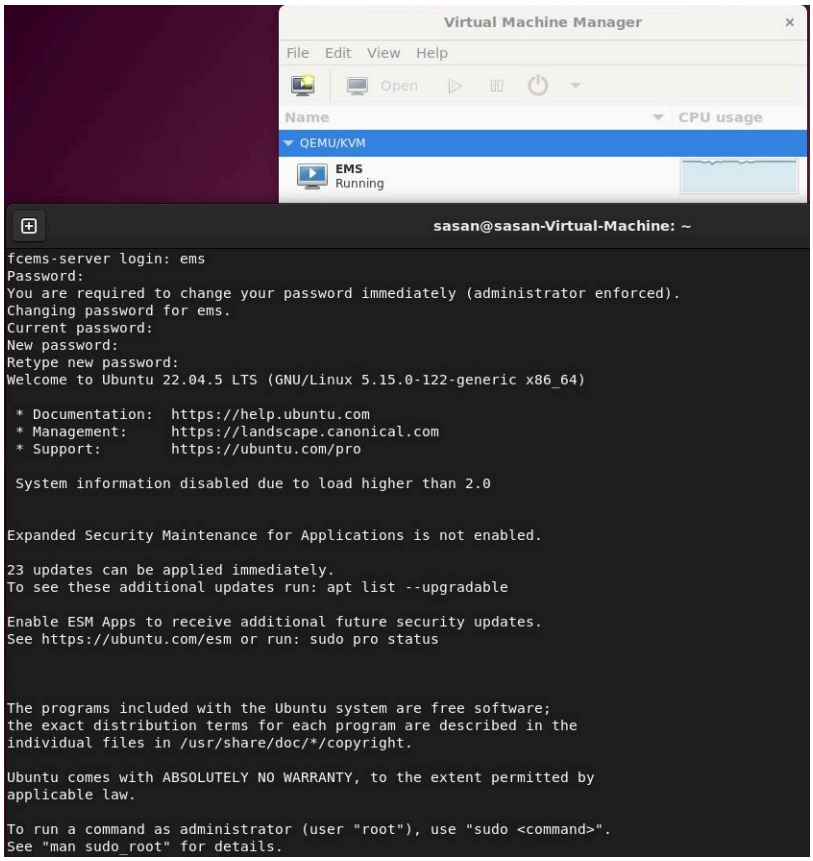
You can change the configuration in the command as needed.



4. Log in to the VM.

The default credentials are:

- Username: `ems`
- Password: `ems`



5. Access the EMS GUI using the VM IP address or FQDN. See Starting FortiClient EMS and logging in.

Proxmox

To deploy EMS on Proxmox:

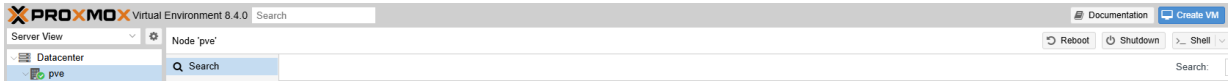
1. Download the EMS KVM image files and copy them to Proxmox:
   a. Log in to the Fortinet Support Portal and select *Support > Firmware Download* from the top menu.
   b. From the *Select Product* dropdown list, select *FortiClientEMS*.
   c. On the *Download* tab, go to *v.700 > 7.4 > 7.4.6*.
   d. Download the qcow2.zip file and extract it.
   e. Copy `forticlientems_vm.7.4.6.2170.M.qcow2` onto a Proxmox node. Typically, you can use the secure copy protocol (SCP). The following shows an example of using `scp forticlientems_vm.7.4.6.2170.M.qcow2 root@proxmox.test.local:/root/ forticlientems_vm.7.4.6.2170.M.qcow2` to copy the `forticlientems_vm.7.4.6.2170.M.qcow2` file to the Proxmox node at `proxmox.test.local`. Edit the node address, which may be an IP address or FQDN, to match your Proxmox environment. The example also assumes that you are running the command as the root user and copying the file to the root user home directory at `/root`:

```
C:\Users\adm\Downloads\forticlientems_vm.7.4.6.2170.M.qcow2>scp forticlientems_
vm.7.4.6.2170.M.qcow2 root@proxmox.test.local:/root/forticlientems_vm.7.4.6.2170.M.qcow2
root@192.168.1.2's password:
```

```
forticlientems_vm.7.4.6.2170.M.qcow2
11% 1134MB  89.8MB/s   01:35 ETA
```

You can get a qcow2 image onto a Proxmox node in various ways. The example uses SCP on the command line. You can use a GUI SCP client, such as WinSCP on Windows or ForkLift on macOS. If you have file transfer protocol set up on your Proxmox node, you can use that as well. Use the method that you are comfortable with.

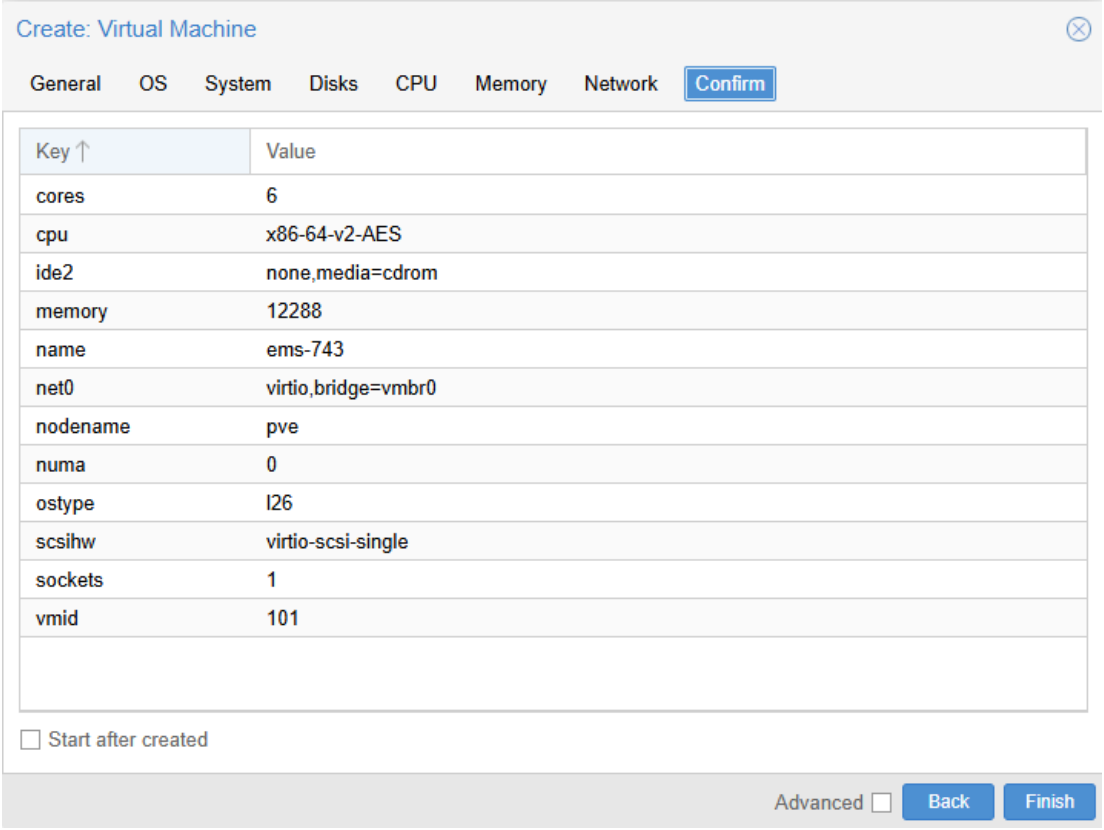
2. Deploy the EMS VM into Proxmox:
   a. In the Proxmox GUI, select the node you copied the EMS image to and click *Create VM*.



   b. In the *Create: Virtual Machine dialog*, change the *VM ID* value if desired. Note this ID value as you use it later. In the *Name* field, give the VM a useful name. Click *Next*.

You may find it useful to add the EMS version number to the end of your VM name.

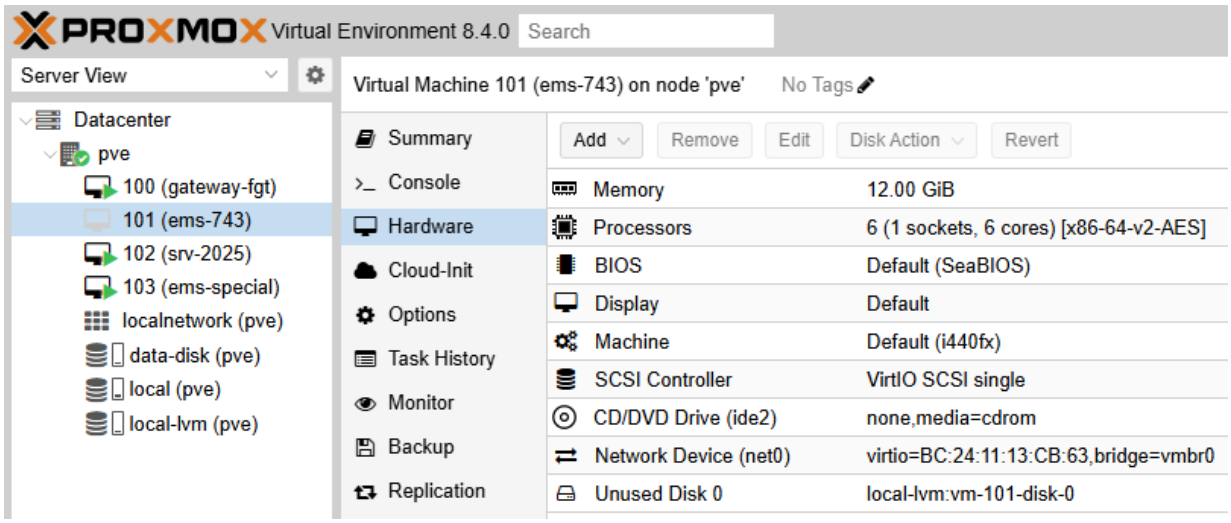
   c. In the *OS* tab, select *Do not use any media*. Leave *Type* and *Version* at their defaults of *Linux* and *6.x - 2.6 Kernel*, respectively. Click *Next*.
   d. In the *System* tab, leave the default values and click *Next*.
   e. In the *Disks* tab, by default, you see an entry for one SCSI disk named *scsi0*. Click the trashcan icon to delete this disk. You see *No Disks* displayed. Click *Next*.
   f. Set the *CPU* and *RAM* tab values per the minimum EMS system requirements. See Management capacity.
   g. In the *Network* tab, deselect *Firewall* and leave the rest of the options at their defaults. Click *Next*.
   h. In the *Confirm* tab, ensure *Start after created* is not selected. Note the *vmid* value as you use it later. Click *Finish* to build the VM.



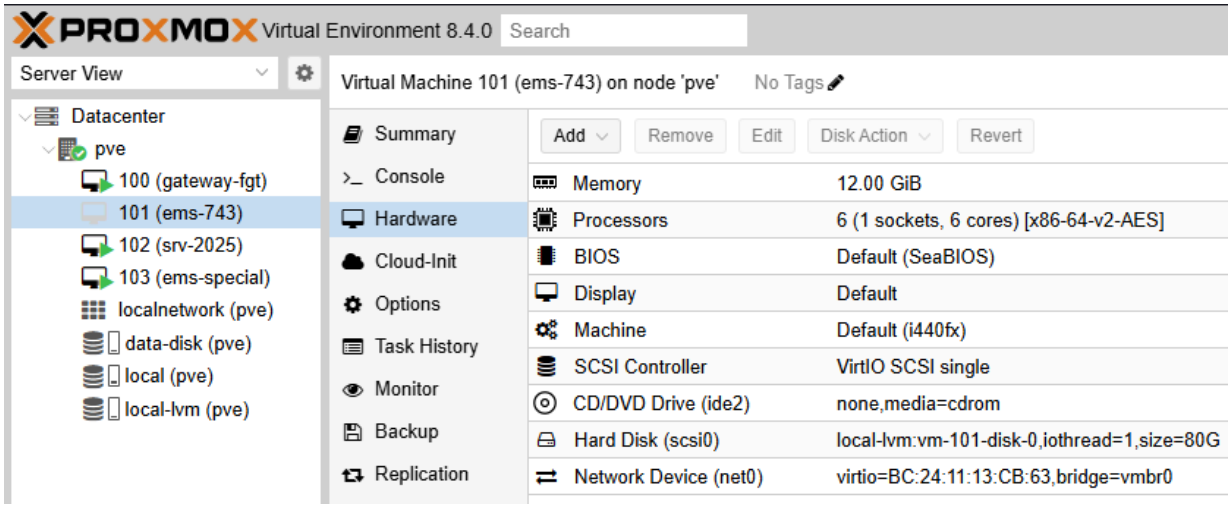
3. Import the qcow2 image into the EMS VM:
   a. After some seconds, you see the new VM in the left sidebar with the configured VM ID and name. Select the newly created VM and click *Hardware* in the middle pane. Note the presence of one network interface named *net0* and the lack of disks.
   b. Select the Proxmox node in the left sidebar and click *Shell* entry in the middle pane. After the shell appears, type `pwd` to ensure you are in the `/root` folder and then type `ls` to display the directory contents. You see the `forticlientems.qcow2` image file copied over earlier.
   c. To import the `forticlientems_vm.7.4.6.2170.M.qcow2` image into your newly created VM, use the `qm disk import` command: `qm disk import <vmid> forticlientems_vm.7.4.6.2170.M.qcow2 <storage device name>`. You must adjust the command to match your configured VM ID and the storage device name of choice. By default, Proxmox creates a local and local-lvm storage device when it is installed. The example uses a VM ID of 101 and the local-lvm storage device. Note the disk name when the command finishes. In the example, the disk name is `local-lvm:vm-101-disk-0'`. See the following example:

```
root@pve:~# qm disk import 101 forticlientems_vm.7.4.3.1926.qcow2 local-lvm
importing disk 'forticlientems_vm.7.4.3.1926.qcow2' to VM 101 ...
  Logical volume "vm-101-disk-0" created.
transferred 0.0 B of 80.0 GiB (0.00%)
transferred 827.4 MiB of 80.0 GiB (1.01%)
[…]
transferred 80.0 GiB of 80.0 GiB (100.00%)
unused0: successfully imported disk 'local-lvm:vm-101-disk-0'
```

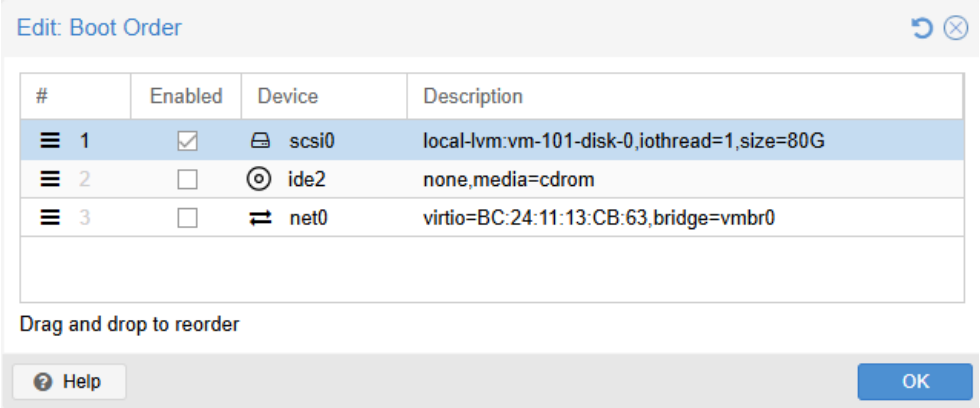
   d. Select the EMS VM in the left sidebar and click *Hardware* in the middle pane. Note the newly imported disk. At this point, it shows as *Unused Disk 0*.



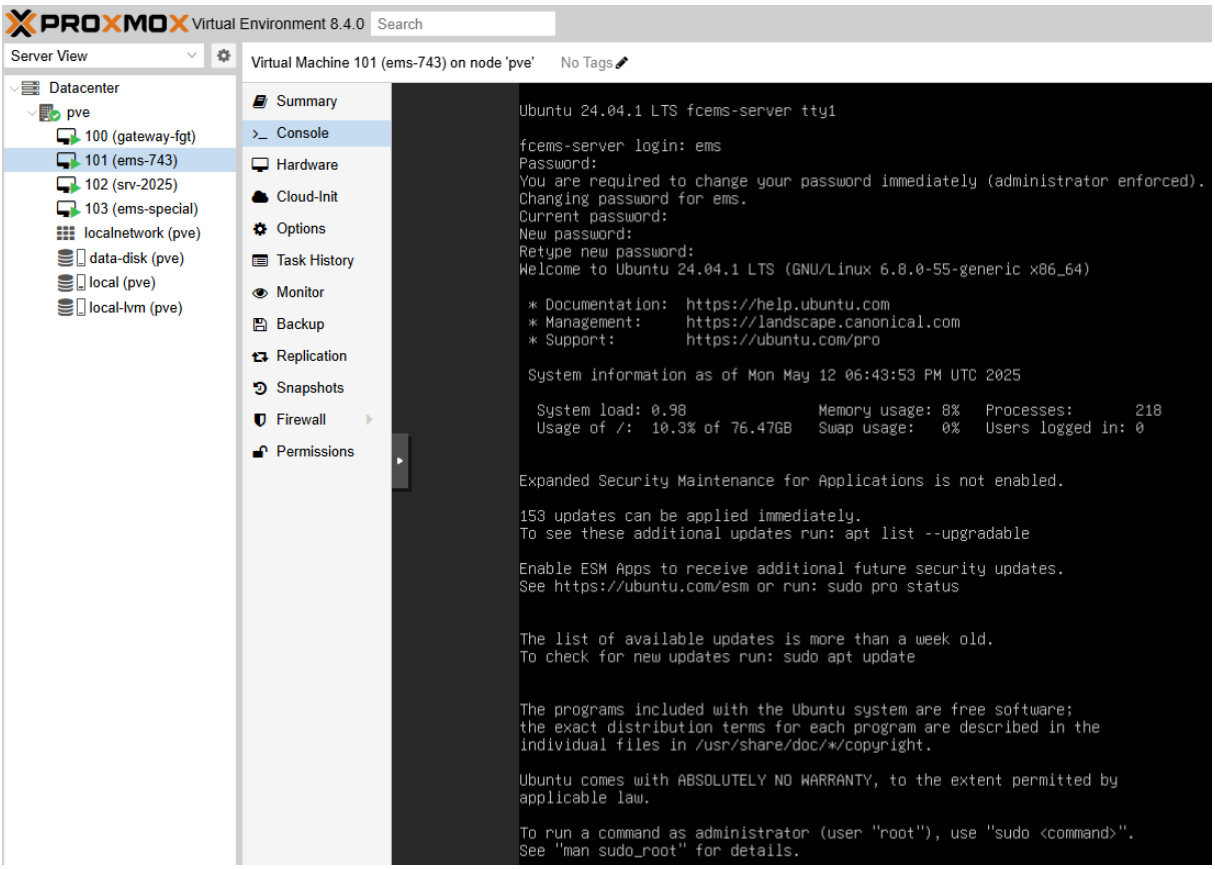
4. Add a boot disk to the EMS VM:
   a. Select *Unused Disk 0* and click *Edit*.
   b. The *Add: Unused Disk* dialog appears. Accept the defaults and click *Add*. Note the newly added *Hard Disk (scsi0)* is now mapped to the `local-lvm:vm-107-disk-0` created earlier.



5. Verify the boot order:
   a. To verify the boot order, select the EMS VM in the left sidebar and click *Options* in the middle pane. Select *Boot Order* and click *Edit*. Alternately, you can double-click *Boot Order*.
   b. The *Edit: Boot Order* dialog appears. Use the selector icons to drag and drop *scsi0* to the top of the list and ensure *Enabled* for that entry is selected.



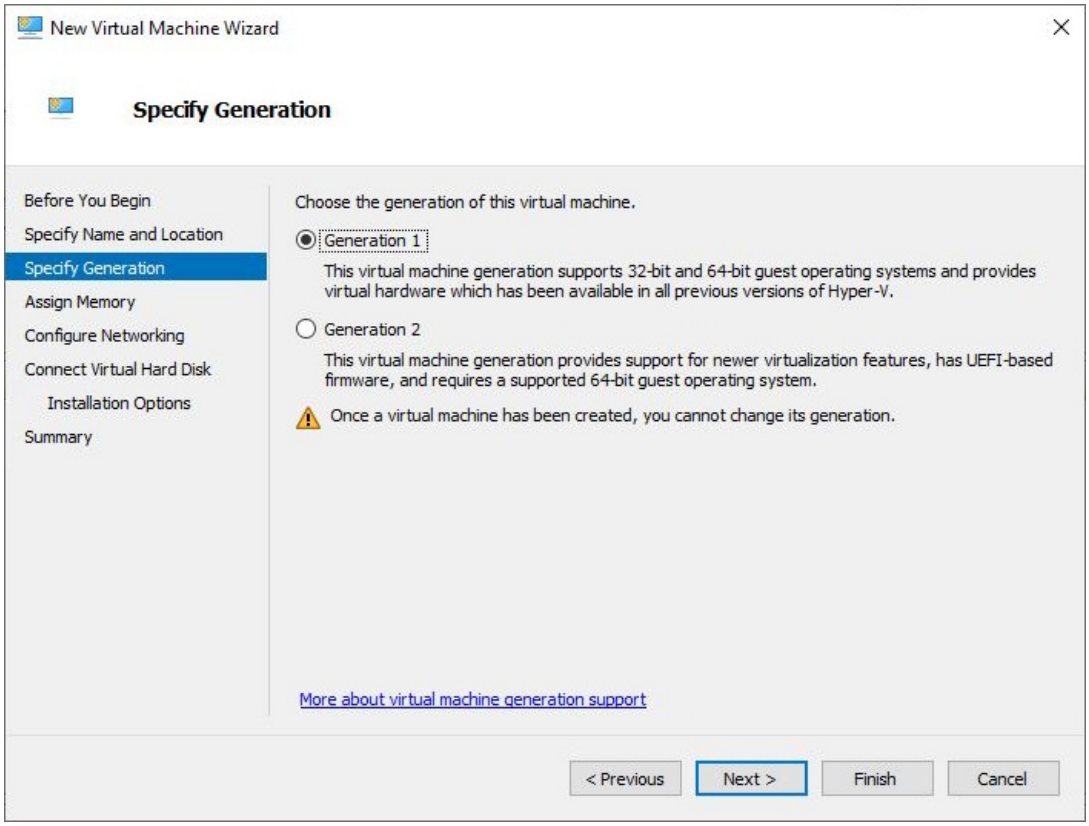
6. Select the EMS VM in the left sidebar and select *Console* in the middle pane. Click *Start* at the top or *Start Now* in the middle of the console.
7. The EMS VM starts to boot. After the reboot, the standard EMS login prompt displays. Log in with a username of ems and password ems. You are prompted to change the password and presented with the EMS shell prompt.



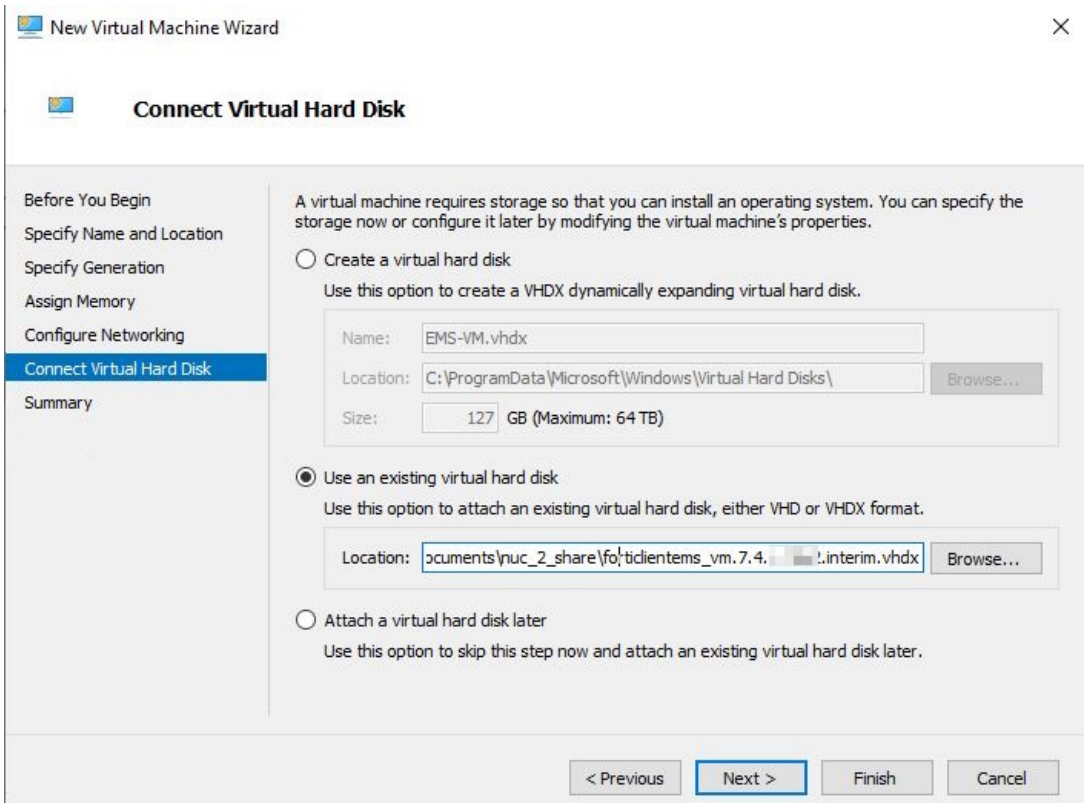
Hyper-V

To deploy EMS on Hyper-V:

1. Open the new VM wizard and define a name for the VM. Click *Next*.
2. In *Specify Generation*, select *Generation 1*. Click *Next*.



3. In *Assign Memory*, configure the required and memory settings.
4. In *Connect Virtual Hard Disk*, select *Use an existing virtual hard disk*. In *Location*, browse to and select the `forticlientems_vm.7.4.6.2170.M.vhdx` file. Click *Finish*. If desired, you can edit the configuration after the VM is created, such as modifying the number of virtual processors.

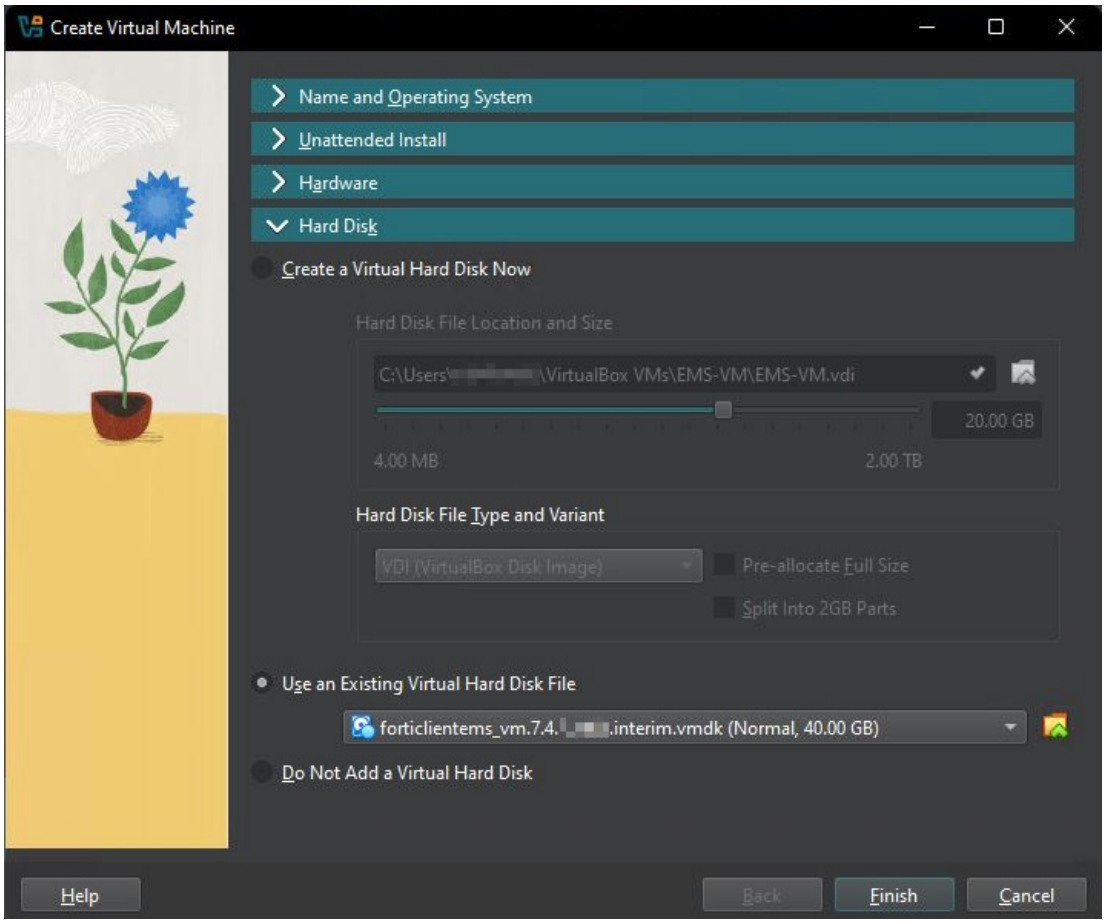


5. Start the VM.
6. After bootup, log in using the default credentials:
   - Username: ems
   - Password: ems

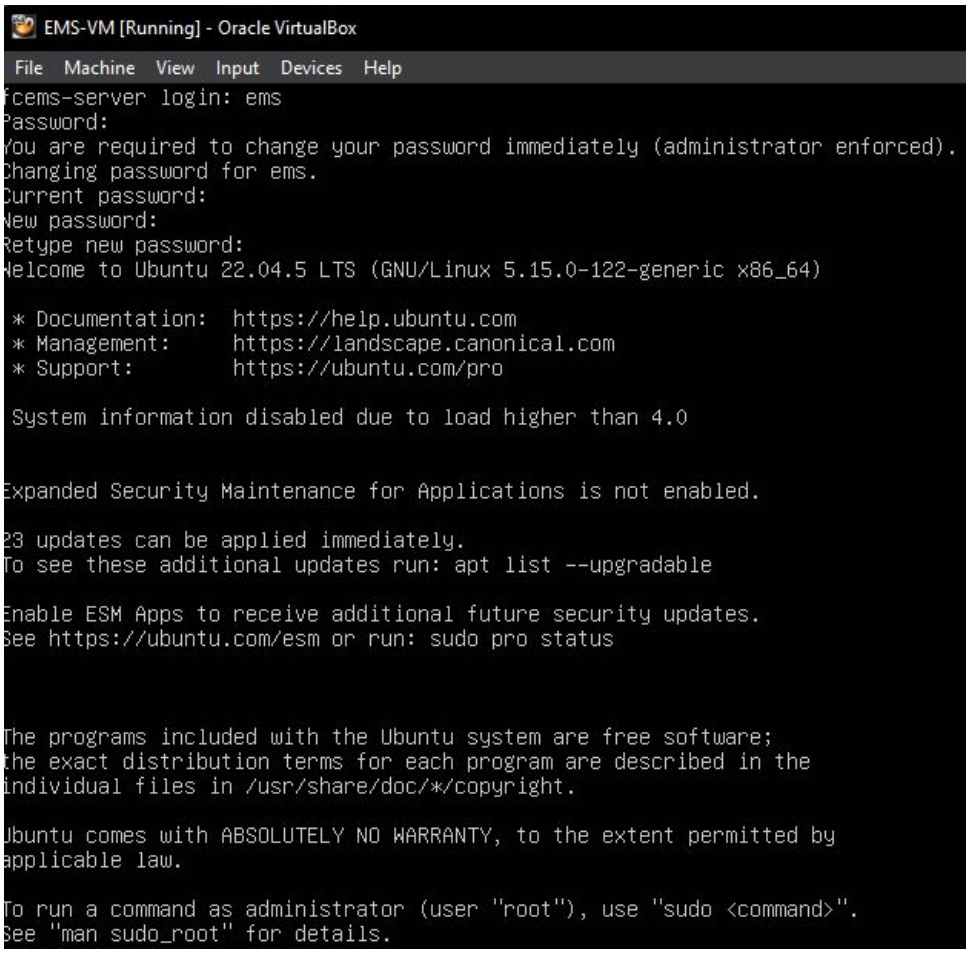
VirtualBox

To deploy EMS on VirtualBox:

1. Open the new VM wizard and define a name for the VM. Click *Next*.
2. In *Hardware*, configure the required hardware settings.
3. In *Hard Disk*, select *Use an Existing Virtual Hard Disk File*. Browse to and select the `forticlientems_vm.7.4.6.2170.M.vmdk` file. Click *Finish*. If desired, you can edit the configuration after the VM is created, such as modifying the number of virtual processors.



4. Start the VM.
5. After bootup, log in using the default credentials:
   - Username: ems
   - Password: ems



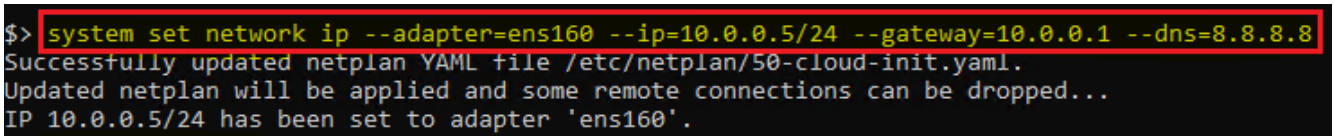
Configuring the IP address

To configure the IP address after deploying EMS as a VM:

```
system set network ip --adapter=<adapter name> --ip=<IP/subnet> --gateway=<gateway IP> --dns=<dns address>
```

For example:

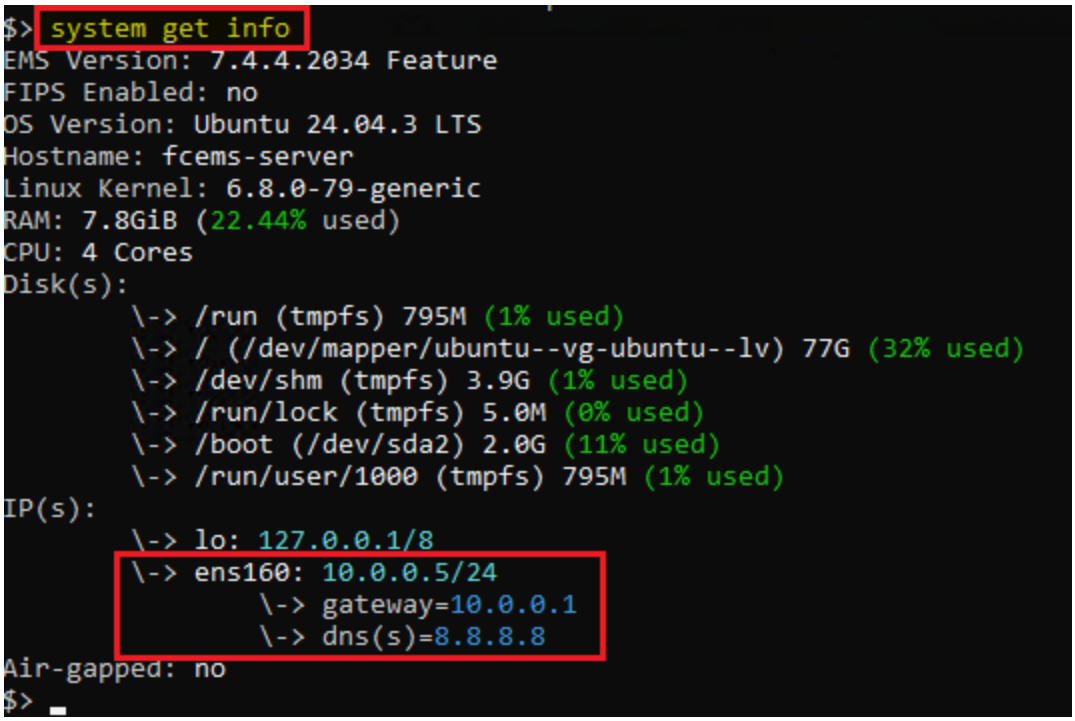
```
system set network ip --adapter=ens160 --ip=10.0.0.5/24 --gateway=10.0.0.1 --dns=8.8.8.8
```



See `system set network ip` for more information.

To verify the IP configuration:

```
system get info
```



Configuring the search domain

You can configure the search domain to allow access to local DNS records for the EMS server. To do so:

```
system set network domain
```

See `system set network domain` for more information.

Upgrading OS packages

You may want to upgrade the underlying operating system packages (such as OS libraries, security patches, and system dependencies) used by EMS VM in the following scenarios:

- Vulnerability scanners or internal security assessments identify OS-level CVEs on the EMS host.
- Fortinet Technical Support advises updating OS packages to resolve a known issue or vulnerability.
- OS package upgrade is require as part of regular system maintenance or patch management cycles where changes are validated and controlled.

OS package upgrades are not expected to affect EMS functionality. However, because EMS relies on underlying OS components, Fortinet cannot guarantee compatibility with all future upstream package changes that are outside EMS release validation. EMS upgrades provided by Fortinet include validated dependencies. However, not all OS-level security patches are bundled with EMS releases.

To upgrade OS packages on the EMS VM, use the `emscli execute upgrade package` command:

- To upgrade a specific package, use the `--package` option followed by the package name. For example:

```
emscli execute upgrade package --package pkgname
```

- To upgrade all underlying operating system packages used by EMS (that are pending upgrade) on your EMS VM:
  a. Perform a full backup of your EMS (both configuration and database).
  b. Schedule a maintenance window to avoid service disruption.
  c. Verify sufficient disk space is available.
  d. Run the following command:

```
emscli execute upgrade package --all
```

  e. After the OS upgrade, validate all EMS services, such as GUI access, endpoint connectivity, services, etc.

Recovering FortiClient EMS VM password

FortiClient EMS VM users can recover the EMS VMs if they forget the password for the "ems" user. See Appendix C - FortiClient EMS VM password recovery for detailed instructions about EMS VM password recovery on different platforms.

Adding and expanding disks on EMS VMs

By default, EMS VMs includes the minimum disk size required for EMS startup. You can provision more disk space during or after the EMS VM deployment process using the management tools of the virtualization platform by following the virtualization platform-specific documentation.

Once a new disk is added to your VM, you can then prepare the disk to be used by the EMS VM using the `emscli lvm` commands.

- `execute lvm add-disk` - adds a physical disk to the logical volume
- `execute lvm expand-disk` - expands the physical disk partition
- `execute lvm expand-volume` - expands the disk logical volume size
- `execute lvm info` - shows info from the disks and logical volume

Example:

The following shows an example of the full workflow of adding physical disk to the EMS VM:

1. Check the current status of the system (using the `system get info` command):

```
system get info
```

Example output:

```
ems@fcems-server $> system get info
EMS Version: 7.4.5.2085 Interim
FIPS Enabled: no
OS Version: Ubuntu 24.04.3 LTS
Hostname: fcems-server
Linux Kernel: 6.8.0-87-generic
RAM: 18GiB (15.67% used)
CPU: 8 Cores
Disk(s):
 \-> /run (tmpfs) 191M (1% used)
 \-> / (/dev/mapper/ubuntu--vg-ubuntu--lv) 77G (19% used)
 \-> /dev/shm (tmpfs) 951M (1% used)
 \-> /run/lock (tmpfs) 5.0M (0% used)
 \-> /boot (/dev/sda2) 2.0G (6% used)
 \-> /run/user/1000 (tmpfs) 191M (1% used)
IP(s):
 \-> lo: 127.0.0.1/8
 \-> eth0: 172.21.161.250/20
Air-gapped: no
```

The example shows a disk "`/dev/mapper/ubuntu--vg-ubuntu--lv`" with 77 GB of capacity where 19% is in use. This is the main part of the volume of the VM and is used by the EMS application. It also shows some other disks and volumes with size and usage information, such as the boot disk : "`/boot (/dev/sda2) 2.0G (6% used)`".

2. Check the current status of disks and logical volumes in the EMS VM (using the `execute lvm info` command):

```
execute lvm info
```

Example output:

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Volume:
 \->Allocated=78.0G,
 \->Free=0.0G
```

The example shows that the main disk is identified as "`sda`" with a total physical size of 78 GB. The physical disk is completely partitioned and fully allocated to the EMS application volume with no expansion possibility (Free=0.0G).

3. Add 10 GB physical disk to the VM using the virtualization platform management tools by following the platform-specific documentation.
4. Check the current status of disks and logical volumes in the EMS VM again:

```
execute lvm info
```

Example output:

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=10.0G,
 \->Partitioned=0.0G-Disk should be added to logical volume.
Disk: sdb,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Volume:
 \->Allocated=78.0G,
 \->Free=0.0G
```

Note that the disk previously identified as "sda" is renamed "sdb" while the new disk is identified as "sda" with a total size of 10 GB that is not partitioned to the VM logical volume yet.

5. Add the "sda" physical size (10 GB) to the logical volume using the command `execute lvm add-disk`:

```
execute lvm add-disk --disk.name sda
```

6. Verify that the volume has grown from 78 GB to 88 GB:

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=10.0G,
 \->Partitioned=10.0G
Disk: sdb,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Volume:
 \->Allocated=88.0G,
 \->Free=0.0G
```

7. Check the current status of the system again and verify that more disk space is available to be used by the EMS application:

```
ems@fcems-server $> system get info
EMS Version: 7.4.5.2085 Interim
FIPS Enabled: no
OS Version: Ubuntu 24.04.3 LTS
Hostname: fcems-server
Linux Kernel: 6.8.0-87-generic
RAM: 19GiB (11.61% used)
CPU: 8 Cores
Disk(s):
 \-> /run (tmpfs) 191M (1% used)
 \-> / (/dev/mapper/ubuntu--vg-ubuntu--lv) 87G (15% used)
 \-> /dev/shm (tmpfs) 951M (1% used)
```

```
 \-> /run/lock (tmpfs) 5.0M (0% used)
 \-> /boot (/dev/sda2) 2.0G (6% used)
 \-> /run/user/1000 (tmpfs) 191M (1% used)
IP(s):
 \-> lo: 127.0.0.1/8
 \-> eth0: 172.21.161.250/20
Air-gapped: no
```

8. Expand the size of the newly-added disk "sda" from 10 GB to 30 GB using the virtualization platform management tools by following the platform-specific documentation.
9. Check the current status of disks and logical volumes in the EMS VM again:

```
execute lvm info
```

Example output:

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Disk: sdb,
 \->Total Size=30.0G,
 \->Partitioned=10.0G-Disk can be expanded.
Volume:
 \->Allocated=88.0G,
 \->Free=0.0G
```

Verify that 20 GB has been added on top of the 10 GB disk already in use and the total capacity of the physical disk "sdb" is now 30 GB.

10. Expand the disk to partition the disk size not yet in use (using the `execute lvm expand-disk` command):

```
execute lvm expand-disk --disk.name sdb
```

11. Verify that all space on physical disk "sdb" has been partitioned (30.0 GB):

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=30.0G,
 \->Partitioned=30.0G
Disk: sdb,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Volume:
 \->Allocated=88.0G,
 \->Free=20.0G-Volume can be expanded.
```

Note that the expanded 20 GB is not yet allocated in the EMS VM and not ready to be used by the EMS application.

12. Allocate the newly-partitioned volume (20 GB) to the EMS VM (so that it can be used by the EMS application) using the `execute lvm expand-volume` command. You can allocate the size partially or fully by expanding the logical volume size of the disk by a specific size or by using all available free space in the volume group.

Try expanding the volume by 5 GB (of the 20 GB free space):

```
execute lvm expand-volume --grow.gb 5.0
```

13. Verify that the volume available to be used by the EMS application has grown from 88 GB to 93 GB and 15.0 GB volume is still available to be allocated:

```
ems@fcems-server $> system get info
EMS Version: 7.4.5.2085 Interim
FIPS Enabled: no
OS Version: Ubuntu 24.04.3 LTS
Hostname: fcems-server
Linux Kernel: 6.8.0-87-generic
RAM: 20GiB (12.74% used)
CPU: 8 Cores
Disk(s):
 \-> /run (tmpfs) 191M (1% used)
 \-> / (/dev/mapper/ubuntu--vg-ubuntu--lv) 92G (14% used)
 \-> /dev/shm (tmpfs) 951M (1% used)
 \-> /run/lock (tmpfs) 5.0M (0% used)
 \-> /boot (/dev/sda2) 2.0G (6% used)
 \-> /run/user/1000 (tmpfs) 191M (1% used)
IP(s):
 \-> lo: 127.0.0.1/8
 \-> eth0: 172.21.161.250/20
Air-gapped: no
```

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Disk: sdb,
 \->Total Size=30.0G,
 \->Partitioned=30.0G
Volume:
 \->Allocated=93.0G,
 \->Free=15.0G-Volume can be expanded.
```

14. Try expanding the logical volume using all available free space (15 GB) in the volume group:

```
ems@fcems-server $> execute lvm expand-volume --grow.free
```

15. Verify that the volume available to be used by the EMS application has grown from 93 GB to 108 GB with no volume available to be allocated:

```
ems@fcems-server $> system get info
EMS Version: 7.4.5.2085 Interim
FIPS Enabled: no
OS Version: Ubuntu 24.04.3 LTS
Hostname: fcems-server
Linux Kernel: 6.8.0-87-generic
RAM: 20GiB (12.81% used)
CPU: 8 Cores
```

```
Disk(s):
 \-> /run (tmpfs) 191M (1% used)
 \-> / (/dev/mapper/ubuntu--vg-ubuntu--lv) 106G (13% used)
 \-> /dev/shm (tmpfs) 951M (1% used)
 \-> /run/lock (tmpfs) 5.0M (0% used)
 \-> /boot (/dev/sda2) 2.0G (6% used)
 \-> /run/user/1000 (tmpfs) 191M (1% used)
IP(s):
 \-> lo: 127.0.0.1/8
 \-> eth0: 172.21.161.250/20
Air-gapped: no
```

```
ems@fcems-server $> execute lvm info
Disk: sda,
 \->Total Size=78.0G,
 \->Partitioned=78.0G
Disk: sdb,
 \->Total Size=30.0G,
 \->Partitioned=30.0G
Volume:
 \->Allocated=108.0G,
 \->Free=0.0G
```

Deploying EMS in air-gapped environments

You can install or upgrade EMS in an environment without direct connectivity to the Internet with the following options:

- Air-gapped install or upgrade with EMS Docker containers using Docker Compose or Kubernetes
- Air-gapped install or upgrade with a dependencies bundle
- Air-gapped install or upgrade using an HTTP proxy

EMS 7.4.6 only supports x64 architecture environments for air-gapped deployment. ARM is not currently supported.

Air-gapped install or upgrade with EMS docker containers

As the package that provides EMS containers is already offline, installing or upgrading with EMS docker containers does not require any external connectivity as long as the Docker or Kubernetes tools are available in the target environment . EMS containers can be deployed using Docker Compose (podman is also supported with the compose plugin) or Kubernetes. See Deploying EMS with Docker Compose on page 42 and Deploying EMS on Kubernetes on page 48.

Air-gapped install or upgrade with a dependencies bundle

EMS 7.4.6+ includes an additional package with the required dependencies to allow installing EMS without active Internet connection.

To install EMS in an air-gapped environment with a dependencies bundle on Ubuntu 22, Ubuntu 24, or Red Hat 9:

1. Download one of the following packages (depending on your EMS platform) along with the EMS installer from the Fortinet support site:
   - **(Ubuntu 22 AMD64)** `forticlientems_7.4.6.2170.M.dependencies_ubuntu22_amd64.tar.gz`
   - **(Ubuntu 24 AMD64)** `forticlientems_7.4.6.2170.M.dependencies_ubuntu24_amd64.tar.gz`
   - **(RHEL 9 AMD64)** `forticlientems_7.4.6.2170.M.dependencies_rhel9_amd64.tar.gz`
2. Specify the location of the dependencies package using the `--offline_bundle` parameter during EMS installation.
   - **Ubuntu 22**:

     ```
     ./forticlientems_7.4.6.2170.M.amd64.bin -- --allowed_hosts '*' --enable_remote_https --
     offline_bundle ./forticlientems_7.4.6.2170.M.dependencies_ubuntu22_amd64.tar.gz
     ```

   - **Ubuntu 24**:

     ```
     ./forticlientems_7.4.6.2170.M.amd64.bin -- --allowed_hosts '*' --enable_remote_https --
     offline_bundle ./forticlientems_7.4.6.2170.M.dependencies_ubuntu24_amd64.tar.gz
     ```

   - **RHEL**:

     ```
     ./forticlientems_7.4.6.2170.M.amd64.bin -- --allowed_hosts '*' --enable_remote_https --
     offline_bundle ./forticlientems_7.4.6.2170.M.dependencies_rhel9_amd64.tar.gz
     ```

- Do not change the extension of the dependencies package. The EMS installer will only be able to work with `tar.gz`.
- The dependencies for each OS will be up-to-date up to the time of the release. After that, they are still reliable to be used but will not contain the latest updates, if any, even if they have vulnerabilities. Since this is an airgapped environment, there' is no way to pull updates for those dependencies and you will have to wait for the new release of EMS to get an updated package.

To upgrade an existing EMS in an air-gapped environment with a dependencies bundle on Ubuntu 22, Ubuntu 24, or Red Hat 9:

1. Download the target version of the following packages (depending on your EMS platform) along with the target version of the EMS installer from the Fortinet support site:
   - **(Ubuntu 22 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_ubuntu22_amd64.tar.gz`
   - **(Ubuntu 24 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_ubuntu24_amd64.tar.gz`
   - **(RHEL 9 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_rhel9_amd64.tar.gz`

2. Specify the location of the new dependencies package using the `--offline_bundle` parameter during EMS upgrade. Only the `--offline_bundle` arguments is passed this time because all other parameters refer to configuration that rarely or never change and the installer is able to keep them as previously defined
   - **Ubuntu 22**:

     ```
     ./forticlientems_x.x.x.XXXX.amd64.bin --offline_bundle ./forticlientems_
     7.4.6.2170.M.dependencies_ubuntu22_amd64.tar.gz
     ```

   - **Ubuntu 24**:

     ```
     ./forticlientems_x.x.x.XXXX.amd64.bin --offline_bundle ./forticlientems_
     x.x.x.XXXX.dependencies_ubuntu24_amd64.tar.gz
     ```

   - **RHEL**:

     ```
     ./forticlientems_x.x.x.XXXX.amd64.bin -- --allowed_hosts '*' --offline_bundle
     ./forticlientems_x.x.x.XXXX.dependencies_rhel9_amd64.tar.gz
     ```

To upgrade air-gapped EMS VMs:

EMS Virtual Appliances do not require Internet connectivity to function as they come pre-installed and pre-setup.

1. Download the target version of the following packages (depending on your EMS VM platform) along with the target version of the EMS installer from the Fortinet support site:
   - **(Ubuntu 22 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_ubuntu22_amd64.tar.gz`
   - **(Ubuntu 24 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_ubuntu24_amd64.tar.gz`
   - **(RHEL 9 AMD64)** `forticlientems_x.x.x.xxxx.dependencies_rhel9_amd64.tar.gz`
2. Copy the dependencies package into the EMS VM using the following command:

   ```
   ems@fcems-server $> execute scp --local.file dependencies.tar.gz --remote.host <name or ip of
   the server that hosts it> --remote.user <user with access to the server and fle> --remote.file
   <full qualified path of the file on the remote server>
   ```

   For example:

   ```
   ems@fcems-server $> execute scp --local.file dependencies.tar.gz --remote.host 10.0.0.5 --
   remote.user myuser --remote.file /home/myuser/forticlientems_x.x.x.XXXX.interim_dependencies_
   ubuntu24_amd64.tar.gz
   ```

3. Upgrade the EMS VM using the following command:

   ```
   ems@fcems-server $> execute upgrade ems --offline.bundle dependencies.tar.gz --local.file
   forticlientems_x.x.x.XXXX.amd64.bin
   ```

Air-gapped install or upgrade using an HTTP proxy

Some air-gapped environments are not directly connected to the Internet but may do so through a proxy for important services. EMS 7.4.6+ supports installation upgrade using an HTTP proxy where the proxy settings are passed to the EMS installer which then pulls dependencies from the Internet through the provided proxy.

To install or upgrade EMS using an HTTP proxy on Linux:

Use the following installer parameters:

- **`--proxy_host`**: Host of the HTTP proxy in the following format: `http://<host>[:<port>]/`, e.g., `http://myproxy.comp.com/`
- **`--proxy_user`**: **(Optional)** User name to authenticate in the proxy. Only required if the HTTP proxy requires authentication.
- **`--proxy_pwd`**: **(Optional)** Password to authenticate in the proxy. Only required if the HTTP proxy requires authentication.

For example:

```
./forticlientems_7.4.6.2170.M.interim.amd64.bin -- --allowed_hosts '*' --enable_remote_https --
proxy_host http://myproxy.comp.com/ --proxy_user myuser --proxy_pwd mypassword
```

If EMS was previously installed used the proxy parameters and the parameters remain valid, they will be recovered from the previous install and you do not need to provide them to the installer again during an upgrade.

To restrict the proxy access to only the domains that EMS uses to pull dependencies:

See the following required domain list for different operating systems:

Ubuntu 22 or 24	RHEL/Rocky Linux 9
• archive.ubuntu.com • security.ubuntu.com • apt.postgresql.org • www.postgresql.org • packages.redis.io • ppa.launchpad.net • api.launchpad.net • dl.winehq.org	• download.rockylinux.org • mirror.centos.org • mirror.stream.centos.org • dl.fedoraproject.org • mirrors.fedoraproject.org • download.postgresql.org • mirrors.rpmfusion.org • download1.rpmfusion.org • rpms.remirepo.net • cdn.remirepo.net • cpanmin.us • cpan.metacpan.org • cdn.redhat.com

To install or upgrade EMS using an HTTP proxy on VM:

Run the following command:

```
system set proxy --url http://myproxy.comp.com/ --user myuser --password mypassword
```

- **`--url`**: Host of the HTTP proxy in the following format: `http://<host>[:<port>]/`, e.g., `http://myproxy.comp.com/`
- **`--user`**: **(Optional)** User name to authenticate in the proxy. Only required if the HTTP proxy requires authentication.
- **`--password`**: **(Optional)** Password to authenticate in the proxy. Only required if the HTTP proxy requires authentication.

Using the HTTP proxy parameters will persist the proxy configuration in the environment, which means everything that runs in the environment and requires Internet access will hit the proxy for such access. If this behavior is not desired, you can remove the HTTP proxy settings from the environment after the EMS installation or upgrade using the EMS CLI:

- **Linux:** `emscli system unset proxy`
- **VM:** `system unset proxy`

To set up a minimalistic HTTP proxy server with Squid:

Fortinet does not maintain nor provides support for Squid. It is an open source tool and should only be used if you need to install or upgrade EMS using this route and lack a production grade HTTP proxy to be used instead.

1. Create a custom docker image on a machine with Internet connection:
   a. Create a file called `Dockerfile` with the following content:

```
FROM alpine:latest
RUN apk add --no-cache squid apache2-utils
RUN mkdir -p /var/cache/squid /var/log/squid && \
chown -R squid:squid /var/cache/squid /var/log/squid
COPY squid.conf /etc/squid/squid.conf
# passwd file will be created at runtime
RUN touch /etc/squid/passwd && \
chown squid:squid /etc/squid/passwd && \
chmod 640 /etc/squid/passwd
EXPOSE 3128
CMD ["squid", "-N", "-d", "1"]
```

   b. Create a file called `squid.conf` with squid's basic configuration:

```
Squid listens on port 3128
http_port 3128
# Authentication settings
auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwd
auth_param basic realm Proxy
auth_param basic credentialsttl 2 hours
# ACLs
acl authenticated proxy_auth REQUIRED
# Allow authenticated users
```

```
http_access allow authenticated
# Deny everything else
http_access deny all
# Logging
access_log stdio:/var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log none
# Cache directory
#cache_dir ufs /var/cache/squid 100 16 256
cache deny all
cache_dir null /var/cache/squid
```

If you want to restrict Squid to access only domains used by EMS to pull dependencies from, use the following configuration for `squid.conf` instead:

```
Squid listens on port 3128
http_port 3128
# Authentication settings
auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwd
auth_param basic realm Proxy
auth_param basic credentialsttl 2 hours

# ACLs
acl authenticated proxy_auth REQUIRED

# --------------------------------
# Allowed domains (Ubuntu / Debian)
# --------------------------------
acl allowed_domains dstdomain \
.archive.ubuntu.com \
.security.ubuntu.com \
.ppa.launchpad.net \
.api.launchpad.net

# --------------------------------
# PostgreSQL
# --------------------------------
acl allowed_domains dstdomain \
.apt.postgresql.org \
.www.postgresql.org \
.download.postgresql.org

# --------------------------------
# Redis
# --------------------------------
acl allowed_domains dstdomain \
.packages.redis.io

# --------------------------------
# WineHQ
# --------------------------------
acl allowed_domains dstdomain \
```

```
.dl.winehq.org

# ------------------------------
# Rocky / CentOS / Fedora / RPMFusion / Remi
# ------------------------------
acl allowed_domains dstdomain \
.download.rockylinux.org \
.mirror.centos.org \
.mirror.stream.centos.org \
.dl.fedoraproject.org \
.mirrors.fedoraproject.org \
.mirrors.rpmfusion.org \
.download1.rpmfusion.org \
.rpms.remirepo.net \
.cdn.remirepo.net \
.cdn.redhat.com

# ------------------------------
# CPAN / Perl
# ------------------------------
acl allowed_domains dstdomain \
.cpanmin.us \
.cpan.metacpan.org

# ------------------------------
# Allow authenticated users ONLY to allowed domains
# ------------------------------
http_access allow authenticated allowed_domains

# Deny everything else
http_access deny all

# Logging
access_log stdio:/var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log none

# Disable caching (recommended for temporary proxies)
cache deny all
cache_dir null /var/cache/squid
```

c. Build the docker image:

```
docker build -t emssquid .\
```

2. Start the server by running the following (by default it listen on port 3128):

```
docker run -d --name emssquidproxy -p 3128:3128 emssquid
```

3. Create a user and password to access squid using the following command:

```
docker exec -ti emssquidproxy htpasswd /etc/squid/passwd ems
```

This example uses the user `ems` but you are free to choose any username of your choice.

Installation parameters

The following provides a list of parameters that the EMS install commands support:

Parameter	Description
`allowed_hosts`	A comma-separated list (without spaces) of hostnames or IP addresses that the EMS web server will respond to (for example, EMS FQDN or FortiGate VIP address). This setting does not control user access by source IP. It only validates the HTTP host header for security.
`das_cache_engine`	Specify DAS cache engine, such as simple.
`db_host`	Remote database (DB) hostname.
`db_hosts`	A comma-separated list (without spaces) of remote database (DB) hostnames in the following format: `<host>:<port>`. For example, `--db_hosts "db1:5432,db2:5432,db3:5543"`.
`db_pass`	Remote DB password.
`db_port`	Port that remote DB uses to communicate with EMS.
`db_prefix`	Used when FortiClient Cloud uses an external DB. Prepend this value to the DBs that FortiClient Cloud uses to ensure unique DB names.
`db_user`	Remote DB username.
`elastic_api_key`	Having EMS connect to the Elasticsearch (ES) cluster using an API encoded key, rather than username and password, is considered best practice. Provide the encoded version of the key to EMS.
`elastic_ca_path`	If the ES cluster is set up to strictly check for a client certificate, you must provide the CA certificate to EMS. This must be a fully qualified path to the CA certificate file. Store the file in a location outside of `/opt/forticlientems` and be readable by the forticlientems user or group. For example, you may configure the following: `--elastic_ca_path /data/certs/es_ca.crt`
`elastic_hosts`	ES server IP address and port that ES uses to communicate with EMS in <ES server IP address>:<port> format. If you have multiple primary ES server nodes, provide their <IP address>:<port> in a comma-separated list with no whitespace in between. For example, you may configure the following: `elastic_hosts "node1.local:8000,node2.local:8000,node3.local:8000"`
`elastic_password`	Password for ES user created for EMS.

Parameter	Description
elastic_user	Username to use to connect to ES. If you provide both an API key and username and password, EMS prefers to use the API key and ignores the username and password.
enable_event_feature	Enable EMS services required for using the new DB for storing and querying events.
enable_fips	Enable FIPS mode. If this parameter is not specified (default), EMS will work in non-FIPS mode.
enable_remote_https	Enable remote HTTPS access to EMS.
offline_bundle	Install the offline bundle for air-gapped environments.
proxy_host	Host of the HTTP proxy in the following format: http://<host>[:<port>]/, e.g., http://myproxy.comp.com/
proxy_user	User name to authenticate in the proxy. Only required if the HTTP proxy requires authentication.
proxy_pwd	Password to authenticate in the proxy. Only required if the HTTP proxy requires authentication.
fileserver_port	Customize webserver ports.
http_port	
https_port	
internal_db_port	PostgreSQL port to set for executing the DB deployment or upgrade (remote or locally).
is_paas	Only used when using EMS with a cloud DB. This parameters tells the installer that this is a special DB and needs special handling.
redis_cluster_hosts	Redis cluster hosts names.
redis_host	Redis hostname.
redis_password	Redis password.
redis_port	Port that Redis uses to communicate with EMS.
redis_username	Redis username.
scep_public_hostname	Specify the FQDN or hostname accessible by mobile endpoints to pull the ZTNA certificate from the SCEP server running on EMS.
skip_db_deploy	Skip express DB deployment.
skip_db_install	Skip express DB install.
skip_event_feature_local_install_local_install	Skip event worker local installation.

Change log

Date	Change description
2026-03-17	Initial release.
2026-03-18	Updated Deploying EMS with Docker Compose on page 42.

www.fortinet.com

Copyright© 2026 Fortinet, Inc. All rights reserved. Fortinet®, FortiGate®, FortiCare® and FortiGuard®, and certain other marks are registered trademarks of Fortinet, Inc., and other Fortinet names herein may also be registered and/or common law trademarks of Fortinet. All other product or company names may be trademarks of their respective owners. Performance and other metrics contained herein were attained in internal lab tests under ideal conditions, and actual performance and other results may vary. Network variables, different network environments and other conditions may affect performance results. Nothing herein represents any binding commitment by Fortinet, and Fortinet disclaims all warranties, whether express or implied, except to the extent Fortinet enters a binding written contract, signed by Fortinet's Chief Legal Officer, with a purchaser that expressly warrants that the identified product will perform according to certain expressly-identified performance metrics and, in such event, only the specific performance metrics expressly identified in such binding written contract shall be binding on Fortinet. For absolute clarity, any such warranty will be limited to performance in the same ideal conditions as in Fortinet's internal lab tests. Fortinet disclaims in full any covenants, representations, and guarantees pursuant hereto, whether express or implied. Fortinet reserves the right to change, modify, transfer, or otherwise revise this publication without notice, and the most current version of the publication shall be applicable.