

Linux Protection

Contents

Chapter 1: Introduction	4
1.1 System requirements.....	5
1.2 How the product works	6
1.3 Key features and benefits	6
1.4 What is harmful content.....	7
1.4.1 Viruses.....	7
1.4.2 Potentially unwanted applications (PUA) and unwanted applications (UA).....	7
1.4.3 Worms.....	8
1.4.4 Trojans.....	8
1.4.5 Backdoors.....	8
1.4.6 Exploits.....	9
1.4.7 Exploit kits.....	9
1.4.8 Rootkits.....	9
Chapter 2: Installation.....	10
2.1 Installing the product for use with WithSecure Elements Endpoint Protection.....	11
2.1.1 Installing the product using a DEB or RPM package.....	11
2.1.2 Installing the product using a generic installer package.....	12
2.1.3 Delaying product activation.....	13
2.2 Migrating from Business Suite to WithSecure Elements.....	13
2.2.1 Inspecting the migration status.....	13
2.2.2 Migrating Linux Security 64 to WithSecure Elements.....	14
2.2.3 Assigning profile to migrated devices.....	14
2.3 Uninstalling the product.....	14
Chapter 3: Using the product.....	15
3.1 Basics of using the WithSecure Elements Endpoint Protection portal.....	16
3.2 Configuring settings with the lsctl utility.....	16
3.2.1 Setting types and the settings tree.....	17
3.2.2 Operating with settings.....	18
3.2.3 Input and output formats.....	19
3.2.4 Working with arrays.....	20
3.2.5 Loading product settings from a file.....	21
3.2.6 Inspecting the state of the product.....	21
3.2.7 Command-line settings for real-time scanning.....	23
3.2.8 Command-line settings for integrity checking.....	26
3.2.9 Command-line settings for offload scanning.....	27
3.2.10 Command-line settings for automatic updates.....	28
3.3 Malware scanning.....	29

3.3.1 Real-time scanning.....	30
3.3.2 Manual scanning.....	30
3.3.3 Scheduled scanning.....	31
3.4 Integrity checking	31
3.5 Checking client status.....	32
3.6 Automatic updates	33
3.7 Using HTTP proxies.....	33
3.8 Example: managing integrity checker profiles.....	34
3.9 Command line usage.....	35
3.9.1 Starting and stopping services.....	35
3.9.2 Scanning the computer manually from the command line.....	35
3.9.3 Updating the product manually from the command line.....	37
3.9.4 Testing the antivirus protection	37
3.9.5 Running the integrity checker from the command line.....	38
Appendix A: Alert severity levels	39
Appendix B: Services and logs.....	40
B.1 Services installed with the product.....	41
B.2 Logs used by the product.....	42
Appendix C: Cloud services.....	43

Chapter 1

Introduction

Topics:

- [System requirements](#)
- [How the product works](#)
- [Key features and benefits](#)
- [What is harmful content](#)

This product provides an integrated, out-of-the-box security solution with strong real-time protection against viruses and potentially unwanted applications, and it also includes host intrusion prevention (HIPS) functionality that provides protection against unauthorized system modifications, as well as userspace and kernel rootkits.

Computer viruses are one of the most harmful threats to the security of data on computers. While some viruses are harmless pranks, other viruses can destroy data and pose a real threat.

The solution can be deployed and managed using the WithSecure Elements Endpoint Protection portal. It is also possible to run Linux Protection without a connection to the management portal, in which case you can use command line tools to manage the product settings.

1.1 System requirements

This section contains important information about the product.

We strongly recommend that you read the entire document before you start using the product.

Supported platforms

Note: WithSecure supports only operating systems that are currently supported by their vendor. Learn more about WithSecure's operating system support policies at [Product updates and supported versions](#). If you are interested in long-term support for a platform that vendors no longer support, contact your sales representative.

The following 64-bit (AMD64/EM64T) distributions are supported:

- AlmaLinux 8, 9, 10
- Amazon Linux 2, 2023
- Debian 11, 12, 13
- Oracle Linux 8, 9, 10
- RHEL 8, 9, 10
- Rocky Linux 8, 9, 10
- SUSE Linux Enterprise Server 15 (Service Pack 6)
- Ubuntu 20.04, 22.04, 24.04

The following 64-bit (ARMv8/AArch64) distributions are supported:

- AlmaLinux 9, 10
- Amazon Linux 2023
- Debian 11, 12, 13
- Oracle Linux 9, 10
- Red Hat Enterprise Linux 9, 10
- Rocky 9, 10
- Ubuntu 22.04, 24.04

Note: All operating systems must have TLS 1.2 configured and enabled.

Support for SELinux

The product supports Security-Enhanced Linux with the following distributions:

- AlmaLinux 8, 9, 10 *
- Debian 11, 12, 13 **
- Oracle Linux 8, 9, 10 *
- RHEL 8, 9, 10 *
- Rocky Linux 8, 9, 10 *

* on systems running the "targeted" SELinux system policy

** the product is compatible with the "default" SELinux system policy

If the distribution is not supported, SELinux must be disabled.

Note: Major distribution upgrades when the product is installed on a host with SELinux enabled policies are not supported. We recommend uninstalling the product before upgrading the distribution and then reinstalling it on the upgraded system.

System requirements

- Processor: 64-bit processor compatible with the distribution; two or more cores recommended
- Memory: 4 GB RAM

- Disk space: At least 2 GB available (recommended)
- Swap: Sufficient swap memory is highly recommended
- Internet connection: Required to validate your subscription, receive product updates, and use cloud-based detection

1.2 How the product works

The product detects and prevents intrusions and protects against malware.

When user downloads a file from the Internet, for example by clicking a link in an email message, the file is scanned when the user tries to open it. If the file is infected, the product protects the system against the malware.

- **Real-time scanning** gives you continuous protection against viruses and potentially unwanted applications as files are opened, copied, and downloaded from the Web. Real-time scanning functions transparently in the background, looking for viruses whenever you access files on the hard disk, removable media, or network drives. If you try to access an infected file, the real-time protection automatically stops the virus from executing.
- When the real-time scanning has been configured to scan a limited set of files, the **manual scanning** can be used to scan the full system or you can use the scheduled scanning to scan the full system at regular intervals.
- **Automatic Updates** keep the virus definitions always up-to-date. The virus definition databases are updated automatically after the product has been installed. The virus definitions updates are signed by WithSecure.

The Host Intrusion Prevention System (**HIPS**) detects any malicious activity on the host, protecting the system on many levels.

- **Integrity Checking** protects the system against unauthorized modifications. It is based on the concept of a known good configuration - the product should be installed before the computer is connected to the network to guarantee that the system is in a known good configuration.

You can create a baseline of the system files that you want to protect and prevent the use of any modified files for all users.

- If an attacker gains a shell access to the system and tries to add a user account to login to the system later, Host Intrusion Prevention System (**HIPS**) detects modified system files and alerts the administrator.
- If an attacker has gained an access to the system and tries to install a userspace rootkit by replacing various system utilities, **HIPS** detects modified system files and alerts the administrator.

1.3 Key features and benefits

The product offers superior protection against viruses and worms and is transparent to end-users.

The product scans files on a wide range of Linux-supported file systems.

- Scans files on any Linux-supported file system.
- Superior detection rate with multiple scanning engines.
- A heuristic scanning engine can detect suspicious, potentially harmful files.
- The product can detect and categorize potentially unwanted applications.
- The product can be configured so that the users cannot bypass the protection.
- Files are scanned for viruses when they are opened or closed and before they are executed.
- You can specify what files to scan, how to scan them, what action to take when malicious content is found and how to alert about the infections.
- Recursive scanning of archive files.
- Virus definition database updates are signed for security.

The product works totally transparently to the end users.

- Virus definition databases are updated automatically without any need for end-user intervention.

Critical information of system files is stored and automatically checked before access is allowed.

- The administrator can protect files against changes so that it is not possible to install, for example, a trojan version of a software.
- An alert is sent to the administrator when a modified system file is found.

The default settings apply in most systems.

- Security policies are configured and distributed from one central location.

The product has extensive monitoring and alerting functions that can be used to notify any administrator in the company network about any infected content that has been found.

1.4 What is harmful content

Harmful applications and files can try to damage your data or gain unauthorized access to your computer system to steal your private information.

1.4.1 Viruses

A virus is usually a program that can attach itself to files and replicate itself repeatedly; it can alter and replace the contents of other files in a way that may damage the computer.

A **virus** is a program that is normally installed on the computer without the user's knowledge. Once there, the virus tries to replicate itself. The virus:

- uses some of the system resources
- may alter or damage files on the computer
- tries to use the computer to infect other computers
- may allow the computer to be used for illegal purposes.

1.4.2 Potentially unwanted applications (PUA) and unwanted applications (UA)

'Potentially unwanted applications' have features that you may like or want. 'Unwanted applications' have features that can harm your device or data more seriously.

An application may be called an 'potentially unwanted' (PUA) if it:

- **Hurts your privacy or productivity** - for example, exposes personal information or does things without your permission
- **Uses too many of your device's resources** - for example, takes up a lot of storage space or memory
- **Makes your device or data less secure** - for example, exposes you to unexpected content or other applications

The harm that these features can cause to your device or data can be from mild to serious. However, they are not dangerous enough to be called malware.

If an application has features that cause serious harm, it is labeled as 'unwanted application' (UA). The product will treat such applications with more caution.

The software will treat an application differently depending on whether it is a 'potentially unwanted' (PUA) or an 'unwanted application' (UA):

- **A potentially unwanted application** - The product stops the application from running. If you are sure that you trust the application, you can exclude it from scanning. To do this, you need to have administrative rights on the file to be excluded.
- **An unwanted application** - The product stops the application from running.

1.4.3 Worms

Worms are programs that send copies of themselves from one device to another over a network. Some worms also perform harmful actions on an affected device.

Many worms are designed to appear attractive to a user. They may look like images, videos, applications or any other kind of useful program or file. The aim of the deception is to lure the user into installing the worm. Other worms are designed to be completely stealthy, as they exploit flaws in the device (or in programs installed on it) to install themselves without ever being noticed by the user.

Once installed, the worm uses the device's physical resources to create copies of itself, and then send those copies to any other devices it can reach over a network. If a large quantity of worm copies is being sent out, the device's performance may suffer. If many devices on a network are affected and sending out worm copies, the network itself may be disrupted. Some worms can also do more direct damage to an affected device, such as modifying files stored on it, installing other harmful applications or stealing data.

Most worms only spread over one particular type of network. Some worms can spread over two or more types, though they are relatively rare. Usually, worms will try and spread over one of the following networks (though there are those that target less popular channels):

- Local networks
- Email networks
- Social media sites
- Peer-to-peer (P2P) connections
- SMS or MMS messages

1.4.4 Trojans

A trojan is a program that offers, or appears to offer, an attractive function or feature, but then quietly performs harmful actions in the background.

Named after the Trojan Horse of Greek legend, trojans are designed to appear attractive to a user. They may look like games, screensavers, application updates or any other kind of useful program or file. Some trojans will mimic or even outrightly copy popular or well-known programs to appear more trustworthy. The aim of the deception is to lure the user into installing the trojan.

Once installed, trojans can also use 'decoys' to maintain the illusion that they are legitimate. For example, a trojan disguised as a screensaver application or a document file will display an image or a document. While the user is distracted by these decoys, the trojan can quietly perform other actions in the background.

Trojans will usually either make harmful changes to the device (such as deleting or encrypting files, or changing program settings) or steal confidential data stored on it. Trojans can be grouped by the actions they perform:

- **Trojan-downloader:** connects to a remote site to download and install other programs
- **Trojan-dropper:** contains one or more additional programs, which it installs
- **Trojan-pws:** Steals passwords stored on the device or entered into a web browser
 - **Banking-trojan:** A specialized trojan-pws that specifically looks for usernames and passwords for online banking portals
- **Trojan-spy:** Monitors activity on the device and forwards the details to a remote site

1.4.5 Backdoors

Backdoors are features or specially made software that let someone bypass the security of a specific program, device, website, or service. Usually, they are used by attackers to get into a system or do something harmful without permission.

A feature in a program, device, website, or service can be considered a backdoor if its design or implementation introduces a security risk. For example, a hidden way for an administrator to access a website that always works with the same password could be considered a backdoor.

Backdoor software takes advantage of weaknesses in the code of a targeted program, device, website, or service. These weaknesses could be mistakes in the code, vulnerabilities, or hidden features that are not meant to be used.

Backdoors are usually used by attackers to get into a system without permission or to do something harmful while getting around security features such as limits on who can access files or devices, and to bypass authentication and encryption.

1.4.6 Exploits

Exploits are objects or ways of taking advantage of a flaw in a software to make it behave in a way that it was not meant. This makes it possible for an attacker to cause harm to your computer or device.

An exploit can be either an object or a method. For example, a specially made software, a piece of code, or a string of characters are all objects whereas a specific order of running commands is a method.

An exploit can be used to take advantage of a flaw or weak spot (also known as a vulnerability) in a software. Because each software is different, each exploit has to be made specifically for that software.

There are several ways an attacker can use an exploit to harm your computer or device:

- **Embedding it inside a hacked or specially made software** - when you install and launch the software, the exploit is launched
- **Embedding it inside a document that is sent as an email attachment** - when you open the attached document, the exploit is launched
- **Hosting it on a hacked or harmful website** - when you visit the website, the exploit is launched

Launching the exploit makes the software behave in a strange way, such as making it crash, or changing the system's storage or memory. This makes it possible for someone to harm your device and data, such as stealing your information or getting access to restricted sections of the operating system.

1.4.7 Exploit kits

Exploit kits are toolkits used by attackers to manage exploits and deliver harmful programs to a vulnerable computer or device.

An exploit kit contains an inventory of exploits, each of which can take advantage of a flaw (vulnerability) in a program, computer or device. The kit itself is usually hosted on a harmful or a hacked site, so that any computer or device that visits the site is exposed to its effects.

When a new computer or device connects to the booby-trapped site, the exploit kit probes it for any flaws that can be affected by an exploit in the kit's inventory. If one is found, the kit launches the exploit to take advantage of that vulnerability.

After the computer or device is compromised, the exploit kit can deliver a payload to it. This is usually another harmful program that is installed and launched on the computer or device, which in turn performs other unauthorized actions.

Exploit kits are designed to be modular and easy to use, so that their controllers can simply add or remove exploits and payloads to the toolkit.

1.4.8 Rootkits

Rootkits are programs that make other **malware** difficult to find.

Rootkit programs subvert the control of the operating system from its legitimate functions. Usually, a rootkit tries to obscure its installation and prevent its removal by concealing running processes, files or system data from the operating system. In general, rootkits do this to hide malicious activity on the computer.

Protection against userspace rootkits

If an attacker has gained an access to the system and tries to install a userspace rootkit by replacing various system utilities, **HIPS** detects modified system files and alerts the administrator.

Chapter 2

Installation

Topics:

This section contains instructions for installing the product.

- [Installing the product for use with WithSecure Elements Endpoint Protection](#)
- [Migrating from Business Suite to WithSecure Elements](#)
- [Uninstalling the product](#)

2.1 Installing the product for use with WithSecure Elements Endpoint Protection

These deployment methods are suitable for companies that use Linux devices and want to deploy WithSecure Elements Endpoint Agent.

WithSecure Elements Endpoint Agent for Linux provides two alternative methods for installing the agent on managed endpoint devices:

- a DEB or RPM package
- a generic installer package

Choose the correct package format based on the Linux distribution:

- DEB packages are compatible with Debian and Ubuntu systems.
- RPM packages are compatible with AlmaLinux, Amazon Linux, Rocky Linux, Oracle Linux, Red Hat Enterprise Linux, and SUSE Linux Enterprise Server systems.
- You can use the generic installer package on all supported systems.

There are two versions available for download: AMD64 and ARM64. You need to select the version that matches your image architecture.

When using any of the installers, you download the installer and make sure that the dependencies are installed. You then run the commands to activate the product.

When you install the product in WithSecure Elements Endpoint Protection managed mode, you can use the WithSecure Elements Endpoint Protection portal to centrally manage product installations.

The installation process consists of two steps: installing the product and activating it. The product requires a connection to cloud services for database updates and for the subscription validation. If the subscription cannot be validated for two weeks, the product stops working.

2.1.1 Installing the product using a DEB or RPM package

Instructions for using a DEB or RPM package to install the product.

Note: There are platform architecture-specific installers. You need to download the matching one.

1. To install the product, do the following:

- Go to the [WithSecure download page](#) and download the DEB or RPM installer package that matches your operating system and architecture, either AMD64 or ARM64.
- Log in to the Linux host as root.
- For checking that the required dependencies are installed, see [System requirements](#) on page 5.
- Deploy the installer package on the endpoint device using the system package manager:

- Depending on whether you use an AMD64 or ARM64 version, on DEB-based distributions, run one of the following commands:

```
dpkg -i linuxsecurity-installer_amd64.deb
```

```
dpkg -i linuxsecurity-installer_arm64.deb
```

- Depending on whether you use an AMD64 or ARM64 version, on RPM-based distributions, run one the following commands:

```
rpm -Uvh linuxsecurity-installer.x86_64.rpm
```

```
rpm -Uvh linuxsecurity-installer.aarch64.rpm
```

2. To activate the product, run the following command:

```
/opt/f-secure/linuxsecurity/bin/activate --psb --subscription-key SUBSCRIPTION-KEY --profile-id PROFILE-ID
```

Note: Replace SUBSCRIPTION-KEY with the product subscription key. The --profile-id parameter is optional but by adding it, the agent installation can be associated with one of the profiles available in the profile editor view in the portal. Replace PROFILE-ID with the numeric identifier of the profile.

Note: You can use the --override-distro option to install the product on a distribution that is not officially supported. For example, --override-distro rhel:8.6. Note that WithSecure cannot offer support for any issues with unsupported distributions.

Note: To use an HTTP proxy during the activation process, add the --http-proxy command line option. You can use this option in the following formats:

- --http-proxy=host:port: This configures the product to use the specified host and port as the network proxy without requiring authentication. If no port number is given, the default port number 3128 is used. Example: --http-proxy=proxy.example.com:8080.
- --http-proxy=username:password@host:port: This configures the product to use the specified host and port as the network proxy, with the given username and password serving as authentication credentials. Use URL encoding for any special characters in the username or password; for instance, if the password includes an '@' character, enter it as %40. Example:
--http-proxy=abc:x%40y%40z@proxy.example.com:8080.

2.1.2 Installing the product using a generic installer package

Instructions for using a generic package to install the product.

Note: There are platform architecture-specific installers. You need to download the matching one.

1. To install the product, do the following:

- a) Log in to WithSecure Elements Security Center.
- b) Select **Endpoint Protection > Downloads**.
- c) Under **WithSecure Elements Agent for Servers**, download the installer that matches your architecture, either AMD64 or ARM64.
- d) Log in to the Linux host as root.
- e) For checking that the required dependencies are installed, see [System requirements](#) on page 5.
- f) Depending on whether you use an AMD64 or ARM64 version, run one of the following commands to set the executable bit on the installer file that you downloaded:

```
chmod +x ./linuxsecurity-installer.amd64.bin
```

```
chmod +x ./linuxsecurity-installer.arm64.bin
```

2. Depending on whether you use an AMD64 or ARM64 version, run one of the following commands to install and activate the product:

```
./linuxsecurity-installer.amd64.bin --subscription-key SUBSCRIPTION-KEY --profile-id PROFILE-ID
```

```
./linuxsecurity-installer.arm64.bin --subscription-key SUBSCRIPTION-KEY --profile-id PROFILE-ID
```

Note: Replace SUBSCRIPTION-KEY with the product subscription key. The PROFILE-ID parameter is optional but by adding it, the agent installation can be associated with one of the profiles available in the profile editor view in the portal. Replace PROFILE-ID with the numeric identifier of the profile.

Note: You can also provide the subscription key and the profile ID for the installation by renaming the "linuxsecurity-installer" program to match one of the following patterns before running the program:
linuxsecurity-installer-[SUBSCRIPTION-KEY] or
linuxsecurity-installer-[SUBSCRIPTION-KEY]-[profile=PROFILE-ID]. It is important

that you use `[]` brackets around the subscription key and `profile=PROFILE-ID`. The renamed installer program can be run without any command line arguments.

Note: You can use the `--override-distro` option to install the product on a distribution that is not officially supported. For example, `--override-distro rhel:8.6`. Note that WithSecure cannot offer support for any issues with unsupported distributions.

Note: To use an HTTP proxy during the activation process, add the `--http-proxy` command line option. You can use this option in the following formats:

- `--http-proxy=host:port`: This configures the product to use the specified host and port as the network proxy without requiring authentication. If no port number is given, the default port number 3128 is used. Example: `--http-proxy=proxy.example.com:8080`.
- `--http-proxy=username:password@host:port`: This configures the product to use the specified host and port as the network proxy, with the given username and password serving as authentication credentials. Use URL encoding for any special characters in the username or password; for instance, if the password includes an '@' character, enter it as `%40`. Example:
`--http-proxy=abc:x%40y%40z@proxy.example.com:8080`.

2.1.3 Delaying product activation

Instead of activating the product immediately during installation or deployment, you can set the product activation to take place the next time the computer is turned on.

For example, you may want to delay activation if you are installing the product within a virtual machine template that is used for multiple virtual machine instances. In such a setup, you could install the product in the template and schedule activation for the next time the virtual machine is started. This approach allows individual virtual machines to be activated separately as they are taken into use.

1. To set activation to take place the next time the computer starts up:

- If you are using an `.rpm` or `.deb` installation package, add the `--next-boot` argument to the `activate` command, for example:

```
/opt/f-secure/linuxsecurity/bin/activate --next-boot --psb --subscription-key
SUBSCRIPTION-KEY
```

- If you are using the WithSecure installer, add the `--next-boot` argument to the installer command, for example:

```
./linuxsecurity-installer --next-boot --subscription-key SUBSCRIPTION-KEY
```

This sets Linux Protection to be activated when the computer is next started.

2. The product activates automatically when the system is rebooted. No user interaction is necessary during this activation process.

2.2 Migrating from Business Suite to WithSecure Elements

You can migrate Policy Manager-managed Linux Security 64 systems to WithSecure Elements using the `elements-migrator` command.

Note: A WithSecure Elements subscription key is required for this migration.

2.2.1 Inspecting the migration status

Instructions for inspecting the migration status.

To check the migration status of WithSecure Elements:

Run the following bash command:

```
/opt/f-secure/linuxsecurity/bin/elements-migrator status
```

2.2.2 Migrating Linux Security 64 to WithSecure Elements

Instructions for migrating Policy Manager-managed Linux Security 64 to WithSecure Elements.

To migrate Linux Security 64:

1. Run the following bash command:

```
/opt/f-secure/linuxsecurity/bin/elements-migrator migrate SUBSCRIPTION-KEY
```

2. Replace SUBSCRIPTION-KEY with your actual WithSecure Elements subscription key.

2.2.3 Assigning profile to migrated devices

You can automatically assign a specific profile to migrated devices.

To do that, you need to specify the profile ID in the migration command using the `--profile-id` option.

1. To assign a profile, run the following bash command:

```
/opt/f-secure/linuxsecurity/bin/elements-migrator migrate --profile-id PROFILE-ID  
SUBSCRIPTION-KEY
```

2. Replace PROFILE-ID with the numeric ID of the target Elements profile.
3. Replace SUBSCRIPTION-KEY with your valid subscription key.

2.3 Uninstalling the product

You can uninstall the product from the command line.

1. Log in to the Linux host as root.
2. Run the uninstallation command:

Note: The command that is given here is for the current version of the product. If you are uninstalling an older version, use `f-secure-linuxsecurity` as the package name instead.

- RHEL-based distributions: `rpm -e linuxsecurity-installer`
- Debian-based distributions: `dpkg -r linuxsecurity-installer`

If the above uninstallation command was not successful and the uninstallation fails, run the following command: `rm -rf /opt/f-secure /etc/opt/f-secure /var/opt/f-secure`. This command removes the remaining files or systemd services if any. Finally, restart the system to make sure that all hanging services and processes are run down.

The uninstall command does not remove configuration files or log files that the product has generated. Check the product directories for remaining files and remove them if you are sure that you do not need them anymore.

Chapter 3

Using the product

Topics:

- [Basics of using the WithSecure Elements Endpoint Protection portal](#)
- [Configuring settings with the lsctl utility](#)
- [Malware scanning](#)
- [Integrity checking](#)
- [Checking client status](#)
- [Automatic updates](#)
- [Using HTTP proxies](#)
- [Example: managing integrity checker profiles](#)
- [Command line usage](#)

You can configure the product settings with the `lsctl` command-line tool or (for centrally managed computers) through the WithSecure Elements Endpoint Protection (EPP) portal.

3.1 Basics of using the WithSecure Elements Endpoint Protection portal

The WithSecure Elements Endpoint Protection portal is used to change the settings and to view statistics of the WithSecure products.

Use the settings on the [Profiles > Linux Protection](#) page to configure the product.

For more information about the WithSecure Elements Endpoint Protection portal, see the [WithSecure Elements Endpoint Protection administrator's guide](#).

3.2 Configuring settings with the lsctl utility

You can check and edit the product settings using the `lsctl` command-line utility.

Note: If a setting for the distributed profile is locked in the Elements Endpoint Protection portal, that setting is enforced and you cannot edit it with the `lsctl` command-line utility.

To use the `lsctl` utility:

1. Log in to the Linux host as `root`.

2. Use the following commands:

- To see the current value of a setting, run `/opt/f-secure/linuxsecurity/bin/lsctl get [OPTIONS] [SETTING]`
- To assign a new value to a setting, run `/opt/f-secure/linuxsecurity/bin/lsctl set [OPTIONS] [SETTING] [VALUE]`
- To add an entry to a setting that holds several values, run `/opt/f-secure/linuxsecurity/bin/lsctl add [OPTIONS] [SETTING] [VALUE]`
- To remove an entry from a setting that holds several values, run `/opt/f-secure/linuxsecurity/bin/lsctl del [OPTIONS] [SETTING] [KEY]`
- To clear the value of a setting, run `/opt/f-secure/linuxsecurity/bin/lsctl reset [OPTION] [SETTING]`
- To see a list of the operations available for a setting, run `/opt/f-secure/linuxsecurity/bin/lsctl help [OPTIONS] [SETTING]`
- To load product settings from a backup, run `/opt/f-secure/linuxsecurity/bin/lsctl load [OPTIONS] [SETTING] [VALUE]`
- To get a list of all available settings, run `/opt/f-secure/linuxsecurity/bin/lsctl -h`

The `get`, `set`, `add`, `del`, and `load` commands support the following options:

- | | |
|-------------------|--|
| --json, -j | Treat input as JSON and produce output in JSON format. |
| --raw, -r | This is currently the default. Raw is an alternative input/output format that is similar to JSON with the following changes: <ul style="list-style-type: none"> • String values are accepted and returned without surrounding quotes. • Boolean values accept additional expressions such as <code>yes</code> and <code>no</code> in addition to <code>true</code> and <code>false</code>. |

Note: The default input/output format for `lsctl` is subject to change. If you rely on the format of `lsctl`, specify either `--json` or `--raw` explicitly when `lsctl` is run. In addition to setting values, the keys for array-type settings are considered as input and need to be entered in the correct format.

The `set`, `add`, and `load` commands accept the following options:

- | | |
|--------------------------|---|
| --prompt, -p | Prompt for input using a text editor. |
| --file, -f [PATH] | Read the input value from a file. If the path is set to <code>-</code> , the input value is read from the standard input. |

If you use the `--prompt` or `--file` options, do not enter a [VALUE] for the command.

Note: The `set` and `load` commands differ in how they handle settings that are not included in the input. The `set` command does not change the current values for any settings that are missing from the input, whereas the `load` command resets those settings to their default values.

3.2.1 Setting types and the settings tree

Each product setting has a type and a value, and the settings are organized in a tree-like structure.

There are five basic types of settings:

- Boolean (`true` or `false`)
- Text
- Numbers
- Objects, which combine a specific group of settings in one entity by assigning a name to each member. You can inspect and assign values to objects.
- Arrays, which are collections that contain a varying number of members. Each array member is identified by a unique key. Array keys are composed of the same basic value types as settings, for example a key might be a string, number, or object. You can add and delete the settings in an array. Each array can have additional restrictions on the type of settings that it accepts, for example it may only accept text-type settings as members.

Product settings are organized in a tree structure, where the root is an object-type setting that contains all the other settings. To inspect this root setting, use the `get` sub-command, for example:

```
# /opt/f-secure/linuxsecurity/bin/lsctl get
```

```
{
  "http_proxy": {
    "enabled": false,
    "host": "localhost",
    "port": 3128
  },
  "ic": {
    "enabled": true,
    "profiles": []
  },
  "oas": {
    "actions": {
      "malware": "rename",
      "pua": "none",
      "suspected": "none"
    },
    "archive_max_nested": 5,
    "block_archive_max_nested": false,
    "block_encrypted_archives": false,
    "detect_pua": true,
    "enabled": true,
    "exclude_paths": [],
    "include_paths": [],
    "scan_archives": false,
    "scan_only_executables": false
  },
  ...
}
```

Note that this example does not include the full output.

The output shows the setting hierarchy, starting from the root object and continuing through all its child settings (`http_proxy`, `ic`, `oas`, etc.) and their sub-settings recursively.

In this example, the root object contains the `http_proxy` setting, which is an object-type setting that contains three sub-settings: `enabled` (boolean value setting), `host` (text setting), and `port` (number setting).

You can also use the `get` sub-command to inspect only parts of the setting tree. It selects the settings to retrieve using the names that object-type settings assign and the keys of array-type settings.

For example, you can retrieve the HTTP proxy settings alone as follows:

```
# /opt/f-secure/linuxsecurity/bin/lscctl get http_proxy
```

```
{
  "enabled": false,
  "host": "localhost",
  "port": 3128
}
```

In the same way, you can specify the command further to check if the use of an HTTP proxy is enabled by running:

```
# /opt/f-secure/linuxsecurity/bin/lscctl get http_proxy enabled
```

```
false
```

In addition to their type, settings can have additional constraints on the allowed values. For example, the `http_proxy port` setting does not allow negative numbers.

3.2.2 Operating with settings

There are six basic sub-commands for operating with settings: `get`, `set`, `add`, `delete`, `reset`, and `load`.

- | | |
|---------------|--|
| get | <pre>lscctl get [--json --raw] SETTING...</pre> <p>Use the <code>get</code> sub-command to inspect the current value of a setting.</p> |
| set | <pre>lscctl set [--json --raw] [--prompt --file PATH] SETTING... VALUE</pre> <p>Use the <code>set</code> sub-command to enter the value for a setting.</p> |
| add | <pre>lscctl add [--json --raw] [--prompt --file PATH] SETTING... VALUE</pre> <p>Use the <code>add</code> sub-command to add new members to an array-type setting. The key that is used for the new array member is automatically derived from the value. How the key is derived is specific to each particular setting.</p> |
| delete | <pre>lscctl delete [--json --raw] SETTING... KEY</pre> <p>Use the <code>delete</code> sub-command to remove members from array-type settings. This means that you can only use it with settings that are immediate descendants of an array-type setting. You can use <code>del</code> as an alias for the <code>delete</code> sub-command.</p> |
| reset | <pre>lscctl reset [--json --raw] [--all] SETTING...</pre> <p>Use the <code>reset</code> sub-command to clear the local value of a setting. This means that the setting's value is set back to its default value. This also enables the setting to be changed using Policy Manager or Protection Service for Business.</p> |
| load | <pre>lscctl load [--json --raw] [--prompt --file PATH] SETTING... VALUE</pre> <p>Use the <code>load</code> sub-command to restore the values for settings.</p> <p>Note: The <code>set</code> and <code>load</code> commands differ in how they handle settings that are not included in the input. The <code>set</code> command does not change the current values for any settings that are missing from the input, whereas the <code>load</code> command resets those settings to their default values.</p> |

Providing values for settings

The `set` and `add` sub-commands are used to alter setting values. Use the final command-line argument to enter the value to assign to a setting (for `set` sub-commands) or the member to add to an array-type setting (for `add` sub-commands).

For example, to set a new value for the `http_proxy` host setting, you could use the following command:

```
lsctl set http_proxy host example.com
```

This assigns `example.com` as the new value for the `host` text-type setting.

When you use the `--prompt (-p)` flag, `lsctl` opens a text editor for you to specify the value for a setting. For example:

```
lsctl set --prompt http_proxy host
```

This opens a text editor that you can use to enter a new value for the `host` setting.

By default, `lsctl` opens the editor specified by the `EDITOR` environment variable. If `EDITOR` is not set or the specified editor is not found in `PATH`, `lsctl` tries to find the following editors in order until it finds one that is available: `nano`, `vim`, `vi`.

When you use the `--file (-f)` switch, `lsctl` reads the value for a setting from a file. For example:

```
lsctl set --file value.txt http_proxy host
```

This reads the value for the `host` setting from the `value.txt` file. You can also use the special `-` value to read the value from the standard input. For example:

```
echo -n 'example.com' | lsctl set --file - value.txt http_proxy host
```

This uses the output of the `echo` command as the value for the `host` setting.

Related tasks

[Loading product settings from a file](#) on page 21

You can use the `lsctl load` command to restore settings from a previously saved JSON file.

3.2.3 Input and output formats

The format defines how the setting values that you provide are interpreted and how `lsctl` prints the setting values in its output.

An input format defines how the entered setting values should be formatted. The input format affects all sources of inputs. This means that it does not matter if the setting value is entered using command line arguments or if it is read from a file (by using the `--file` switch); all input must follow the same formatting rules.

An output format defines how setting values are formatted when `lsctl` prints them.

`lsctl` currently supports two formats: `json` and `raw`. You can select these using the `--json (-j)` and `--raw (-r)` flags respectively. By default, `lsctl` currently uses the `raw` format, but this is subject to change. If you depend on the exact input or output format of `lsctl`, always specify the desired format explicitly by using these flags.

JSON

When you specify the `--json` flag, `lsctl` expects the input to be valid JSON and outputs valid JSON. For example:

```
lsctl set --json http_proxy host '"example.com"'
```

Note the need for double quotes. JSON strings are always enclosed in double quotes (`"`). However, because of the way `shell` handles word splitting, you need to enclose the JSON string within an additional pair of single quotes to prevent `shell` from removing the quotes from the value. As this is something specific to the way `shell` handles command line arguments, you do not need this when the value is read from a file (by using the `--file` switch) or received from an interactive editor (the `--prompt` flag).

Format flags also affect the output format. For example:

```
lsctl get --json http_proxy host
```

```
"example.com"
```

Note that the output is a valid JSON string literal.

Raw

When you specify the `--raw` flag, `lsctl` uses the raw format. The raw format modifies the JSON format slightly to make certain operations more ergonomic for command line users. The differences between `json` and `raw` formats are:

- Text values for settings do not need to be enclosed in double quotes. This means that the following command is valid:

```
lsctl set http_proxy host example.com
```

- Boolean values for settings support additional literals (`yes`, `y`, `no` and `n`) in addition to `true` and `false`. This makes the following command valid:

```
lsctl set http_proxy enabled yes
```

Note that these additions only apply when you are operating directly with text or boolean value settings. They are not applied when these settings appear as a part of other settings. Therefore, the following command **is not valid**:

```
lsctl set http proxy '{"enabled": yes, "host": example.com, "port": 9090}'
```

3.2.4 Working with arrays

Arrays differ from other settings in that they can contain a variable number of member settings.

You can add members to and remove them from arrays using the `add` and `delete` sub-commands.

For example, you can add a new directory to include in on-access scanning by adding a new member to the `oas_include_paths` setting, which is an array setting that contains string members:

```
lsctl add oas include_paths /home
```

The above command adds the `/home` directory to the array.

To remove the newly added directory from `include_paths`, use the following command:

```
lsctl del oas include_paths /home
```

Arrays can also include more complex settings are also possible. For example, integrity checker profiles are stored in the `ic_profiles` array. Each integrity checker profile is an object that defines the profile properties. To add a new integrity checker profile, you could use the following command:

```
lsctl add --file - ic profiles <<EOF
{
  "path": "/home",
  "verify_attributes": {
    "mode": false,
    "user": false,
    "group": false,
    "size": false,
    "mtime": false
  },
  "read_action": "deny",
  "write_action": "allow"
}
EOF
```

Note: This example uses shell's `here` document syntax to make it easier to enter multi-line values.

Note the use of the `--file` switch with the `-` argument to indicate that the value should be read from the standard input. Alternatively, you could use the `--prompt` flag to make entering large input values easier.

To delete the newly added profile, use the following command:

```
lsctl del ic profiles /home
```

Here the path member is the key that is used to identify members within the `profiles` array. How an array's elements are identified is specific to each array.

You can also access settings within arrays using the usual sub-commands. For example, you could retrieve the value of the `write_action` setting inside the newly added profile:

```
lsctl get ic profiles /home write_action
```

```
allow
```

Or you could change the value of the mode setting within `verify_attributes`:

```
lsctl set ic profiles /home verify_attributes mode yes
```

In addition to affecting the way that setting values are handled, input and output formats also affect how you enter array keys. For example, if you want to retrieve the value of `write_action` settings when using the JSON format, you would have to enter the `lsctl` command like this:

```
lsctl get --json ic profiles '/home' write_action
```

```
"allow"
```

3.2.5 Loading product settings from a file

You can use the `lsctl load` command to restore settings from a previously saved JSON file.

This allows you to save your tested, functioning product configuration as a backup and then restore it if necessary. To save the current product configuration, you could use the following command, for example:

```
lsctl get --json > [PATH]
```

For this command, replace `[PATH]` with the full path and filename for the JSON backup file.

To restore the product settings, run the following command:

```
lsctl load --json --file [PATH]
```

This applies all the settings that are available in the backup file specified as `[PATH]` to your product configuration. Any settings that are missing are reset to their default value.

3.2.6 Inspecting the state of the product

You can use the `lsctl` command-line utility to inspect the state of the product and to view various scanning and performance statistics with the `status` sub-command.

Inspecting different status values works similarly to inspecting setting values:

- To get a status value, run `lsctl status get [--json | --raw] STATUS...`
- To see the help for a status, run `lsctl status help [--json | --raw] STATUS...`

scan_service	Type: JSON object All statistics related to the scanning service.
scan_service connection_count	Type: integer The number of connected hosts.
scan_service last_detection	Type: string

	The name of the last detection.
scan_service max_connection_count	Type: JSON object All statistics related to the maximum number of connected hosts.
scan_service max_connection_count daily	Type: integer The maximum number of connected hosts in the last 24 hours.
scan_service max_connection_count weekly	Type: integer The maximum number of connected hosts in the last week.
scan_service max_connection_count monthly	Type: integer The maximum number of connected hosts in the last month.
scan_service transaction_count	Type: JSON object All statistics related to the scanning transaction figures.
scan_service transaction_count total	Type: integer The total number of transactions.
scan_service transaction_count daily	Type: integer The number of transactions in the last 24 hours.
scan_service transaction_count weekly	Type: integer The number of transactions in the last week.
scan_service transaction_count monthly	Type: integer The number of transactions in the last month.
scan_service detection_count	Type: JSON object All statistics related to the detection figures.
scan_service detection_count total	Type: integer The number of detections.
scan_service detection_count daily	Type: integer The number of detections in the last 24 hours.
scan_service detection_count weekly	Type: integer The number of detections in the last week.
scan_service detection_count monthly	Type: integer The number of detections in the last month.
scan_service min_transaction_latency	Type: JSON object All statistics related to the minimum latency in scanning transactions.
scan_service min_transaction_latency daily	Type: integer The minimum transaction latency (milliseconds) in the last 24 hours.
scan_service min_transaction_latency weekly	Type: integer The minimum transaction latency (milliseconds) in the last week.
scan_service min_transaction_latency monthly	Type: integer The minimum transaction latency (milliseconds) in the last month.
scan_service max_transaction_latency	Type: JSON object All statistics related to the maximum latency in scanning transactions.

scan_service max_transaction_latency daily	Type: integer The maximum transaction latency (milliseconds) in the last 24 hours.
scan_service max_transaction_latency weekly	Type: integer The maximum transaction latency (milliseconds) in the last week.
scan_service max_transaction_latency monthly	Type: integer The maximum transaction latency (milliseconds) in the last month.
scan_service avg_transaction_latency	Type: JSON object All statistics related to the average latency in scanning transactions.
scan_service avg_transaction_latency daily	Type: integer The average transaction latency (milliseconds) in the last 24 hours.
scan_service avg_transaction_latency weekly	Type: integer The average transaction latency (milliseconds) in the last week.
scan_service avg_transaction_latency monthly	Type: integer The average transaction latency (milliseconds) in the last month.
product_version	Type: JSON object All information related to the product version.
product_version name	Type: string The name of the pinned product version. This can be an empty string if no pinned product version has been specified. Note: If you use a pinned version of the product, it works for one year from the date when the new pinnable product version is released. You should update the product to the new version during that time.
product_version status	Type: string A string representing the status of the pinned product version. The value can be unset, error, in-progress, or applied.

3.2.7 Command-line settings for real-time scanning

The settings related to real-time scanning that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

oas	Type: JSON object All settings related to real-time scanning.
oas enabled	Type: boolean Controls whether real-time scanning is on or off.
oas include_paths	Type: JSON array The paths included in real-time scanning.
oas include_paths [PATH]	Type: string A path included in real-time scanning. For use with <code>add</code> and <code>del</code> commands. <code>[PATH]</code> must be a full, absolute path name.

	You can use wildcards in paths, with ** matching any subdirectories.
oas exclude_paths	Type: JSON array The paths excluded from real-time scanning.
oas exclude_paths [PATH]	Type: string A path excluded from real-time scanning. For use with add and del commands. [PATH] must be a full, absolute path name. You can use wildcards in paths, with ** matching any subdirectories.
oas trusted_application_paths	Type: JSON array The paths to trusted applications whose file access events are excluded from real-time scanning.
oas trusted_application_paths [PATH]	Type: string A path to the executable file of a trusted application whose file access events are excluded from real-time scanning. For use with add and del commands. [PATH] must be a full, absolute path name. The path is a prefix match: any file path that starts with this path matches. For example, if you set /opt/app/ , then /opt/app/bin/myapp matches. You can use wildcards in paths, with ** matching any subdirectories. Note: Exclusions reduce scanning but do not guarantee complete bypass, especially for short-lived processes.
oas actions	Type: JSON object The actions configuration for real-time scanning.
oas actions malware	Type: string The action on detecting files that are identified as malware. Allowed values are: • remove - delete the file • rename - add a .malware extension to the file name • none - do not do anything to the file
oas actions suspected	Type: string The action on detecting files that are suspected to be unsafe. Allowed values are: • remove - delete the file • rename - add a .suspected extension to the file name • none - do not do anything to the file
oas detect_pua	Type: boolean Controls whether scanning for potentially unwanted applications (PUA) is on or off.
oas actions pua	Type: string The action on detecting files that are identified as potentially unwanted applications. Allowed values are: • remove - delete the file • rename - add a .pua extension to the file name • none - do not do anything to the file
oas scan_archives	Type: boolean Controls whether scanning inside archive files is on or off.
oas block_archive_max_nested	Type: boolean

	Controls whether or not archive files that are too deeply nested are considered unsafe.
oas archive_max_nested	Type: integer The maximum allowed level of nesting for archive files. If <code>oas block_archive_max_nested</code> is set to <code>true</code> , archive files that exceed this level of nesting are considered unsafe.
oas block_encrypted_archives	Type: boolean Controls whether or not encrypted archive files are considered unsafe.
oas scan_only_executables	Type: boolean Controls whether or not scanning is restricted to files that have execute permissions.

Settings for manual and scheduled scanning

The settings related to manual and scheduled scanning that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

ods	Type: JSON object All settings related to manual and scheduled scanning.
ods exclude_paths	Type: JSON array The paths excluded from scheduled scanning.
ods exclude_paths [PATH]	Type: string A path excluded from scheduled scanning. For use with <code>add</code> and <code>del</code> commands. <code>[PATH]</code> must be a full, absolute path name. You can use wildcards in paths, with <code>**</code> matching any subdirectories.
ods actions	Type: JSON object The actions configuration for manual and scheduled scanning.
ods actions malware	Type: string The action on detecting files that are identified as malware. Allowed values are: <code>remove</code> - delete the file <code>rename</code> - add a <code>.malware</code> extension to the file name <code>none</code> - do not do anything to the file
ods actions suspected	Type: string The action on detecting files that are suspected to be unsafe. Allowed values are: <code>remove</code> - delete the file <code>rename</code> - add a <code>.suspected</code> extension to the file name <code>none</code> - do not do anything to the file
ods detect_pua	Type: boolean Controls whether scanning for potentially unwanted applications (PUA) is on or off.
ods actions pua	Type: string The action on detecting files that are identified as potentially unwanted applications. Allowed values are: <code>remove</code> - delete the file <code>rename</code> - add a <code>.pua</code> extension to the file name

- none - do not do anything to the file

ods scan_archives	Type: boolean Controls whether scanning inside archive files is on or off.
ods block_archive_max_nested	Type: boolean Controls whether or not archive files that are too deeply nested are considered unsafe.
ods archive_max_nested	Type: integer The maximum allowed level of nesting for archive files. If <code>ods block_archive_max_nested</code> is set to <code>true</code> , archive files that exceed this level of nesting are considered unsafe.
ods block_encrypted_archives	Type: boolean Controls whether or not encrypted archive files are considered unsafe.
ods schedule	Type: JSON object All settings related to scheduled scanning.
ods schedule schedule_type	Type: string Type of scheduled scanning in use. Currently the only allowed value is <code>weekly</code> .
ods schedule weekly_schedule	Type: JSON object Settings for scanning on a weekly schedule.
ods schedule weekly_schedule [monday tuesday wednesday thursday friday saturday sunday]	Type: boolean This group of settings controls if the scheduled scan is run on the given weekday.
ods schedule weekly_schedule time_of_day	Type: string The time to start a scheduled scan on the set days. The value is a 24-hour time value (HH:MM) in the endpoint's time zone.

ORSP settings

The settings related to the Object Reputation Services Platform (ORSP) that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

orsp	Type: JSON object Settings related to using the cloud-based Object Reputation Services Platform (ORSP) in scanning.
orsp enabled	Type: boolean Controls whether ORSP is switched on or off for file scanning.

3.2.8 Command-line settings for integrity checking

The settings related to integrity checking that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

ic	Type: JSON object All settings related to integrity checking.
ic enabled	Type: boolean Controls whether integrity checking is on or off.
ic profiles	Type: JSON array

	The baseline list of path names included in integrity checking.
ic profiles [PATH]	Type: JSON object All integrity checking settings for the given path.
ic profiles [PATH] path	Type: string A path name included in integrity checking.
ic profiles [PATH] verify_attributes	Type: JSON object All monitored attributes for the given path.
ic profiles [PATH] verify_attributes [mode user group size mtime]	Type: boolean This group of settings controls whether to monitor changes in the file mode, the owning user or group, size, or modification time of the given path.
ic profiles [PATH] read_action	Type: string The action on an attempt to read from the given path if its contents or monitored attributes have changed. The allowed values are a <code>l</code> low and <code>deny</code> , which respectively permit or prevent access to the file.
ic profiles [PATH] write_action	Type: string The action on an attempt to write to the given path. The allowed values are a <code>l</code> low and <code>deny</code> , which respectively permit or prevent access to the file.
ic profiles [PATH] exclude_patterns	Type: JSON array List of file name patterns to be excluded from integrity checking. For example, <code>"*.log"</code> pattern could be used to exclude all files with <code>.log</code> file extension from integrity checking. Note: You can use wildcards in file name patterns, but the path does not support them.
ic system_updates	Type: boolean This group of settings controls whether baseline is automatically updated whenever packages are installed or upgraded with package managers.
ic system_updates apt	Type: boolean This group of settings is for the APT package management system.
ic system_updates apt enabled	Type: boolean When the apt integration is enabled, integrity checking updates the baseline automatically whenever packages are installed or upgraded with apt and its related tools like apt - get.
ic system_updates dnf	Type: boolean This group of settings is for the DNF package manager.
ic system_updates dnf enabled	Type: boolean When the dnf integration is enabled, integrity checking updates the baseline automatically whenever packages are installed or upgraded with the dnf package manager.

3.2.9 Command-line settings for offload scanning

The settings related to offload scanning that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

offload_scanning	Type: boolean
-------------------------	---------------

	All settings related to offload scanning.
offload_scanning enabled	Type: boolean Controls whether offload scanning is on or off.
offload_scanning icap_servers	Type: JSON array List of ICAP servers that can be used for offloading scans.
offload_scanning icap_servers [SERVER]	Type: string The ICAP server settings.
offload_scanning icap_servers [SERVER] address	Type: string The address of the ICAP server that performs the offload scanning.
offload_scanning icap_servers [SERVER] port	Type: integer The port number for offload scanning on the ICAP server.
offload_scanning timeout	Type: integer This setting controls the time, in seconds, to wait for a response from the offloading server.

3.2.10 Command-line settings for automatic updates

The settings related to automatic updates that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

updates	Type: JSON object All settings related to automatic updates.
updates enabled	Type: boolean Controls whether automatic updates are switched on or off.
updates product_version	Type: string Note: Do not use this setting. This setting is reserved for future feature releases.
updates schedule	Type: JSON object The automatic update schedule.
updates schedule regime	Type: string The type of scheduling for automatic updates. The allowed values are: • <code>on_arrival</code> - updates are applied as soon as they become available • <code>at_date</code> - updates are postponed until the time specified in the <code>at_date_schedule</code> setting • <code>with_repetition</code> - updates are applied regularly according to the <code>repetition_schedule</code> settings
updates schedule at_date_schedule	Type: integer The time to apply updates when <code>regime</code> is set to <code>at_date</code> . The value must be a timestamp in epoch time format.
updates schedule repetition_schedule	Type: JSON object Settings related to automatic updates with a fixed repetition schedule.
updates schedule repetition_schedule day	Type: string The scheduled day for repeated updates. Allowed values are <code>daily</code> , to check for updates every day at the specific time, or <code>monday</code> ,

	tuesday, wednesday, thursday, friday, saturday, or sunday to check for updates on the given day each week.
updates schedule repetition_schedule time_of_day	Type: string The scheduled time of day for applying updates. The value is a 24-hour UTC time value (HH : MM).
updates use_tls	Type: boolean Controls whether Transport Layer Security (TLS) is used when downloading updates.

Related concepts

[Services installed with the product](#) on page 41

The product installs several services that are used to handle various product features.

Related tasks

[Updating the product manually using an update archive](#)

HTTP proxy settings

The settings related to HTTP proxy usage that are available for `lsctl`.

Tip: To see an example of the structure for any of the settings, you can check the output for the following command: `/opt/f-secure/linuxsecurity/bin/lsctl get [--json|--raw] [SETTING]`

http_proxy	Type: JSON object All HTTP proxy settings.
http_proxy enabled	Type: boolean Controls whether or not to use an HTTP proxy for features that require network access.
http_proxy host	Type: string The host name of the HTTP proxy.
http_proxy port	Type: integer The HTTP proxy port. The value must be a number between 1 and 65535.

3.3 Malware scanning

The product protects the computer from programs that may damage files, steal personal information, or use it for illegal purposes.

The product also does the following:

- It can protect the computer against **malware** in both real time in the background and on demand by using a manual scan.
- It can scan specified files and directories, any removable media (such as portable drives) and downloaded content automatically. The product monitors the computer for any changes that may indicate **malware**.

Real-time scanning is enabled by default, but the list of files and directories to be included in scanning needs to be added in the Elements Endpoint Protection (EPP) portal before scanning takes place.

Related concepts

[Real-time scanning](#) on page 30

Real-time scanning protects the computer by scanning files when they are accessed and blocking access to files that contain **malware**.

[Manual scanning](#) on page 30

The product can scan entire computers to verify that there are no harmful files or unwanted applications.

[Scheduled scanning](#) on page 31

Scheduled scanning can help to scan computers when they are not in use and to periodically make sure that the computer is clean.

3.3.1 Real-time scanning

Real-time scanning protects the computer by scanning files when they are accessed and blocking access to files that contain **malware**.

Note: Individual files and directories that require scanning in real time need to be specified in the Elements Endpoint Protection (EPP) portal. With the default settings, real-time scanning does not scan any files.

Real-time scanning works as follows:

1. The computer tries to access a file.
2. The file is immediately scanned for **malware** before the computer is allowed access to the file.
3. If **malware** is found in the file, real-time scanning blocks access to the file so the **malware** cannot harm the computer.
4. Based on the settings, real-time scanning may either rename or delete the infected file.

Scanned files are classified as clean, malware, potentially unwanted applications, or suspicious. Clean files are not affected by real-time scanning. Infected files are prevented from being opened.

The amount of time and system resources that real-time scanning takes depends on the contents, location, and type of the file.

Files that take a longer time to scan:

- Compressed files, such as .zip archives. Note that these files are not scanned by default.
- Files on network file systems.
- Large files may be affected.

Real-time scanning may slow down your computer when a lot of files are accessed at the same time.

Note: You can find all scan events in the access . Log file.

Related concepts

[Command-line settings for real-time scanning](#) on page 23

The settings related to real-time scanning that are available for `lsctl`.

Related tasks

[Using real-time scanning with Policy Manager](#)

[Scanning the computer manually from the command line](#) on page 35

You can scan the computer for malware manually from the command line of a Linux host that has the product installed.

3.3.2 Manual scanning

The product can scan entire computers to verify that there are no harmful files or unwanted applications.

A full computer scan can take a long time to complete, so by limiting the set of files and directories, this takes less time. All other files locations and directories can be excluded from the scan through the Elements Endpoint Protection (EPP) portal.

For finer-grained control over manual scanning, the command line functionality using the `fsanalyze` command can be used.

Related tasks

[Scanning the computer manually from the command line](#) on page 35

You can scan the computer for malware manually from the command line of a Linux host that has the product installed.

3.3.3 Scheduled scanning

Scheduled scanning can help to scan computers when they are not in use and to periodically make sure that the computer is clean.

Scanning of specific files and directories or any removable media can be scheduled to take place at certain times of the day and on specific days of the week.

3.4 Integrity checking

Integrity checking protects important system files against unauthorized modifications.

You can use integrity checking to detect any modifications to protected files and prevent their use, regardless of file system permissions.

To use integrity checking, you need to add the files that you want to protect to a baseline list. The files on this list are protected against unauthorized changes.

Integrity checking works by comparing files on the disk to the baseline attributes, which form a cryptographically signed list of file properties. Integrity checking sends alerts to the administrator of attempts to modify the monitored files.

Note: Some aspects of integrity checking depend on the type of file system in use. Fully supported file systems include Ext4, ZFS, BTRFS, NFS, CIFS, and others. In essence, full integrity checking takes advantage of file attributes, which are available on most modern file systems, but not all.

The following table gives more details on how integrity checking handles various events for files that are included in the baseline.

Event	Integrity checking options
An untrusted process modifies a file	Configured with the read and write action settings: - If the write action is set to deny, the modification attempt is blocked and no alert is sent. - If the write action is set to allow, integrity checking sends an alert of the modification to the administrator. - If the read action is set to deny, integrity checking blocks any subsequent attempt to open the file after it has been modified.
A trusted process modifies a file	If the program file for a process is included in the baseline and has not been tampered with, it is considered a trusted process. In such cases, integrity checking does not take any action.
Deleting a file	Integrity checking does not prevent or send an alert for the event.
Renaming a file	Integrity checking does not prevent or send an alert for the event, but continues to monitor the renamed file.
Creating a hard or soft link to a file	Integrity checking follows the link.

Event	Integrity checking options
Replacing a file	Integrity checking does not prevent or send an alert for the event.
Changing file permissions	The change is not blocked, but the read and write action settings determine if subsequent attempts to open the file are blocked or prompt an alert.
Changing file ownership	The change is not blocked, but the read and write action settings determine if subsequent attempts to open the file are blocked or prompt an alert.
Changing file attributes	This includes SELinux labels. Integrity checking does not prevent or send an alert for the event.

Note: Regarding the alerts that integrity checking sends:

- tampering alerts mean that a process is accessing a file that has already been tampered with.
- tampering-action alerts mean that a process is tampering with a file. These alerts are only sent if the write action is set to allow.
- If the write action is set to allow, all alerts are sent, regardless of the read action setting.

Related concepts

[Command-line settings for integrity checking](#) on page 26

The settings related to integrity checking that are available for `lsctl`.

Related tasks

[Using integrity checking with Policy Manager](#)

[Running the integrity checker from the command line](#) on page 38

Integrity checking functionality is partially available on the command line.

3.5 Checking client status

The **Send full status update** request is an option through the WithSecure Elements Endpoint Protection portal (EPP) for checking the status of client installations.

By selecting the **Send full status update** option on the **Device** page, the clients sync with our backend and update the portal with any available information about their current status.

The request also checks for any available updates at the same time. If the client is configured to install updates immediately, the client also installs the updates available.

In short, the **Send full status update** request enables the following:

- It checks the clients status information,
- It checks for any available updates (product and engine), and
- It installs any available updates, if configured on the client side.

In effect, this is a way to check for updates on a regular basis.

3.6 Automatic updates

Automatic updates make sure that the Linux hosts in your managed network stay protected.

With automatic updates, the managed hosts retrieve the latest updates based on the policy settings. By default, automatic updates are turned on. We strongly recommend that you keep this feature turned on, as new virus definitions are made available several times a day.

WithSecure uses dedicated service connections between the backend systems and installed products, referred to as channels, to deliver updates to the malware definition databases, scanning engines, and the actual products. Each channel provides updates for a specific product component or function.

The settings for automatic updates apply to the three WithSecure product channels (`linuxsecurity`, `fsbg`, and `baseguard`) that the product uses. The automatic update settings do not affect updates for other channels, such as the scanning engines, as they are applied immediately when they are available. The exception to this is the main feature switch for automatic updates: turning automatic updates off disables the updates for all channels.

In addition, product channel updates are applied one week after the update has been published at the latest.

Checking the latest virus definition database versions

To ensure your systems are using the most up-to-date virus definition databases, you can query the Elements API using the following endpoint:

Endpoint:	GET /databases/v1/latest-versions
Purpose:	Retrieve the latest version numbers of all available virus definition databases.
Response:	A list of databases, each with its corresponding latest version number.
Usage:	Use this endpoint to verify that your systems are aligned with the most current database versions. This helps maintain optimal security and performance across your environment.

Note: Even if a newer database version is available, your endpoints might not have received it yet, as this depends on how frequently the endpoint client applications check for updates.

For information on how to use this endpoint, see

<https://connect.withsecure.com/api-reference/elements#get-/databases/v1/latest-versions>

To get started with the Elements API, see <https://connect.withsecure.com/getting-started/elements>

Related concepts

[Command-line settings for automatic updates](#) on page 28

The settings related to automatic updates that are available for `lsctl`.

Related tasks

[Configuring automatic update options with Policy Manager](#)

[Updating the product manually using an update archive](#)

3.7 Using HTTP proxies

The product has a single, global HTTP proxy setting, which is always used when the product makes HTTP requests to external services.

You can configure the global HTTP proxy setting using a command line option either when activating the product or with the `lsctl` utility after you have activated the product.

If you use the command line option to define the product settings, the product saves the configuration as a local setting override. This means that once you have finished activating the product, you can view the settings with the following command: `lsctl get http_proxy`.

Local overrides take precedence over the configuration that is retrieved from remote management services (WithSecure Elements Endpoint Protection or Policy Manager). To configure the proxy settings remotely, you have to mark the setting as locked in the management service. When the setting is locked, the value set in the profile or policy that is distributed from the management service is always used, even if a local configuration exists.

If you specify the global HTTP proxy, the product uses the proxy to communicate with Elements Endpoint Protection. You can also change the proxy settings in the Elements Endpoint Protection portal.

1. Set the HTTP proxy:

- To set the proxy when you run the activation or installer command, include the `--http-proxy` option in the command:

```
--http-proxy=<host address>:<host port number>
```

Note: The product supports basic authentication for HTTP proxies. If the specified proxy requires authentication, use the `--http-proxy=username:password@host:port` format to specify the credentials for the proxy.

- To set the proxy after activation, run the following `lsctl` commands:

```
/opt/f-secure/linuxsecurity/bin/lsctl set http_proxy host <host address>
/opt/f-secure/linuxsecurity/bin/lsctl set http_proxy port <host port number>
/opt/f-secure/linuxsecurity/bin/lsctl set http_proxy enabled true
```

2. Run the following command to check the proxy setting:

```
/opt/f-secure/linuxsecurity/bin/lsctl get http_proxy
```

This returns the current settings:

```
{
  "enabled": true,
  "host": <host address>,
  "port": <host port number>
}
```

Related tasks

[Deploying the installation package](#)

[Installing the product for use with WithSecure Elements Endpoint Protection](#)

3.8 Example: managing integrity checker profiles

This example of a common task where you can use `lsctl` shows how to modify integrity checker profiles through the `ic_profiles` array.

The following steps outline how to add an integrity checker profile to prevent writing to files inside the `/opt` directory.

1. Run the following command:

```
/opt/f-secure/linuxsecurity/bin/lsctl set ic enabled yes
/opt/f-secure/linuxsecurity/bin/lsctl add --file - ic_profiles <<EOF
{
  "path": "/opt",
  "verify_attributes": {
    "mode": false,
    "user": false,
    "group": false,
    "size": false,
    "mtime": false
  },
  "read_action": "allow",
  "write_action": "deny"
}
EOF
```

This example uses shell's here document syntax for entering multi-line input.

2. To edit the profile, run the following command:

```
/opt/f-secure/linuxsecurity/bin/lscctl set --prompt ic profiles /opt
```

This opens a text editor with the current value of the `/opt` profile specified as a template.

3. To remove all integrity checker profiles, run the following command to set `ic profiles` to an empty array:

```
/opt/f-secure/linuxsecurity/bin/lscctl set ic profiles '[]'
```

3.9 Command line usage

This section describes the commands that you can run from the command line in the product.

3.9.1 Starting and stopping services

The product contains a number of background services, which you can control using the `/opt/f-secure/fsbg/bin/master-switch` command.

You can use the `master-switch` command to control all the services related to the product. The syntax for the command is:

```
/opt/f-secure/fsbg/bin/master-switch ( on | off | status | enable | disable | is-enabled )
```

Note: WithSecure Elements Endpoint Protection controls whether or not the `master-switch` command can be used for turning off services. Change this using the [Tamper protection > Allow user to turn off all security features](#) setting in the Elements Endpoint Protection (EPP) portal.

Use the `off` sub-command to temporarily stop all WithSecure services:

```
/opt/f-secure/fsbg/bin/master-switch off
```

The `on` sub-command restarts the stopped services:

```
/opt/f-secure/fsbg/bin/master-switch on
```

To view the status of WithSecure services, use the `status` sub-command:

```
/opt/f-secure/fsbg/bin/master-switch status
```

The `disable` sub-command turns off all WithSecure services. The services stay turned off after you restart the computer. The `enable` sub-command turns on all WithSecure services again. Use the `is-enabled` sub-command to check if WithSecure services are turned on.

3.9.2 Scanning the computer manually from the command line

You can scan the computer for malware manually from the command line of a Linux host that has the product installed.

The `fsanalyze` program scans files with the privileges of the user who runs it. The user must have read access to files, and read and execute access to all directories to be scanned. To rename or remove infected files, the user must have write access to the directories where the files are located.

The `fsanalyze` command scans specified targets (files or directories) and reports any malicious code it detects.

If no files are specified on the command line, `fsanalyze` reads the standard input for content to analyze. Otherwise, every given file is analyzed and every given directory is scanned recursively, with the following exceptions:

- Special files (such as socket, FIFO, or device files), and files on the /proc and /sys special file systems are skipped.
- Recursive scanning does not follow symbolic links unless this is explicitly requested with the `--follow` command line option.
- Recursive scanning does not automatically cross file system boundaries. To scan files mounted on other file systems, specify the mount points explicitly on the command line.

Note: The exclusions that are set in the current policy for manual scanning are applied when you use manual scanning from the command line. If you have excluded any paths in the manual scanning policy, recursive scanning crosses file system boundaries.

Run the following command from the shell to start the manual scan:

```
/opt/f-secure/linuxsecurity/bin/fsanalyze [OPTIONS] [FILE...]
```

Usage: fsanalyze [OPTIONS] [FILE...]

Analyze each FILE for potential malicious content. If no file is specified, analyze standard input, in which case the only supported action is none.

-h, --help	Show help.
--malware=ACTION	Action to take when a file is detected as malware. If ACTION is <code>remove</code> the file is removed; if ACTION is <code>rename</code> the file is renamed adding the detection type as an extension; if ACTION is <code>none</code> , the infection is only reported on standard output. Default: <code>rename</code>
--pua=ACTION	Action to take when a file is detected as a potentially unwanted application. Options for the action are the same as for malware. Default: <code>none</code>
--suspected=ACTION	Action to take when a file is detected as suspicious. Options for the action are the same as for malware. Default: <code>none</code>
-L, --follow	Follow symbolic links.
-l, --list	List all scanned files.
--preserve-atime=VALUE	Preserve the access time of files that are scanned. Value can be <code>yes</code> or <code>no</code> . Default: <code>no</code>
-q, --quiet	Only output the scan results.
--scan-archives=VALUE	Enable archive scanning. Value can be <code>yes</code> or <code>no</code> . Default: <code>no</code>
--max-nested=NUMBER	Set the maximum allowed nesting level of scanned archives. Default: <code>5</code>
--detect-max-nested=VALUE	If enabled, treat archives that exceed the maximum depth as malware. Value can be <code>yes</code> or <code>no</code> . Default: <code>no</code>
--detect-encrypted-archives=VALUE	Treat encrypted archives as malware. Value can be <code>yes</code> or <code>no</code> . Default: <code>no</code>
--detect-pua=VALUE	If disabled, potentially unwanted applications are ignored. Value can be <code>yes</code> or <code>no</code> . Default: <code>yes</code>
--use-orsp=VALUE	If enabled, use the reputation service (ORSP) from Security Cloud. Value can be <code>yes</code> or <code>no</code> . Default: <code>yes</code>

-v, --version	Display program version and exit.
--quoting-style=STYLE	Set how to quote the output. If STYLE is <code>ur l</code> , the output is URL-encoded; if STYLE is <code>escape</code> , ASCII control characters in the output are escaped as <code><NAME></code> , and if the output is not redirected to a file, they are also highlighted; if STYLE is <code>l i t e r a l</code> , the output is printed as is. Default: <code>ur l</code>
--ignore-exclude-paths	Does not use the exclusions set from the current policy for manual scanning.

The scanning exit codes are as follows:

- 0 = scanning was completed with no errors, and no malware, PUA, or suspicious content was detected
- 1 = scanning failed for one or more files
- 2 = scanning was completed with no errors, but malware, PUA, or suspicious content was detected

3.9.3 Updating the product manually from the command line

You can update the product, the malware definition databases, and the scanning engines manually to their latest versions using the `lsctl` program.

You can update the product from the command line by running the following command as the root user:

```
/opt/f-secure/linuxsecurity/bin/lsctl update
```

This command checks for the availability of new updates over the network, and downloads and installs them on the system to make sure that everything is up to date.

The `--timeout` option can be used to specify the period of time to wait, in seconds, for the update to complete before letting it finish in the background. The value 0, which is used as the default if the option is not specified, disables the timeout.

Note: There are a few things to consider before and during the update:

- Before running the command, make sure that the product services are running.
- Once the update has started, downloading and installing the updates cannot be canceled. When using the `--timeout` option, the operation is left to complete in the background after the timeout.

Related concepts

[Starting and stopping services](#) on page 35

The product contains a number of background services, which you can control using the `/opt/f-secure/fsbg/bin/master-switch` command.

Related tasks

[Updating the product manually using an update archive](#)

3.9.4 Testing the antivirus protection

To test whether the product operates correctly, you can use a special test file that is detected as a virus.

The EICAR (EICAR is the European Institute of Computer Anti-virus Research) standard antivirus test file is detected by several antivirus programs. The Eicar info page can be found at <https://www.f-secure.com/v-descs/eicar.shtml>.

1. Download or create the EICAR test file.

- Download the EICAR test file from <https://www.f-secure.com/v-descs/eicar.shtml>, or
- Use any text editor to create the `eicar.com` file with the following single line in it:
`X50!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*`

2. Run the following command: `/opt/f-secure/linuxsecurity/bin/fsanalyze eicar.com`

The product should detect the EICAR test file as a virus.

3.9.5 Running the integrity checker from the command line

Integrity checking functionality is partially available on the command line.

To run the integrity checker from the command line on a Linux host:

1. Log in to the Linux host as root.
2. Run the following command from the shell: `/opt/f-secure/linuxsecurity/bin/fsic`

Usage:

- `fsic verify`: This checks each file in the baseline that is configured for the policy. It checks if the file has been tampered with and also scans the file for malware. The command prints every file that fails verification to the standard output.
- `fsic unprotect`: This allows baseline files to be modified.
- `fsic protect`: This regenerates the baseline index and prevents modifications to the baseline files.

3. Check the exit code for the command:

- 0 = No errors or failed verification results were encountered
- 1 = An error occurred that prevented the command from running successfully
- 2 = One or more files have been tampered with or contained malware, PUA, or suspicious content

Appendix

A

Alert severity levels

Alerts are divided into severity levels.

Severity level	Syslog priority	Description
Informational	info	Normal operating information from the host.
Warning	warning	A warning from the host. For example, an error when trying to read a file.
Error	err	Recoverable error on the host. For example, the virus definition database update is older than the previously accepted version.
Fatal Error	emerg	Unrecoverable error on the host that requires attention from the administrator. For example, a process fails to start or loading a kernel module fails.
Security alert	alert	A security alert on the host. For example, a virus-alert. The alert includes information of the infection and the performed operation.

Appendix B

Services and logs

Topics:

Lists of services and logs that the product uses.

- [Services installed with the product](#)
- [Logs used by the product](#)

B.1 Services installed with the product

The product installs several services that are used to handle various product features.

Note: The services listed here are subject to change without advance notice.

Important: Do not manually start or stop any of the services listed here.

Active services

f-secure-baseguard-accd.service	Real-time scanner service that handles file access notifications.
f-secure-baseguard-as.service	A BaseGuard facility for email spam scanning. Note: As of linuxsecurity channel version 12 . 0 . 286, this service is no longer running.
f-secure-baseguard-cleanup.service	Makes sure that older channel updates are cleaned up to save disk space.
f-secure-baseguard-doormand.service	Facilitates handling subscriptions, registration and cloud service lookups.
f-secure-baseguard-icap.service	The malware analysis service used for realtime, scheduled and manual scanning.
f-secure-baseguard-telemetry.service	Collects information for monitoring and analysis to keep the product features working as they should.
f-secure-baseguard-tokenverify.service	An OAuth 2 authentication manager used with HTTP APIs. This service is running, but is not used by the product.
fsbg-hotfixctl.service	WithSecure hotfix service that checks if there are hotfixes available. Usually in an inactive state.
fsbg-hotfixctl.timer	WithSecure hotfix service timer. Runs fsbg-hotfixctl.service to check hotfixes once per hour.
f-secure-linuxsecurity-fsacd.service	Maintains the file integrity checker baseline.
f-secure-linuxsecurity-scand.service	Manages manual and scheduled scans.
f-secure-linuxsecurity-lspmd.service	Locally distributes centrally managed settings to product services.
f-secure-linuxsecurity-statusd.service	Collects status and statistics information from product services and relays them to the central management agent (fsma2).
f-secure-linuxsecurity-webserver.service	Provides an internal interface for centrally managed settings.
fsbg-pmd.service	Locally distributes centrally managed settings to BaseGuard services.
fsbg-statusd.service	Collects status and statistics information from BaseGuard services and relays them to the central management agent (fsma2).
fsbg-updated.service	Schedules the installation of online channel updates.
fsma2.service	Handles centrally managed settings and communication (for example, with Policy Manager or F-Secure Elements Endpoint Protection).

Inactive services

These services are included during installation, but are not loaded by the product.

f-secure-baseguard-authorize.service	An OAuth 2 authorization server used with HTTP APIs.
f-secure-baseguard-sensor.service	Handles the functionality of F-Secure Elements Endpoint Detection and Response. If the Endpoint Detection and Response feature is enabled for the subscription key in use, this service is active, but it is inactive by default.
f-secure-linuxsecurity-rmmd.service	Provides status information for 3rd party RMM systems.
fsbg-enable-platform-selinux-policy.service	Turns on the SELinux confinement for product services to restrict their access to resources and actions based on the policies that are defined in the SELinux configuration.
fsbg-disable-platform-selinux-policy.service	Turns off the SELinux confinement for product services so that their access is not restricted.

Note: Both *fsbg-{enable,disable}-platform-selinux-policy.service* services are normally inactive, but on SELinux-capable systems, they may run temporarily after the initial product installation or after changes are made to the product configuration.

B.2 Logs used by the product

The product uses several logs for various product features.

Note: The format and contents of logs listed here are subject to change without advance notice.

systemd journal / syslog	All services log events into the syslog.
access.log	Records all log entries caused by scanning.
fsicapd.log	Records errors from the scanning service.
update.log	Records information related to updates.

Appendix C

Cloud services

The product connects to various services over the network, for example to check for updates to malware definitions.

The product may need to access any subdomain within the `fsapi.com` domain. Ensure it can connect to all addresses under `*.fsapi.com`.

Note: More detailed information about network addresses can be found in the [WithSecure Community page](#).