VMware vRealize Log Insight Agent
Administration Guide

vRealize Log Insight 3.0

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.
About vRealize Log Insight Agent Administration Guide

The vRealize Log Insight Agent Administration Guide describes how to install and configure VMware vRealize Log Insight Windows and Linux agents. It also includes information to troubleshoot Log Insight Agents.

For information about how to create configuration classes for agents with the Log Insight server, refer to the vRealize Log Insight Administration Guide.

Intended Audience

This information is intended for anyone who wants to install, configure, or troubleshoot Log Insight Agents. The information is written for experienced Windows or Linux system administrators who are familiar with virtual machine technology and datacenter operations.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation, go to http://www.vmware.com/support/pubs.
vRealize Log Insight agents collect events from log files on Linux and Windows machines and forwards them to the vRealize Log Insight server.

This chapter includes the following topics:

- “Overview of the Log Insight Windows Agent,” on page 7
- “Overview of the Log Insight Linux Agent,” on page 7

**Overview of the Log Insight Windows Agent**

The Log Insight Windows Agent collects events from Windows event channels and log files, and forwards them to the vRealize Log Insight server.

A Windows event channel is a pool for collecting related events in a Windows system. By default the Log Insight Windows Agent collects events from the Application, System, and Security channels.

In a Windows system, applications can store log data in flat text files on the file system. The Log Insight Windows Agent can monitor directories and collect events from flat text log files.

The Log Insight Windows Agent has a limit of 64 KB per request to the vRealize Log Insight server.

The Log Insight Windows Agent runs as a Windows service and starts immediately after installation. During and after installation, you can configure the following options for the Log Insight Windows Agent:

- Select the target vRealize Log Insight server to which the Log Insight Windows Agent forwards events.
- Select the communication protocol and port that the Log Insight Windows Agent uses.
- Add additional Windows event channels from which the Log Insight Windows Agent collects events.
- Select Windows directories to monitor and add flat log files to the collection.

The Log Insight Windows Agent requires Windows Vista or later, or Windows Server 2008 or later.

Verify that you have a copy of the Log Insight Windows Agent .msi file. See “Download the Windows agent .msi File,” on page 9

**Overview of the Log Insight Linux Agent**

The Log Insight Linux Agent collects events from log files on Linux machines and forwards them to the vRealize Log Insight server.

In a Linux system, applications can store log data in flat text files on the file system. The Log Insight Linux Agent can monitor directories and collect events from flat text log files.
The Log Insight Linux Agent runs as a daemon and starts immediately after installation. After installation, you can configure the following options:

- Select the target Log Insight server to which the Log Insight Linux Agent forwards events.
- Configure which directories the Log Insight Linux Agent monitors. By default the Log Insight Linux Agent is configured to collect messages and syslog files from the /var/log directory.

```
[filelog]messages
  directory=/var/log
  include=messages;messages.?
```

```
[filelog]syslog
  directory=/var/log
  include=syslog;syslog.?
```

The Log Insight Linux Agent supports the following distributions and versions.

- RHEL 5 Update 10, RHEL 6 Update 5
- SLES 11 SP3
- Ubuntu 12.04 LTS and 14.04 LTS

The Log Insight Linux Agent writes its own operation log files to /var/log/loginsight-agent/liagent_*.log. Log files are rotated when the Log Insight Linux Agent is restarted and when they reach a size of 10 MB. A combined limit of 50 MB are kept in rotation.

To download the Log Insight Linux Agent package, navigate to the Administration page of the vRealize Log Insight Web user interface, click Agents in the Management section, and click the appropriate package link.

If you implement a default installation of the Log Insight Linux Agent for a user without root privileges to use, the default configuration might create problems with the data collection. The agent does not log a warning that the subscription to the channel is unsuccessful. and files in the collection do not have read permissions. The message Inaccessible log file ... will try later is repeatedly added to the log. You can comment out the default configuration that is causing the problem or change the user permissions.

If you use an rpm or DEB package to install Linux agents, the init.d script named liagentd is installed as part of the package installation. The bin package adds the script, but does not register it. You can register the script manually.

You can verify that the installation was successful by running the (/sbin/)service liagentd status command.
The Log Insight Windows and Linux agents collect events from Windows and Linux machines and forwards them to the vRealize Log Insight server. You can install and configure parameters for the server, port, and protocol or chose to keep the default settings.

This chapter includes the following topics:

- “Download the Windows agent .msi File,” on page 9
- “Install the Log Insight Windows Agent with Default Configuration,” on page 9
- “Install and Configure the Log Insight Windows Agent,” on page 10
- “Deploy the Log Insight Windows Agent to Multiple Machines,” on page 11
- “Install or Update the vRealize Log Insight Linux Agent RPM package,” on page 13
- “Install or Update the vRealize Log Insight Linux Agent DEB package,” on page 14
- “Install the Log Insight Linux Agent Binary Package,” on page 15

Download the Windows agent .msi File

Before you install and configure the Windows agent, you need to download the Windows agent .msi file.

Procedure

1. Navigate to the Administration page of the vRealize Log Insight Web user interface.
2. In the Management section, click Agents.
3. Click the Download Log Insight Windows agent link.

What to do next

Use the .msi and .mst files to deploy the Log Insight Windows Agent.

Install the Log Insight Windows Agent with Default Configuration

You can install the Log Insight Windows Agent without configuring command-line parameters.

Prerequisites

- Verify that you have a copy of the Log Insight Windows Agent .msi file. See “Download the Windows agent .msi File,” on page 9
- Verify that you have permissions to perform installations and start services on the Windows machine.
Procedure

1. Log in to the Windows machine on which to install the vRealize Log Insight Windows agent.
2. Change to the directory where you have the vRealize Log Insight Windows agent .msi file.
3. Double-click the Log Insight Windows Agent .msi file, accept the terms of the License Agreement, and click Next.
4. Enter the IP address or host name of the vRealize Log Insight server and click Install.
   The wizard installs the Log Insight Windows Agent as an automatic Windows Service under the LocalSystem service account.
5. Click Finish.

What to do next
Configure the vRealize Log Insight Windows agent by editing liagent.ini file. See “Configure the Log Insight Windows Agent After Installation,” on page 17.

Install and Configure the Log Insight Windows Agent

You can install the Log Insight Windows Agent, specify a service account, and configure command-line parameters for the server, port, and protocol.

For MSI command-line options, see the Microsoft Developer Network (MSDN) Library Web site and search for MSI command-line options.

Prerequisites

- Verify that you have a copy of the Log Insight Windows Agent .msi file. See “Download the Windows agent .msi File,” on page 9
- Verify that you have permissions to perform installations and start services on the Windows machine.
- If you use the silent installation options /quiet or /qn, verify that you run the installation as an administrator. If you are not an administrator and run silent installation, the installation does not prompt for administrator privileges and fails. Use the logging option and parameters /lxv* file_name for diagnostic purposes.

Procedure

1. Log in to the Windows machine on which to install the vRealize Log Insight Windows agent.
2. Open a Command Prompt window.
3. Change to the directory where you have the vRealize Log Insight Windows agent .msi file.
4. Run the command to start the installation and replace Version-Build_Number with your version and build number.

   Drive:\path-to-msi_file>VMware-Log-Insight-Agent-Version-Build_Number.msi

   Drive:\path-to-msi_file>VMware-Log-Insight-Agent-30.msi.
5  (Optional) Specify a user service account for the Log Insight Windows Agent service to run under.

```
Drive:\path-to-msi_file>VMware-Log-Insight-Agent-* .msi SERVICEACCOUNT=domain\user SERVICEPASSWORD=user_password
```

**NOTE**  The account supplied in the SERVICEACCOUNT parameter is granted with the Log On As a Service right and full-write access to the %ProgramData%\VMware\Log Insight Agent directory. If the supplied account does not exist it is created. The username must not exceed 20 characters. If you do not specify a SERVICEACCOUNT parameter, the Log Insight Windows Agent service is installed under the LocalSystem service account.

6  (Optional) Enter the vRealize Log Insight server, port, and protocol.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVERHOST</td>
<td>IP address or host name of the vRealize Log Insight virtual appliance.</td>
</tr>
<tr>
<td>SERVERPROTO</td>
<td>Protocol that the agent uses to send events to the Log Insight server.</td>
</tr>
<tr>
<td>SERVERPORT</td>
<td>The port number depends on the value of SERVERPROTO. The default value</td>
</tr>
</tbody>
</table>

The command-line parameters correspond to hostname, proto, and port in the [server] section of the liagent.ini file.

7  Press Enter.

The command installs the Log Insight Windows Agent as a Windows service. The Log Insight Windows Agent service starts when you start the Windows machine.

**What to do next**

Verify that the command-line parameters you set are applied correctly in the liagent.ini file. See “Configure the Log Insight Windows Agent After Installation,” on page 17.

---

**Deploy the Log Insight Windows Agent to Multiple Machines**

You can deploy the Log Insight Windows Agent to multiple target machines in a Windows domain.

**Prepare to Deploy the Log Insight Windows Agent .mst file**

To specify installation parameters to be used during deployment, you create an .mst transform file. You can configure the Log Insight Windows Agent to send events to a vRealize Log Insight server, and to set the communication protocol, port, and user account for installing and starting the Log Insight Agent service.

**Prerequisites**

- Verify that you have a copy of the Log Insight Windows Agent .msi file. See “Download the Windows agent .msi File,” on page 9

**Procedure**

1  Open the Log Insight Windows Agent .msi file in the Orca editor and select Transform > New Transform.
2 Edit the Property table and add necessary parameters and values for a customized installation or upgrade.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVERHOST</td>
<td>IP address or host name of the vRealize Log Insight virtual appliance.</td>
</tr>
<tr>
<td>SERVERPROTO</td>
<td>Protocol that the agent uses to send events to the Log Insight server. The possible values are cfapi and syslog. Use the default cfapi setting.</td>
</tr>
<tr>
<td>SERVERPORT</td>
<td>Communication port that the agent uses to send events to the vRealize Log Insight server. The default values are 9543 for cfapi with SSL enabled, 9000 for cfapi with SSL disabled, 6514 for syslog with SSL enabled and 514 for syslog with SSL disabled.</td>
</tr>
<tr>
<td>SERVICEACCOUNT</td>
<td>User service account under which the Log Insight Windows Agent service will run. Note: The account supplied in the SERVICEACCOUNT parameter must have the Log On As a Service privilege and write access to %ProgramData%\VMware\Log Insight Agent directory so that the installer runs correctly. If you do not specify a SERVICEACCOUNT parameter, the vRealize Log Insight Windows agent service is installed under the LocalSystem service account.</td>
</tr>
<tr>
<td>SERVICEPASSWORD</td>
<td>Password of the user service account.</td>
</tr>
</tbody>
</table>

3 Select Transform > Generate Transform and save the .mst file.

What to do next
Use the .msi and .mst files to deploy the Log Insight Windows Agent.

Deploy Multiple Instances of the Log Insight Windows Agent
You can deploy multiple instances of the Log Insight Windows Agent on target computers in a Windows domain.

For more information about why you need to reboot the client machine twice, see support.microsoft.com/kb/305293.

Prerequisites
- Verify that you have an administrator account or an account with administrative privileges on the domain controller.
- Verify that you have a copy of the Log Insight Windows Agent .msi file. See “Download the Windows agent .msi File,” on page 9
- Familiarize yourself with the procedures described in http://support.microsoft.com/kb/887405 and http://support.microsoft.com/kb/816102.

Procedure
1 Log in to the domain controller as an administrator or a user with administrative privileges.
2 Create a distribution point and copy the Log Insight Windows Agent .msi file to the distribution point.
3 Open the Group Policy Management Console and create a Group Policy Object to deploy the Log Insight Windows Agent .msi file.
4 Edit the Group Policy Object for software deployment and assign a package.
5 (Optional) If you generated an .mst file before deployment, select the .mst configuration file on the Modifications tab of the GPO Properties window, and use the Advanced method to edit a Group Policy Object to deploy the .msi package.
6 (Optional) Upgrade the Log Insight Windows Agent.
   a Copy the upgrade .msi file to the distribution point.
   b Click the Upgrade tab on the Group Policy Object Properties window.
   c Add the initially installed version of the .msi file in the Packages that this package will upgrade section.

7 Deploy the Log Insight Windows Agent to specific security groups that include the domain users.

8 Close all Group Policy Management Console and Group Policy Management Editor windows on the domain controller and restart the client machines.
   If Fast Login Optimization is enabled, reboot the client machines twice.

9 Verify that Log Insight Windows Agent is installed on the client machines as a local service.
   If you configured SERVICEACCOUNT and SERVICEPASSWORD parameters for using an .mst file to deploy multiple instances of Log Insight Windows Agent, verify that Log Insight Windows Agent is installed on the client machines under the user account that you specified.

What to do next
   If the multiple instances of Log Insight Windows Agent is not successful, see “Mass Deployment of the Log Insight Windows Agent is Not Successful,” on page 52.

Install or Update the vRealize Log Insight Linux Agent RPM package

You can install or update the vRealize Log Insight Linux agent as a root or non-root user and you can set the target server during installation. After installation, you can verify the installed version.

Prerequisites
   ■ Log in as root or use sudo to run console commands.
   ■ The vRealize Log Insight Linux agent needs access to syslog and networking services to function. Install and run the vRealize Log Insight Linux agent on run levels 3 and 5. If you want the vRealize Log Insight Linux agent to work under other runlevels, configure the system appropriately.

Procedure
1 Open a console and run the rpm –i package_name command to install the vRealize Log Insight Linux agent.
   Replace package_name with the appropriate version.
   ```
   rpm –i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.rpm
   ```

**Note**

   sudo SERVERHOST=hostname rpm –i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.rpm

2 To set the target vRealize Log Insight server during installation run the sudo command and replace hostname with the IP address or hostname of the vRealize Log Insight server.
   ```
   sudo SERVERHOST=hostname rpm –i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.rpm
   ```

3 (Optional) To update the vRealize Log Insight Linux agent run the rpm –Uhv command.
   ```
   rpm –Uhv VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.rpm
   ```

**Note**
You can run other RPM commands such as –h, --hash, --version,--allfiles and so on during the installation, update, or uninstallation of the vRealize Log Insight Linux agent RPM package, but they are not supported.
4 (Optional) To install the vRealize Log Insight Linux agent as a non root user, run the sudo command.

```
sudo LIAGENTUSER=liagent rpm -i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.rpm
```

If the specified user does not exist, the vRealize Log Insight Linux agent creates the user account during the installation. The created account is not deleted after uninstallation. If you install the vRealize Log Insight Linux agent with the LIAGENTUSER=non_root_user parameter and try to upgrade with LIAGENTUSER=non_root_user2, a conflict occurs and warnings appear because non_root_user2 user does not have the permissions of the user non_root_user.

5 (Optional) Double click the appropriate version of the RPM package to install or update the vRealize Log Insight Linux agent.

6 (Optional) Verify the installed version by running the `rpm -qa | grep Log-Insight-Agent` command.

### Install or Update the vRealize Log Insight Linux Agent DEB package

When you install or update the vRealize Log Insight Linux agent DEB package, you can set the target server during installation and keep or replace the liagent.ini configuration file. After installation, you can verify the installed version.

#### Prerequisites

- Log in as root or use sudo to run console commands.
- Verify that the vRealize Log Insight Linux agent has access to syslog and networking services to function. By default, the vRealize Log Insight Linux agent runs on runlevels 2, 3, 4, and 5 and stops on runlevels 0, 1, and 6.

#### Procedure

1. Open a console and run the `dpkg -i package_name` command to install or update the vRealize Log InsightLinux agent.

   ```
   dpkg -i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.deb
   ```

2. To set the target vRealize Log Insight server during installation run the `sudo` command and replace `hostname` with the IP address or hostname of the vRealize Log Insight server.

   ```
   sudo SERVERHOST=hostname dpkg -i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.deb
   ```

   Unless you enabled the `--force-confold` flag during installation, whenever you update to a newer version, the system prompts you to keep or replace the liagent.ini configuration file. The following system message appears.

   ```
   Configuration file `/var/lib/loginsight-agent/liagent.ini'
   ==> Modified (by you or by a script) since installation.
   ==> Package distributor has shipped an updated version.
   What would you like to do about it? Your options are:
   Y or I  : install the package maintainer's version
   N or O  : keep your currently-installed version
   D      : show the differences between the versions
   Z      : start a shell to examine the situation
   The default action is to keep your current version.
   ** liagent.ini (Y/I/N/O/D/Z) [default=N] ?
   ```

3. (Optional) To preserve the existing configuration, use [default=N]. The additional parameters passed from the comand line are still applied.
(Optional) To run the vRealize Log Insight Linux agent as a non root user run the sudo command.

    sudo LIAGENTUSER=liagent dpkg -i VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER.deb

If the specified user does not exist, the vRealize Log Insight Linux agent creates the user account during the installation. The created account is not deleted after uninstallation. If you install the vRealize Log Insight Linux agent with the LIAGENTUSER=non_root_user parameter and try to upgrade with the LIAGENTUSER=non_root_user2 parameter, a conflict occurs and warnings appear because the non_root_user2 user does not have the permissions of the non_root_user user.

(Optional) Verify the installed version by running the dpkg -l | grep -i VMware-Log-Insight-Agent command.

Install the Log Insight Linux Agent Binary Package

Installing the binary package includes changing the .bin file to an executable file and then installing the agent.

Upgrading the .bin package is not officially supported. If you used the .bin package to install an existing Log Insight Linux Agent, make a backup copy of the liagent.ini file located in /var/lib/loginsight-agent directory to keep the local configuration. After you have a backup copy, manually uninstall the Log Insight Linux Agent. See “Uninstall the Log Insight Linux Agent bin package,” on page 46.

If you use the .bin package to install Linux agents, the init.d script named liagentd is installed as part of the package installation, but the package does not register the script. You can register the script manually. You can verify that the installation is successful by running (/sbin/)service liagentd status command.

Prerequisites

- Download and copy the Log Insight Linux Agent .bin package to the target Linux machine.
- Log in as root or use sudo to run console commands.
- Verify that the Log Insight Linux Agent has access to syslog and networking services.

Procedure

1. Open a console and run the chmod command to change the .bin file to an executable file.
   
   Replace `filename-version` with the appropriate version.
   
   ```bash
   chmod +x filename-version.bin
   ```

2. Run the `./filename-version.bin` command to install the agent.
   
   Replace `filename-version` with the appropriate version.

   ```bash
   sudo SERVERHOST=hostname ./filename-version.bin
   ```

3. To set the target vRealize Log Insight server during installation, run the sudo SERVERHOST=hostname ./filename-version.bin command.

   Replace `hostname` with the IP address or hostname of the vRealize Log Insight server.
(Optional) To run the Log Insight Linux Agent as a non root user run the sudo command.

```bash
sudo LIAGENTUSER=liagent ./filename-version.bin
```

If the specified user does not exist, the Log Insight Linux Agent creates the user account during the installation. The created account is not deleted after uninstallation. If you install the Log Insight Linux Agent with the LIAGENTUSER=non_root_user parameter and try to upgrade with the LIAGENTUSER=non_root_user2 parameter, a conflict occurs and warnings appear because the non_root_user2 user does not have the permissions of the non_root_user user.
After you have deployed an agent, you can configure it to send events to the vRealize Log Insight server that you select, specify communication protocols, and so on.

Use these instructions as required to configure your agents to your requirements.

- **Configure the Log Insight Windows Agent After Installation** on page 17
  You can configure the Log Insight Windows Agent after the installation. You must edit the `liagent.ini` file to configure Log Insight Windows Agent to send events to a vRealize Log Insight server of your choice, set communication protocol and port, add Windows event channels, and configure flat file log collection.

- **Configure the Log Insight Linux Agent** on page 27
  You can configure the Log Insight Linux Agent after you install it. The `liagent.ini` file is located in `/var/lib/loginsight-agent/`. Edit the file to configure the Log Insight Linux Agent to send events to a vRealize Log Insight server of your choice, set communication protocol and port, and configure flat file log collection.

- **Centralized Configuration of vRealize Log Insight Agents** on page 31
  You can configure multiple Windows or Linux vRealize Log Insight agents.

- **Parsing Logs** on page 33
  Agent-side log parsers extract structured data from raw logs before delivering to the vRealize Log Insight server. Using log parsers, vRealize Log Insight can analyze logs, extract information from them, and show those results on the server. Log parsers can be configured for both Windows and Linux vRealize Log Insight Agents.

**Configure the Log Insight Windows Agent After Installation**
You can configure the Log Insight Windows Agent after the installation. You must edit the `liagent.ini` file to configure Log Insight Windows Agent to send events to a vRealize Log Insight server of your choice, set communication protocol and port, add Windows event channels, and configure flat file log collection.

**Default Configuration of the Log Insight Windows Agent**
After installation, the `liagent.ini` file contains preconfigured default settings for the Log Insight Windows Agent.

**Log Insight Windows Agent liagent.ini Default Configuration**
If you use non-ASCII names and values, save the configuration as UTF-8.

The final configuration is this file joined with settings from the server to form the `liagent-effective.ini` file.
You may find it more efficient to configure the settings from the server's agents page.

```
[server]
hostname=LOGINSIGHT
; Hostname or IP address of your Log Insight server / cluster load balancer. Default:
;hostname=LOGINSIGHT

; Protocol can be cfapi (Log Insight REST API), syslog. Default:
;proto=cfapi

; Log Insight server port to connect to. Default ports for protocols (all TCP):
; syslog: 514; syslog with ssl: 6514; cfapi: 9000; cfapi with ssl: 9543. Default:
;port=9000

; SSL usage. Default:
;ssl=no
; Example of configuration with trusted CA:
;ssl=yes
;ssl_ca_path=/etc/pki/tls/certs/ca.pem

; Time in minutes to force reconnection to the server.
; This option mitigates imbalances caused by long-lived TCP connections. Default:
;reconnect=30

[logging]
; Logging verbosity: 0 (no debug messages), 1 (essentials), 2 (verbose with more impact on
; performance).
; This option should always be 0 under normal operating conditions. Default:
;debug_level=0

[storage]
; Max local storage usage limit (data + logs) in MBs. Valid range: 100-2000 MB.
;max_disk_buffer=200

; Uncomment the following sections to collect these channels.
; The recommended way is to enable Windows content pack from LI server.
;[winlog|Application]
;channel=Application

;[winlog|Security]
;channel=Security

;[winlog|System]
;channel=System
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proto</td>
<td>cfapi</td>
<td>Protocol that the agent uses to send events to the Log Insight server. The possible values are cfapi and syslog. Use the default cfapi setting.</td>
</tr>
<tr>
<td>hostname</td>
<td>LOGINSIGHT</td>
<td>IP address or host name of the vRealize Log Insight virtual appliance.</td>
</tr>
</tbody>
</table>
### Parameter | Value | Description
--- | --- | ---
**port** | 9543, 9000, 6514, and 514 | Communication port that the agent uses to send events to the vRealize Log Insight server. The default values are 9543 for cfapi with SSL enabled, 9000 for cfapi with SSL disabled, 6514 for syslog with SSL enabled and 514 for syslog with SSL disabled.

**ssl** | no | Enables or disables SSL. The default value is no.

**max_disk_buffer** | 200 | The maximum disk space in MB that the Log Insight Windows Agent uses to buffer events and its own logs. When the specified max_disk_buffer is reached, the agent begins to drop new incoming events.

**debug_level** | 0 | Defines the log details level. See “Define Log Details Level in the Log Insight Agents,” on page 48.

**channel** | Application, Security, System | The Application, Security, and System Windows Event Log channels are commented by default; the Log Insight Windows Agent does not collect logs from these channels. See “Collect Events from Windows Events Channels,” on page 21.

---

**Set Target vRealize Log Insight Server**

You can set or change the target vRealize Log Insight server that the vRealize Log Insight Windows agent sends event to, if you have not set the values during the installation process.

**Prerequisites**

- Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.
- If you have a vRealize Log Insight cluster with an enabled Integrated Load Balancer, see Enable Integrated Load Balancer for custom SSL certificate specific requirements.

**Procedure**

1. Navigate to the program data folder of the vRealize Log Insight Windows agent.
   
   `%ProgramData%\VMware\Log Insight Agent`

2. Open the `liagent.ini` file in any text editor.

3. Modify the following parameters and set the values for your environment.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssl</td>
<td>Enables or disables SSL. The default value is no.</td>
</tr>
<tr>
<td>reconnect</td>
<td>The time in minutes to force reconnection to the server. The default value is 30.</td>
</tr>
</tbody>
</table>

```ini
[server]
; Log Insight server hostname or ip address
; If omitted the default value is LOGINSIGHT
;hostname=LOGINSIGHT

; Set protocol to use:
; cfapi - Log Insight REST API
; syslog - Syslog protocol
; If omitted the default value is cfapi
;proto=cfapi

; Log Insight server port to connect to. If omitted the default value is:
; for syslog: 514
; for cfapi without ssl: 9000
; for cfapi with ssl: 9543
;port=9543

;ssl - enable/disable SSL.
; Possible values are yes or no. If omitted the default value is no.
;ssl=no

; Time in minutes to force reconnection to the server
; If omitted the default value is 30
;reconnect=30
```

4 Save and close the liagent.ini file.

**Example: Configuration**

The following configuration example sets a target vRealize Log Insight server.

```ini
[server]
hostname=LOGINSIGHT
; Hostname or IP address of your Log Insight server / cluster load balancer. Default:
;hostname=LOGINSIGHT

; Protocol can be cfapi (Log Insight REST API), syslog. Default:
;proto=cfapi

; Log Insight server port to connect to. Default ports for protocols (all TCP):
; syslog: 514; syslog with ssl: 6514; cfapi: 9000; cfapi with ssl: 9543. Default:
;port=9000

; SSL usage. Default:
;ssl=no
; Example of configuration with trusted CA:
;ssl=yes
;ssl_ca_path=/etc/pki/tls/certs/ca.pem
```
What to do next

You can configure additional SSL options for the vRealize Log Insight Windows agent. See Configure SSL Connection Between the Server and the Log Insight Agents.

Collect Events from Windows Events Channels

You can add a Windows event channel to the Log Insight Windows Agent configuration. The Log Insight Windows Agent will collect the events and send them to the vRealize Log Insight server.

Prerequisites

Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.

Procedure

1. Navigate to the program data folder of the vRealize Log Insight Windows agent.
   %ProgramData%\VMware\Log Insight Agent
2. Open the liagent.ini file in any text editor.
3. Add the following parameters and set the values for your environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[winlog</td>
<td>section_name]</td>
</tr>
<tr>
<td>channel</td>
<td>The full name of the event channel as shown in the Event Viewer built-in Windows application. To copy the correct channel name, right-click a channel in Event Viewer, select Properties and copy the contents of Full Name field.</td>
</tr>
<tr>
<td>enabled</td>
<td>An optional parameter to enable or disable the configuration section. The possible values are yes or no. The default value is yes.</td>
</tr>
<tr>
<td>tags</td>
<td>Optional parameter to add custom tags to the fields of collected events. Define tags using JSON notation. Tag names can contain letters, numbers, and underscores. A tag name can only begin with a letter or an underscore and cannot exceed 64 characters. Tag names are not case sensitive. For example, if you use tags=&quot;{&quot;tag_name1&quot; : &quot;Tag value 1&quot;, &quot;Tag_Name1&quot; : &quot;tag value 2&quot;}, Tag_Name1 will be ignored as a duplicate. You cannot use event_type and timestamp as tag names. Any duplicates within the same declaration are ignored.</td>
</tr>
<tr>
<td>whitelist, blacklist</td>
<td>Optional parameters to explicitly include or exclude log events. Note: The blacklist option only works for fields; it cannot be used to blacklist text.</td>
</tr>
<tr>
<td>exclude_fields</td>
<td>(Optional) A parameter to exclude individual fields from collection. You can provide multiple values as a semicolon separated list. For example, exclude_fields=EventId; ProviderName</td>
</tr>
</tbody>
</table>

4. Save and close the liagent.ini file.
**Example: Configurations**

See the following [winlog] configuration examples.

```
[winlog|Events_Firewall ]
channel=Microsoft-Windows-Windows Firewall With Advanced Security/Firewall
enabled=no

[winlog|custom]
channel=Custom
tags={"ChannelDescription": "Events testing channel"}
```

**Set up Windows Event Channel Filtering**

You can set up filters for Windows Event Channels to explicitly include or exclude log events.

You use the `whitelist` and `blacklist` parameters to evaluate a filter expression. The filter expression is a Boolean expression that consists of Windows event fields and operators.

**Note** The `blacklist` option only works for fields; it cannot be used to blacklist text.

- `whitelist` collects only log events for which the filter expression evaluates to non-zero. If you omit `whitelist`, the value is an implied 1.
- `blacklist` excludes log events for which the filter expression evaluates to non-zero. The default value is 0.

For a complete list of Windows event fields and operators see “Event Fields and Operators,” on page 23.

**Prerequisites**

Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.

**Procedure**

1. Navigate to the program data folder of the vRealize Log Insight Windows agent.
   ```
   %ProgramData%\VMware\Log Insight Agent
   ```
2. Open the `liagent.ini` file in any text editor.
3. Add a whitelist or blacklist parameter in the `[winlog]` section.
   For example
   ```
   [winlog|unique_section_name]
   channel = event_channel_name
   blacklist = filter_expression
   ```
4. Create a filter expression from Windows events fields and operators.
   For example
   ```
   whitelist = level > WINLOG_LEVEL_SUCCESS and level < WINLOG_LEVEL_INFO
   ```
5. Save and close the `liagent.ini` file.

**Example: Filter Configurations**

You can configure the agent to collect only error events, for example

```
[winlog|Security-Error]
channel = Security
whitelist = Level == WINLOG_LEVEL_CRITICAL or Level == WINLOG_LEVEL_ERROR
```
You can configure the agent to collect only VMware Network events from Application channel, for example

```
[winlog|VMwareNetwork]
channel = Application
whitelist = ProviderName == "VMnetAdapter" or ProviderName == "VMnetBridge" or ProviderName == "VMnetDHCP"
```

You can configure the agent to collect all events from Security channel except particular events, for example

```
[winlog|Security-Verbose]
channel = Security
blacklist = EventID == 4688 or EventID == 5447
```

### Event Fields and Operators

Use the Windows event fields and operators to build filter expressions.

#### Filter Expression Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>==, !=</td>
<td>equal and not equal. Use with both numeric and string fields.</td>
</tr>
<tr>
<td>&gt;=, &gt;, &lt;, &lt;=</td>
<td>greater or equal, greater than, less than, less than or equal. Use with numeric fields only.</td>
</tr>
<tr>
<td>&amp;,,</td>
<td>Bitwise AND, OR, XOR and complement operators. Use with numeric fields only.</td>
</tr>
<tr>
<td>and, or</td>
<td>Logical AND and OR. Use to build complex expressions by combining simple expressions.</td>
</tr>
<tr>
<td>not</td>
<td>Unary logical NOT operator. Use to reverse the value of an expression.</td>
</tr>
<tr>
<td>()</td>
<td>Use parentheses in a logical expression to change the order of evaluation.</td>
</tr>
</tbody>
</table>

#### Windows Event Fields

You can use the following Windows event fields in a filter expression.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>string</td>
</tr>
<tr>
<td>Text</td>
<td>string</td>
</tr>
<tr>
<td>ProviderName</td>
<td>string</td>
</tr>
<tr>
<td>EventSourceName</td>
<td>string</td>
</tr>
<tr>
<td>EventID</td>
<td>numeric</td>
</tr>
<tr>
<td>EventRecordID</td>
<td>numeric</td>
</tr>
<tr>
<td>Channel</td>
<td>string</td>
</tr>
<tr>
<td>UserID</td>
<td>string</td>
</tr>
<tr>
<td>Level</td>
<td>numeric</td>
</tr>
<tr>
<td>Task</td>
<td>numeric</td>
</tr>
</tbody>
</table>

You can use the following predefined constants:

- `WINLOG_LEVEL_SUCCESS = 0`
- `WINLOG_LEVEL_CRITICAL = 1`
- `WINLOG_LEVEL_ERROR = 2`
- `WINLOG_LEVEL_WARNING = 3`
- `WINLOG_LEVEL_INFO = 4`
- `WINLOG_LEVEL_VERBOSE = 5`
<table>
<thead>
<tr>
<th>Field name</th>
<th>Field type</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpCode</td>
<td>numeric</td>
</tr>
<tr>
<td>Keywords</td>
<td>numeric</td>
</tr>
</tbody>
</table>

You can use the following predefined bit masks

- WINLOG_KEYWORD_RESPONSETIME = 0x0001000000000000;
- WINLOG_KEYWORD_WDICONTEXT = 0x0002000000000000;
- WINLOG_KEYWORD_WDIDIAGNOSTIC = 0x0004000000000000;
- WINLOG_KEYWORD_SQM = 0x0008000000000000;
- WINLOG_KEYWORD_AUDITFAILURE = 0x0010000000000000;
- WINLOG_KEYWORD_AUDITSUCCESS = 0x0020000000000000;
- WINLOG_KEYWORD_CORRELATIONHINT = 0x0040000000000000;
- WINLOG_KEYWORD_CLASSIC = 0x0080000000000000;

Examples

Collect all critical, error and warning events

```
[wilog|app]
channel = Application
whitelist = level > WINLOG_LEVEL_SUCCESS and level < WINLOG_LEVEL_INFO
```

Collect only Audit Failure events from Security channel

```
[wilog|security]
channel = Security
whitelist = Keywords & WINLOG_KEYWORD_AUDITFAILURE
```

Collect Events from a Log File

You can configure the vRealize Log Insight Windows agent to collect events from one or more log files.

Collecting from Encrypted Folders

An agent is able to collect from encrypted folders. The Agent will collect from an encrypted folder only if it is run by the user who encrypted the folder.

Prerequisites

Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.

Procedure

1. Navigate to the program data folder of the vRealize Log Insight Windows agent.
   ```
   %ProgramData%\VMware\Log Insight Agent
   ```
2. Open the liagent.ini file in any text editor.
3. Add configuration parameters and set the values for your environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[filelog</td>
<td>section_name] directory</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>include</strong></td>
<td>(Optional) The name of a file name or a file mask (glob pattern) from which to collect data. You can provide values as a semicolon separated list. The default value is &quot;,&quot; which means that all files are included. The parameter is case sensitive.</td>
</tr>
<tr>
<td><strong>exclude</strong></td>
<td>(Optional) A file name or file mask (glob pattern) to exclude from collection. You can provide values as a semicolon separated list. The default value is empty, which means that no file is excluded.</td>
</tr>
<tr>
<td><strong>event_marker</strong></td>
<td>(Optional) A regular expression that denotes the start of an event in the log file. If omitted defaults to newline. The expressions you type must use the Perl regular expressions syntax.</td>
</tr>
<tr>
<td><strong>enabled</strong></td>
<td>(Optional) A parameter to enable or disable the configuration section. The possible values are yes or no. The default value is yes.</td>
</tr>
<tr>
<td><strong>charset</strong></td>
<td>(Optional) The character encoding of the log files that the agent monitors. The possible values are UTF-8, UTF-16LE, and UTF-16BE. The default value is UTF-8.</td>
</tr>
</tbody>
</table>
| **tags**        | (Optional) A parameter to add custom tags to the fields of collected events. Define tags using JSON notation. Tag names can contain letters, numbers, and underscores. A tag name can only begin with a letter or an underscore and cannot exceed 64 characters. Tag names are not case sensitive. For example, if you use tags="{"tag_name1": "tag value 1", "Tag_Name1": "tag value 2" }", Tag_Name1 will be ignored as a duplicate. You cannot use event_type and timestamp as tag names. Any duplicates within the same declaration are ignored. Tags can override the APP-NAME field, if the destination is a syslog server. For example, tags="{"appname": "VROPS"}.
| **exclude_fields** | (Optional) A parameter to exclude individual fields from collection. You can provide multiple values as a semicolon separated list. For example, exclude_fields=hostname; filepath                                      |

```
[filelog|section_name]
directory=path_to_log_directory
include=glob_pattern
```
Example: Configurations

[filelog|vCenterMain]
directory=C:\ProgramData\VMware\VMware VirtualCenter\Logs
include=vpdx-*\.log
exclude=vpdx-alert-*\.log;vpdx-profiler-*\.log
event_marker=^\d{4}-\d{2}-\d{2}[^A-Z]\d{2}:\d{2}:\d{2}\.\d{3}

[filelog|ApacheAccessLogs]
enabled=yes
directory=C:\Program Files (x86)\Apache Software Foundation\Apache2.2\logs
include=*.log
exclude=*_old.log
tags={"Provider" : "Apache"}

[filelog|MSSQL]
directory=C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Log
charset=UTF-16LE
event_marker=^[^\s]

Forward Events to the Log Insight Windows Agent

You can forward events from Windows machines to a machine where the Log Insight Windows Agent is running.

You can use Windows Event Forwarding to forward events from multiple Windows machines to a machine on which the Log Insight Windows Agent is installed. You can then configure the Log Insight Windows Agent to collect all forwarded events and send them to a vRealize Log Insight server.


Prerequisites

See “Collect Events from Windows Events Channels,” on page 21.

Procedure

1. Add a new section to the Log Insight Windows Agent configuration to collect events from the Windows event channel that receives forwarded events.
   
   The default channel name is ForwardedEvents.

2. Set up Windows Event Forwarding.

What to do next

Go to the vRealize Log Insight Web user interface and verify that forwarded events are arriving.
Configure the Log Insight Linux Agent

You can configure the Log Insight Linux Agent after you install it. The liagent.ini file is located in /var/lib/loginsight-agent/. Edit the file to configure the Log Insight Linux Agent to send events to a vRealize Log Insight server of your choice, set communication protocol and port, and configure flat file log collection.

Default Configuration of the vRealize Log Insight Linux Agent

After installation, the liagent.ini file contains preconfigured default settings for the Log Insight Windows Agent.

vRealize Log Insight Linux Agent liagent.ini Default Configuration

If you use non-ASCII names and values, save the configuration as UTF-8.

The final configuration is this file joined with settings from the server to form the liagent-effective.ini file.

You may find it more efficient to configure the settings from the server's agents page.

```
[server]
; Hostname or IP address of your Log Insight server / cluster load balancer. Default:
;hostname=LOGINSIGHT

; Protocol can be cfapi (Log Insight REST API), syslog. Default:
;proto=cfapi

; Log Insight server port to connect to. Default ports for protocols (all TCP):
; syslog: 514; syslog with ssl: 6514; cfapi: 9000; cfapi with ssl: 9543. Default:
;port=9000

; SSL usage. Default:
;ssl=no
; Example of configuration with trusted CA:
;ssl=yes
;ssl_ca_path=/etc/pki/tls/certs/ca.pem

; Time in minutes to force reconnection to the server.
; This option mitigates imbalances caused by long-lived TCP connections. Default:
;reconnect=30

[logging]
; Logging verbosity: 0 (no debug messages), 1 (essentials), 2 (verbose with more impact on performance).
; This option should always be 0 under normal operating conditions. Default:
;debug_level=0

[storage]
; Max local storage usage limit (data + logs) in MBs. Valid range: 100-2000 MB.
;max_disk_buffer=200

; Uncomment the appropriate section to collect system logs
; The recommended way is to enable the Linux content pack from LI server
;[filelog|syslog]
directory=/var/log
;include=messages;messages.?;syslog;syslog.?
### Set Target vRealize Log Insight Server

You can set or change the target vRealize Log Insight server that the vRealize Log Insight Linux agent sends events to.

**Prerequisites**

- Log in as `root` or use `sudo` to run console commands.
- Log in to the Linux machine on which you installed the vRealize Log Insight Linux agent, open a console and run `pgrep liagent` to verify that the vRealize Log Insight Linux agent is installed and running.
- If you have a vRealize Log Insight cluster with an enabled Integrated Load Balancer, see [Enable Integrated Load Balancer](#) for custom SSL certificate specific requirements.

**Procedure**

2. Modify the following parameters and set the values for your environment.

### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proto</td>
<td>cfapi</td>
<td>Protocol that the agent uses to send events to the Log Insight server. The possible values are cfapi and syslog. Use the default cfapi setting.</td>
</tr>
<tr>
<td>hostname</td>
<td>LOGINSIGHT</td>
<td>IP address or host name of the vRealize Log Insight virtual appliance.</td>
</tr>
<tr>
<td>port</td>
<td>9543</td>
<td>Communication port that the agent uses to send events to the vRealize Log Insight server. The default values are 9543 for cfapi with SSL enabled, 9000 for cfapi with SSL disabled, 6514 for syslog with SSL enabled and 514 for syslog with SSL disabled.</td>
</tr>
<tr>
<td>ssl</td>
<td>no</td>
<td>Enables or disables SSL. The default value is no.</td>
</tr>
<tr>
<td>max_disk_buffer</td>
<td>200</td>
<td>The maximum disk space in MB that the Log Insight Windows Agent uses to buffer events and its own logs. When the specified max_disk_buffer is reached, the agent begins to drop new incoming events.</td>
</tr>
<tr>
<td>debug_level</td>
<td>0</td>
<td>Defines the log details level. See “Define Log Details Level in the Log Insight Agents,” on page 48.</td>
</tr>
<tr>
<td>channel</td>
<td>Application, Security, System</td>
<td>The default Windows Event Log channels that the Log Insight Windows Agent collects. See “Collect Events from Windows Events Channels,” on page 21</td>
</tr>
</tbody>
</table>
### Parameter | Description
--- | ---
**port** | Communication port that the agent uses to send events to the vRealize Log Insight server. The default values are 9543 for cfapi with SSL enabled, 9000 for cfapi with SSL disabled, 6514 for syslog with SSL enabled and 514 for syslog with SSL disabled.

**ssl** | Enables or disables SSL. The default value is no.

**reconnect** | The time in minutes to force reconnection to the server. The default value is 30.

```ini
[server]
; Log Insight server hostname or ip address
; If omitted the default value is LOGINSIGHT
;hostname=LOGINSIGHT

; Set protocol to use:
; cfapi – Log Insight REST API
; syslog – Syslog protocol
; If omitted the default value is cfapi
; proto=cfapi

; Log Insight server port to connect to. If omitted the default value is:
; for syslog: 514
; for cfapi without ssl: 9000
; for cfapi with ssl: 9543
;port=9543

;ssl – enable/disable SSL.
; Possible values are yes or no. If omitted the default value is no.
;ssl=no

; Time in minutes to force reconnection to the server
; If omitted the default value is 30
;reconnect=30
```

3. Save and close the liagent.ini file.

**Example: Configuration**

```ini
[server]
hostname=LOGINSIGHT
; Hostname or IP address of your Log Insight server / cluster load balancer. Default:
;hostname=LOGINSIGHT

; Protocol can be cfapi (Log Insight REST API), syslog. Default:
;proto=cfapi

; Log Insight server port to connect to. Default ports for protocols (all TCP):
; syslog: 514; syslog with ssl: 6514; cfapi: 9000; cfapi with ssl: 9543. Default: 
;port=9000

; SSL usage. Default:
;ssl=no
; Example of configuration with trusted CA:
;ssl=yes
;ssl_ca_path=/etc/pki/tls/certs/ca.pem
```
What to do next

You can configure additional SSL options for the vRealize Log Insight Linux agent. See Configure SSL Connection Between the Server and the Log Insight Agents.

Collect Events from a Log File

You can configure the vRealize Log Insight Linux agent to collect events from one or more log files.

**Note** By default the vRealize Log Insight Linux agent collects hidden files created by programs or editors. The hidden file names start with a period. You can prevent the vRealize Log Insight Linux agent from collecting hidden files, by adding an exclude `exclude=^\.*`.

**Prerequisites**

- Log in as root or use sudo to run console commands.
- Log in to the Linux machine on which you installed the vRealize Log Insight Linux agent, open a console and run `pgrep liagent` to verify that the vRealize Log Insight Linux agent is installed and running.

**Procedure**

2. Add configuration parameters and set the values for your environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[filelog</td>
<td>section_name]</td>
</tr>
<tr>
<td>directory</td>
<td>The full path to the log file directory. You can define the same directory under one or more different configuration sections, to collect logs from the same file multiple times. This process makes it possible to apply different tags and filters to the same source of events. <strong>Note</strong> If you use exactly the same configurations for these sections, duplicated events are observed on the server side.</td>
</tr>
<tr>
<td>include</td>
<td>(Optional) The name of a file name or a file mask (glob pattern) from which to collect data. You can provide values as a semicolon separated list. The default value is <code>*</code>, which means that all files are included. The parameter is case sensitive. <strong>Note</strong> By default .zip and .gz files are excluded from collection. If you want to collect .zip and .gz files, add them using the <code>include</code> parameter. <strong>Important</strong> If you are collecting a rotated log file, use the <code>include</code> and <code>exclude</code> parameters to specify a glob pattern that matches both the primary and the rotated file. If the glob pattern matches only the primary log file, the vRealize Log Insight agents might miss events during rotation. The vRealize Log Insight agents automatically determine the correct order of rotated files and sends events to the Log Insight server in the right order. For example, if your primary log file is named <code>myapp.log</code> and rotated logs are <code>myapp.log.1</code>, <code>myapp.log.2</code> and so on you can use the following <code>include</code> pattern: <code>include= myapp.log;myapp.log.*</code></td>
</tr>
<tr>
<td>exclude</td>
<td>(Optional) A file name or file mask (glob pattern) to exclude from collection. You can provide values as a semicolon separated list. The default value is empty, which means that no file is excluded.</td>
</tr>
<tr>
<td>event_marker</td>
<td>(Optional) A regular expression that denotes the start of an event in the log file. If omitted defaults to newline. The expressions you type must use the Perl regular expressions syntax. <strong>Note</strong> Symbols, for example quotation marks (&quot;&quot;), are not treated as wrappers for regular expressions. They are treated as part of the pattern.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enabled</td>
<td>(Optional) A parameter to enable or disable the configuration section. The possible values are yes or no. The default value is yes.</td>
</tr>
<tr>
<td>charset</td>
<td>(Optional) The character encoding of the log files that the agent monitors. The possible values are UTF-8, UTF-16LE, and UTF-16BE. The default value is UTF-8.</td>
</tr>
<tr>
<td>tags</td>
<td>(Optional) A parameter to add custom tags to the fields of collected events. Define tags using JSON notation. Tag names can contain letters, numbers, and underscores. A tag name can only begin with a letter or an underscore and cannot exceed 64 characters. Tag names are not case sensitive. For example, if you use tags=&quot;{'tag_name1' : 'tag value 1', 'Tag_Name1' : 'tag value 2'}&quot;, Tag_Name1 will be ignored as a duplicate. You cannot use event_type and timestamp as tag names. Any duplicates within the same declaration are ignored. Tags can override the APP-NAME field, if the destination is a syslog server. For example, tags=&quot;{'appname' : 'VROPS'}&quot;.</td>
</tr>
<tr>
<td>exclude_fields</td>
<td>(Optional) A parameter to exclude individual fields from collection. You can provide multiple values as a semicolon separated list. For example, exclude_fields=hostname; filepath</td>
</tr>
</tbody>
</table>

```
[filelog|section_name]
directory=path_to_log_directory
include=glob_pattern
```

3 Save and close the liagent.ini file.

**Example: Configurations**

```
[filelog|messages]
directory=/var/log
include=messages;messages.?
```

```
[filelog|syslog]
directory=/var/log
include=syslog;syslog.?
```

```
[filelog|Apache]
directory=/var/log/apache2
include=*  
```

**Centralized Configuration of vRealize Log Insight Agents**

You can configure multiple Windows or Linux vRealize Log Insight agents.

Each vRealize Log Insight agent has a local configuration and a server-side configuration. The local configuration is stored in the liagent.ini file on the machine where the vRealize Log Insight agent is installed. The server-side configuration is accessible and editable, for example, in Windows from Administration > Agents in the Web user interface. The configuration of each vRealize Log Insight agent is composed of sections and keys. Keys have configurable values.

The vRealize Log Insight agents periodically poll the vRealize Log Insight server and receive the server-side configuration. The server-side configuration and the local configuration are merged and the result is the effective configuration. Each vRealize Log Insight agent uses the effective configuration as its operating configuration. Configurations merge section by section and key by key. The values in the server-side configuration override the values in the local configuration. The merging rules are the following:

- If a section is present only in the local configuration or only in the server-side configuration, this section and all its content become a part of the effective configuration.
If a section is present in both the local and server-side configuration, the keys in the section are merged according to the following rules:

- If a key is present only in the local configuration or only in the server-side configuration, the key and its value become a part of this section in the effective configuration.
- If a key is present in both the local configuration and the server-side configuration, the key becomes a part of this section in the effective configuration, and the value in the server-side configuration is used.

An Admin vRealize Log Insight user can apply centralized configuration to all vRealize Log Insight agents. For example, in Windows, you can navigate to the Administration page, and in the Management section, click Agents. Enter the configuration settings in the Agent Configuration box and click Save Configuration for All Agents. The configuration is applied to all the connected agents during the next poll cycle.

**Note** You can apply centralized configuration only to vRealize Log Insight agents that use the cfapi protocol.

See “Configure the Log Insight Windows Agent After Installation,” on page 17.

**An Example of Configuration Merging**

An example of merging local and server-side configuration of the Log Insight Windows Agent.

**Local Configuration**

You can have the following local configuration of the Log Insight Windows Agent.

```
[server]
proto=cfapi
hostname=HOST
port=9000

[winlog|Application]
channel=Application

[winlog|Security]
channel=Security

[winlog|System]
channel=System

[filelog|ApacheAccessLogs]
enabled=yes
directory=C:\Program Files (x86)\Apache Software Foundation\Apache2.2\logs
include=*.*
exclude=*_.old.log
event_marker=^\d{1,3}(\.\d{1,3}){3}\d{1,3} -- 
```

**Server-Side Configuration**

You can use the Administration > Agents page of the Web user interface to apply centralized configuration to all agents. For example, you can exclude and add collection channels, and change the default reconnect setting.

```
[server]
reconnect=20
```

```
[winlog|Security]
```
Effective Configuration

The effective configuration is a result of the merging of the local and the server-side configurations. The Log Insight Windows Agent is configured to:

- reconnect to the vRealize Log Insight server every 20 minutes
- continue to collect Application and System event channels
- stop collecting Security event channel
- start to collect Microsoft-Windows-DeviceSetupManager/Operational event channel
- continue to collect ApacheAccessLogs

Parsing Logs

Agent-side log parsers extract structured data from raw logs before delivering to the vRealize Log Insight server. Using log parsers, vRealize Log Insight can analyze logs, extract information from them, and show those results on the server. Log parsers can be configured for both Windows and Linux vRealize Log Insight Agents.
Configure Log Parsers

You can configure parsers for both FileLog and WinLog collectors.

Prerequisites

For the vRealize Log Insight Linux Agent:

- Log in as root or use `sudo` to run console commands.
- Log in to the Linux machine on which you installed the Log Insight Linux Agent, open a console and run `pgrep liagent` to verify that the Log Insight Linux Agent is installed and running.

For the vRealize Log Insight Windows Agent:

- Log in to the Windows machine on which you installed the Log Insight Windows Agent and start the Services manager to verify that the vRealize Log Insight service is installed.

Procedure

1. Navigate to the folder containing the `liagent.ini` file.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>/var/lib/loginsight-agent/</td>
</tr>
<tr>
<td>Windows</td>
<td>%ProgramData%\VMware\Log Insight Agent</td>
</tr>
</tbody>
</table>

2. Open the `liagent.ini` file in any text editor.

3. To configure a specific parser, define a parser section. `[parser|myparser]`

   Where `myparser` is an arbitrary name of the parser which can be referred from log sources. Parser section should refer to any built in (or any other defined) parser and configure that parser’s mandatory options and non-required options if needed.

   For example, `base_parser=csv` shows that `myparser` parser is derived from built-in parser `csv`. It expects that input logs consist of two fields which are separated with a semicolon.

   `[parser|myparser]`  
   
   `base_parser=csv`  
   `fields=field_name1,field_name2`  
   `delimiter=";"`

4. After defining `myparser`, refer to it from log sources `winlog` or `filelog`.

   `[filelog|some_csv_logs]`  
   `directory=D:\Logs`  
   `include=*.txt;*.txt.*`  
   `parser=myparser`

   The logs collected from `some_csv_logs` sources, for example from the `D:\Logs` directory, are parsed by `myparser` and extracted events appear on the server as `field_name1` and `field_name2` respectively.

   **Note**: The static logs in the `D:\Logs` directory are not get pulled into vRealize Log Insight by the agent. However, new files that are created in the `D:\Logs` directory are available in vRealize Log Insight.
5  Save and close the liagent.ini file.

**Common Options for Parsers**

You can configure common options for all parsers that produce named fields (except the timestamp parser).

Field names are restricted. The following field names are reserved and cannot be used as field names:

- event_type
- hostname
- source
- text
- timestamp

<table>
<thead>
<tr>
<th>Common Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>base_parser</td>
<td>The name of the base parser that this custom parser extends. It can be a built-in parser name or another customer parser name. This configuration key is mandatory.</td>
</tr>
<tr>
<td>field_decoder</td>
<td>Nested parsers specified as a JSON string where keys are the names of the field to apply nested parser to and the value is the name of the parser to use for that field. Each nested parser is applied to the appropriate field decoded by the base parser. Field decoders are useful when the value of a field is a complex value, for example, a timestamp.</td>
</tr>
<tr>
<td>field_rename</td>
<td>Renames extracted fields. This is a JSON string where keys are the original names of the fields and values are the new desired names of the fields. Note that field_decoder is always applied before field_rename. The order of these options in the INI file is not important. For clarity, specify field_decoder first.</td>
</tr>
<tr>
<td>next_parser</td>
<td>Name of the next parser to run. Allows multiple parsers to run sequentially on the same input. <strong>Note</strong>: Parsers process all consequent parsers defined by the next_parser keyword and may replace a field value already extracted by a previous parser.</td>
</tr>
<tr>
<td>exclude_fields</td>
<td>A list of semicolon separated field names to remove from the event before it is delivered to the server. This is applied before event filtering is performed so that the field that you excluding during parsing cannot be used in the filter condition.</td>
</tr>
<tr>
<td>debug</td>
<td>Yes or No option that enables debugging of particular parser. With debugging enabled, the parser performs detailed logging of input it receives, the operation it performed and the result it produced. The option applies per-section, that is, only to the parser that is defined by the particular section.</td>
</tr>
</tbody>
</table>

**CSV Log Parsers**

Comma-Separated Value (CSV) parsers are available in vRealize Log Insight. You can configure parsers for both FileLog and WinLog collectors.

**CSV Log Parser Configuration**

The parser name is csv.
The options that are available for the CSV parser are fields and delimiter.

Specify field names using the fields configuration option.

Double quotes surrounding the field value are optional, depending on the field content.

**fields Option**

The fields option specifies the names of the fields that exist in the log. The total number of the listed field names must be equal to the total number of comma-separated fields in the logs.

The fields option is mandatory for the CSV parser. If it is not specified, nothing is parsed.

Field names must be separated by commas, for example

```plaintext
fields = field_name1, field_name2, field_name3, field_name4
```

This definition assumes that the names field_name1, field_name2, field_name3 and field_name4 are assigned sequentially to the extracted fields.

If some fields must be omitted by the CSV parser, their names can be omitted from the list. For example,

```plaintext
fields = field_name1, , field_name3, field_name4
```

In this case, the parser extracts only the first, third and fourth fields from the event and subsequently assigns the names field_name1, field_name3 and field_name4 to them.

If the fields option does not specify a complete list of the fields in your logs, the parser returns an empty list. For example, if the log file contains field1, field2, field3, field4, and field5, but only fields= field1,field2,field3 is specified, the parser returns an empty fields list.

You cannot use `fields=*` for a CSV parser, because the parser returns an empty fields list. A complete list of fields must be specified, unless you need certain fields omitted as already described.

**delimiter Option**

The delimiter option specifies the delimiter for the parser to use.

The delimiter option is not mandatory.

By default, the CSV parser uses a comma as a delimiter. The delimiter must be enclosed in double quotes, `delimiter=";"`.

**Example: Parsing Logs Collected from winlog or filelog Sources**

To parse logs collected from either winlog or filelog sources, use the following configuration.

```plaintext
[filelog|some_csv_logs]
directory=D:\Logs
include=*.txt;*.txt.*
parser=myparser

[parser|myparser]
base_parser = csv
fields = timestamp,field_name1, field_name2, field_name3
delimiter = ";"
field_decoder={"timestamp": "tsp_parser"}
[parser|tsp_parser]
; timestamp is a built-in parser
base_parser=timestamp
; "format" is an option of timestamp parser
format=%Y-%m-%d %H:%M:%S
```
With this configuration, logs collected from some_csv_logs source (for example, from the directory=D:\Logs directory) are parsed by myparser. If the collected logs contain three values that are separated by a semicolon, the parsed events sequentially receive the field_name1, field_name2 and field_name3 names.

**Common Log Format (Apache) Log Parser**

Common Log Format (CLF) Apache parser are available in vRealize Log Insight. You can configure this parser for both FileLog and WinLog collectors.

**Common Log Format (Apache)**

The default CLF parser defines the following order and names of fields.

- host
- ident
- authuser
- datetime
- request
- statuscode
- bytes

Parser name: clf

The CLF parser-specific option is format.

**format Option**

The format option specifies the format with which Apache logs are generated. The option is not mandatory.

If no format is specified, use the following default common log format.

```%
%h %l %u %t "%r" %s %b
```

To parse other log formats, specify that format in the agent’s configuration. Parsed fields appear on the server side with following names.

**Note** In the cases in which a variable is required, if {VARNAME} is not provided in the configuration, the fields are ignored.

- `%a`: "remote_ip"
- `%A`: "local_ip"
- `%B`, `%b`: "response_size"
- `{VARNAME}C`: dependent on the name of variable specified in the format
- `%D`: "request_time_ms"
- `%E`: "error_status"
- `{VARNAME}E`: dependent on the name of variable specified in the format
- `%F`, `%f`: "file_name"
- `%h`: "remote_host"
- `%H`: "request_protocol"
- `{VARNAME}i`: dependent on the name of variable specified in the format
- `%k`: "keepalive_request_count"
- `%l`: "remote_log_name"
- `%L`: "request_log_id"
- `%M`: "log_message"(parser stops parsing of input log after reaching this specifier)
- `%m`: "request_method"
- `{VARNAME}n`: dependent on the name of variable specified in the format
- `{VARNAME}o`: dependent on the name of variable specified in the format
- `%p`: "server_port"
- `%P`: "process_id"
- `%q`: "query_string" (this is not generated by Apache, and might be excluded)
- `%r`: "request"
- `%R`: "response_handler"
- `%s`: "status_code"
- `%t`: "timestamp" will work as event timestamp on ingestion, engages timestamp parser. To override timestamp auto detection, date & time format can be specified in curly braces: %Y-%m-%d %H:%M:%S, see “Timestamp Parser,” on page 43 for more details.
"%T": "request_time_sec"
"%t": "date_time"  ("timestamp" will work as event timestamp on ingestion)
"%u": "remote_auth_user"
"%U": "requested_url"
"%v": "server_name"
"%V": "self_referential_server_name"
"%X": "connection_status"
"%I": "received_bytes"
"%O": "sent_bytes"
"%S": "transferred_size"

For example, to parse logs collected from either winlog or filelog sources with the CLF parser, specify the following configuration:

[filelog|clflogs]
directory=D:\Logs
parser=myclf

[parser|myclf]
format=%h %l %u %t "%r" %s

Using this configuration, logs that are collected from the clflogs source, for example from the directory=D:\Logs directory, are parsed by myclf. The myclf parser only parses those logs that were generated with the format described in the configuration.

Parsing Logs that were Generated Using CLF

To parse logs that were generated using CLF, you must define the corresponding format in the configuration. For example,

format=%h %l %u %{%a, %d %b %Y %H:%M:%S}t "%r" %>s %b "%{Referer}i" "%{User-Agent}i"

Fields that are not empty that use the specifiers %{Referer}i and %{User-Agent}i appear on the vRealize Log Insight server with the names referer and user_agent respectively.

Integrating the Timestamp Parser with the CLF Parser

You can parse Apache logs with a custom time format.

Access logs that have a custom time format as follows.

format = %h %l %u %t "%r" %s %b %M

If a custom time is not specified, the CLF parser attempts to deduce the time format automatically by running the automatic timestamp parser, otherwise the custom time format is used.

The supported custom time formats that are supported for error logs are:

<table>
<thead>
<tr>
<th>Custom Time Format</th>
<th>Description</th>
<th>Configuration Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>%ut</td>
<td>Current time including micro-seconds</td>
<td>format=[%{ut}t] [%l] [pid %P] [client %a] %M</td>
</tr>
<tr>
<td>%cut</td>
<td>Current time in compact ISO 8601 format, including micro-seconds</td>
<td>format=[%{cut}t] [%l] [pid %P] [client %a] %M</td>
</tr>
</tbody>
</table>

For a full list of supported timestamp specifiers, see “Timestamp Parser,” on page 43.
Example: Apache Access and Error Logs Configuration for Windows

Example: Examples For Apache Custom Log Formats

This example shows how you can format Apache v2.4 access and error logs configuration for Windows.

; -------------------------- EXAMPLE FOR DEFAULT ACCESS LOG --------------------------

; ACCESS LOG
;127.0.0.1 - - [13/May/2015:14:44:05 +0400] "GET /xampp/navi.php HTTP/1.1" 200 4023
"http://localhost/xampp/" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:37.0) Gecko/20100101
Firefox/37.0" ;format=%h %l %u %{%d/%b/%Y:%H:%M:%S %z}t "%r" %s %b "%{Referer}i" "%{User_agent}i"

; Section to collect Apache ACCESS logs
[filelog|clflogs-access]
   directory=C:\xampp\apache\logs
   include=acc*
   parser=clfparser_apache_access
   enabled=yes

; Parser to parse Apache ACCESS logs
[parser|clfparser_apache_access]
   debug=yes
   base_parser=clf
   format=%h %l %u %{%d/%b/%Y:%H:%M:%S %z}t "%r" %s %b "%{Referer}i" "%{User_agent}i"

; -------------------------- EXAMPLE FOR DEFAULT ERROR LOG --------------------------

; ERROR LOG
; [Wed May 13 14:37:27.042371 2015] [mpm_winnt:notice] [pid 5288] AH00418: Parent: Created child process 3480
;format=[%{%a %b %d %H:%M:%S%f %Y}t] [%m:%{severity}i] [pid %P:%{thread_id}i] %E: %M
;format=[%{%a %b %d %H:%M:%S%f %Y}t] [%m:%{severity}i] [pid %P] %E: %M

; Section to collect Apache ERROR logs
[filelog|clflogs-error]
   directory=C:\xampp\apache\logs
   include=err*
   parser=clfparser_apache_error
   enabled=yes

; Parser to parse Apache ERROR logs
[parser|clfparser_apache_error]
   debug=yes
   base_parser=clf
   format=[%{%a %b %d %H:%M:%S%f %Y}t] [%m:%{severity}i] [pid %P:%{thread_id}i] %E: %M
   next_parser=clfparser_apache_error2

; Parser to parse Apache ERROR logs
The provided names correspond to the combined log format. Apache error logs are also described using the above formatting keys, not the Apache error log format.

----- ACCESS LOG -----
1) Configure Apache for access log format (httpd.conf)
   LogFormat "%h %l %u %{%d-%b-%Y:%H:%M:%S}t "%r" %a %A %e %k %l %m %n %T %V %s %b "%\{Referer\}i" "%{User-Agent}i"" combined

2) Configure liagent.ini
   ACCESS LOG
   127.0.0.1 unknown - 21-May-2015:13:59:35 "GET /xampp/navi.php HTTP/1.1" 127.0.0.1 127.0.0.1 - 0
   unknown - GET - 1 localhost localhost 200 4023 "http://localhost/xampp/" "-

   ERROR LOG
   [Thu May 21 11:50:06 2015] [mpm_winnt:notice] [pid 5544] notice mpm_winnt 272
   localhost AHO0354; Child: Starting 150 worker threads.

----- ERROR LOG -----
1) Configure Apache for error log format (httpd.conf)
   ErrorLogFormat "[%t] [%m:%{severity}i] [pid %P] %F %l %m %T %V %E: %M"

2) Configure liagent.ini
   ERROR LOG
   [Thu May 21 11:50:06 2015] [mpm_winnt:notice] [pid 5544] notice mpm_winnt 272
   localhost AHO0354; Child: Starting 150 worker threads.

   }
Key/Value Data Parser

Key/Value (KV) parser is available in vRealize Log Insight. You can configure this parser for both FileLog and WinLog collectors.

Key/Value (KV)

[parser|simple_kvp]
base_parser = kvp
fields=*  

The kvp command finds and extracts all key=value matches from an arbitrary log message text. For example, the key-value log can be in the format

scope=local; abstract=false; lazyInit=false; autowireMode=0; dependencyCheck=0;

If no delimiters are specified in the configuration, the key-value parser uses default delimiters for parsing. Default delimiters are the space character, newline characters, comma, and semicolon characters. To change the default delimiters to specific ones, users must define them in the configuration in the format delimiter = ";^\|^\|\|\|\|\|\|\|\|\|". This definition means that each of the characters which are enclosed in the double quotes will be used as a delimiter.

The delimiter = ";^\|^\|\|\|\|\|\|\|\|
includes tab and new line characters as delimiters. If these characters are used, use escaping for them. For example, to define the backslash character as a delimiter, escape the backslash character when defining it as a delimiter, like this delimiter="\\".

With the key-value parser, you must specify the fields from which the values are to be extracted. For example, if the definition fields=name, lastname, country exists in the configuration, only the values with the specified keys are parsed and sent to the server.

You can use fields=* to parse all fields, if required.

Example: KV Parser Configuration

[parser|mykvp]
debug=yes
base_parser=kvp
delimiter="^\|\|\|\|\|\|\|\|\|\|
fields=*  
;OR fields=scope,abstract,lazyInit,autowireMode,dependencyCheck
field_decoder={"field1":"field1_parser1"}

[parser|field1_parser1]
base_parser=clf
format=[%{value1}i]
field_decoder={"value1":"field_parser2"}

Note the following information about the structure of the parser.
<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>debug</td>
<td>yes/no</td>
<td>Optional. The default, is no value is specified, is no. When the option is set to yes, you can view detailed logs of the parser ingestion in <code>liagent_&lt;date&gt;.log</code>.</td>
</tr>
<tr>
<td>field_decoder</td>
<td>Nested parser</td>
<td>Nested parsers are specified as a JSON string in which the keys are the names of the field to apply to the nested parser, and the value is the name of the parser to use for that field. Each nested parser is applied to the appropriate field, as decoded by the base parser. Field decoders are useful when the value of a key-value pair is a complex value such as a timestamp or a comma-separated list.</td>
</tr>
</tbody>
</table>

### Example: Nested Parser Examples

This is an example of a simple KV parser.

```plaintext
[filelog|MyLog]
directory=C:\<folder_name>|Parser_logs
include=*.*.log
parser=my_KVP_parser

[parsers|my_KVP_parser]
base_parser=kvp
fields=*  
```

This is an example of a complex KV parser.

```plaintext
[filelog|MyLog]
directory=C:\<folder_name>|Parser_logs
include=*.*.log
parser=my_KVP_parser

[parsers|my_KVP_parser]
base_parser=kvp
fields=*  
field_decoder={"field1": "field1_parser1"}

[parsers|field1_parser1]
base_parser=clf
format=[%{value1}i]
field_decoder={"value1": "field1_parser2"}
```

Note the following considerations.

- If the key in a key/value pair is not followed by an equals sign and a VALUE is not provided, the option is skipped, as with free text.
- An equals sign that is not followed by a value is treated as free text and is skipped.
- A value can be a string of characters that are surrounded by double quote characters, or it can be empty. Use a backslash for escaping special characters that are part of the value.
**Timestamp Parser**

The Timestamp parser does not produce fields but instead transforms its input from a string to a `ptime`-type value.

The only supported configuration option is `format`. For example, `format=%Y-%m-%d %H:%M:%S`.

Unlike the CLF parser, the Timestamp parser can parse time when there are no delimiters between time specifiers, for example `%A%B%d%H%M%S%Y%z`.

Format specifiers that are used by the Timestamp parser are:

- `%a`: Abbreviated weekday name, for example: Thu
- `%A`: Full weekday name, for example: Thursday
- `%b`: Abbreviated month name, for example: Aug
- `%B`: Full month name, for example: August
- `%d`: Day of the month, zero-padded (01-31), for example: 03
- `%e`: Day of the month, space-padded ( 1-31), for example: 3
- `%f`: Fractional seconds of time, for example: .036 'f' specifier assumes that '.' or ',' character should exist before fractional seconds and there is no need to mention that character in the format. If none of these characters precedes fractional seconds, timestamp wouldn't be parsed.
- `%H`: Hour in 24h format (00-23), for example: 14
- `%I`: Hour in 12h format (01-12), for example: 02
- `%m`: Month as a decimal number (01-12), for example: 08
- `%M`: Minute (00-59), for example: 55
- `%p`: AM or PM designation, for example: PM
- `%s`: Second (00-61), for example: 02
- `%y`: Year divided by 100 and truncated to integer (00-99), for example: 20
- `%g`: Week-based year, last two digits (00-99), for example, 01
- `%G`: Week-based year, for example, 2001
- `%j`: Day of the year (001-366), for example: 235
- `%w`: Weekday as a decimal number with Sunday as 0 (0-6), for example: 4
- `%u`: ISO 8601 weekday as number with Monday as 1 (1-7), for example: 4
- `%V`: ISO 8601 week number (00-53), for example: 34
- `%W`: Weekday as a decimal number with Sunday as 0 (0-6), for example: 4
- `%z`: ISO 8601 offset from UTC in timezone (1 minute=1, 1 hour=100)., for example: +100

Additional specifiers are accepted by the Timestamp parser, but their values are ignored and do not affect the parsed time:

- `%C`: Year divided by 100 and truncated to integer (00-99), for example: 20
- `%k`: Week-based year, last two digits (00-99), for example, 01
- `%j`: Day of the year (001-366), for example: 235
- `%w`: Weekday as a decimal number with Sunday as 0 (0-6), for example: 4
- `%u`: Week number with the first Monday as the first day of week one (00-53), for example: 34
- `%y`: Year, last two digits (00-99), for example: 01

If a `format` parameter is not defined, the Timestamp parser parses the timestamps using the default formats.

### Example: A Timestamp Parser with the Default Configuration

This example shows a Timestamp parser with a default configuration.

```
[parser|tsp_parser]
base_parser=timestamp
debug=yes
format=%Y-%m-%d %H:%M:%S%f
```
To integrate a Timestamp parser with other parsers, for example the CSV parser, specify the following configuration.

```
[parser|mycsv]
base_parser=csv
fields=timestamp,action,source_id,dest
field_decoder={"timestamp": "tsp_parser"}
```

When this configuration is defined, mycsv parser extracts the fields with the names that are specified in the configuration, and runs tsp_parser on the content of the timestamp field. If tsp_parser retrieves a valid timestamp, the server uses that timestamp for the log message.

### Automatic Log Parser

The automatic parser automatically detects the timestamp within the first 200 characters of a line. The format of auto-detected time stamps are the same as for the timestamp parser.

The automatic parser does not have any options. In addition to the automatic detection of the timestamp, the Key/Value parser runs on the log entry and automatically detects any existing key/value pairs in the logs and extracts the fields accordingly. For example,

```
[filelog|some_logs]
directory=/var/log
include=* 
parser=auto
```

As with other parsers, you can define a separate action for the automatic parser.

```
[filelog|kvplogs]
directory=C:\temp_logs\csv-itbm
include=*.txt
parser=myauto
[parser|myauto]
```

```
base_parser=auto
debug=yes
```

If you have debug enabled for the automatic parser, additional information about parsing is printed. For example, information about on which log the automatic parser was run, and which fields ere extracted from the log.
Uninstalling Log Insight Agents

Should you need to uninstall a vRealize Log Insight agent, follow the instructions that are appropriate to the agent package that you installed.

This chapter includes the following topics:
- “Uninstall the Log Insight Windows Agent,” on page 45
- “Uninstall the Log Insight Linux Agent RPM package,” on page 45
- “Uninstall the Log Insight Linux Agent DEB package,” on page 46
- “Uninstall the Log Insight Linux Agent bin package,” on page 46

Uninstall the Log Insight Windows Agent

You can uninstall the Log Insight Windows Agent.

Prerequisites

Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.

Procedure

1. Go to Control Panel > Programs and Features.
2. Select the VMware vRealize Log Insight Windows Agent and click Uninstall.

The uninstaller stops the VMware vRealize Log Insight Windows Agent service and removes its files from the system.

Uninstall the Log Insight Linux Agent RPM package

You can uninstall the Log Insight Linux Agent RPM package.

Prerequisites

- Log in as root or use sudo to run console commands.
- Log in to the Linux machine on which you installed the Log Insight Linux Agent, open a terminal console and run pgrep lioagent to verify that the VMware Log Insight Linux Agent is installed and running.
Procedure

- Run the following command replacing VERSION and BUILD_NUMBER with the version and build number of the installed agent.

```
rpm -e VMware-Log-Insight-Agent-VERSION-BUILD_NUMBER
```

The uninstaller stops the VMware Log Insight Linux Agent daemon and removes all its files except its own logs from the system.

Uninstall the Log Insight Linux Agent DEB package

You can uninstall the Log Insight Linux Agent DEB package.

Prerequisites

- Log in as root or use sudo to run console commands.
- Log in to the Linux machine on which you installed the Log Insight Linux Agent, open a terminal console and run `pgrep liagent` to verify that the VMware Log Insight Linux Agent is installed and running.

Procedure

- Run the following command

```
dpkg -P vmware-log-insight-agent
```

The uninstaller stops the VMware Log Insight Linux Agent daemon and removes all its files except its own logs from the system.

Uninstall the Log Insight Linux Agent bin package

You can uninstall the Log Insight Linux Agent .bin package.

Prerequisites

- Log in as root or use sudo to run console commands.
- Log in to the Linux machine on which you installed the Log Insight Linux Agent, open a terminal console and run `pgrep liagent` to verify that the VMware vRealize Log Insight Linux Agent is installed and running.

Procedure

1. Stop the Log Insight Linux Agent daemon by running the following command

```
sudo service liagentd stop or sudo /sbin/service liagentd stop for older Linux distributions.
```

2. Manually remove the Log Insight Linux Agent files

- `/usr/lib/loginsight-agent` - Daemon binary and license files directory.
- `/usr/bin/loginsight-agent-support` - Used to generate the support bundle for the Log Insight Linux Agent.
- `/var/lib/loginsight-agent` - Configuration files and database storage directory.
- `/var/log/loginsight-agent` - Log directory for the Log Insight Linux Agent.
- `/var/run/liagent/liagent.pid` - Log Insight Linux Agent PID file. If it is not deleted automatically, remove the file manually.
- `/etc/init.d/liagentd` - Script directory for the Log Insight Linux Agent daemon.
Troubleshooting the Log Insight Agents

Known troubleshooting information can help you diagnose and correct problems related to the operation of the Log Insight Agents.

This chapter includes the following topics:

- “Create a Support Bundle for the Log Insight Windows Agent,” on page 47
- “Create a Support Bundle for the Log Insight Linux Agent,” on page 48
- “Define Log Details Level in the Log Insight Agents,” on page 48
- “Administration UI Does Not Show Log Insight Agents,” on page 49
- “Log Insight Agents Do Not Send Events,” on page 49
- “Add an Outbound Exception Rule for the Log Insight Windows Agent,” on page 50
- “Allow Outbound Connections from the Log Insight Windows Agent in a Windows Firewall,” on page 51
- “Mass Deployment of the Log Insight Windows Agent is Not Successful,” on page 52
- “Installation of RPM Package Update Fails,” on page 52
- “Log Insight Agents Reject Self-Signed Certificate,” on page 53
- “vRealize Log Insight Server Rejects the Connection for Non-encrypted Traffic,” on page 53
- “Agent Service Fails on RPM-based systemd Systems Without Linux Standard Based Packages,” on page 54

Create a Support Bundle for the Log Insight Windows Agent

If the Log Insight Windows Agent does not operate as expected because of a problem, you can send a copy of the log and configuration files to VMware Support Services.

Procedure

1. Log in to the target machine where you installed the Log Insight Windows Agent.
2. Click the Windows Start button and then click VMware > Log Insight Agent - Collect support Bundle.
3. (Optional) If the shortcut is not available, navigate to the installation directory of the Log Insight Windows Agent and double-click loginsight-agent-support.exe.

   **NOTE:** The default installation directory is C:\Program Files (x86)\VMware\Log Insight Agent

The bundle is generated and saved as a .zip file in My Documents.
What to do next
Forward the support bundle to VMware Support Services as requested.

Create a Support Bundle for the Log Insight Linux Agent

If the Log Insight Linux Agent does not operate as expected because of a problem, you can send a copy of the log and configuration files to VMware Support Services.

Procedure
1. Log in to the target machine where you installed the Log Insight Linux Agent.
2. Run the following command.
   ```bash
   /usr/lib/loginsight-agent/bin/loginsight-agent-support
   ```
   The bundle is generated and saved as a .zip file in the current directory.

What to do next
Forward the support bundle to VMware Support Services as requested.

Define Log Details Level in the Log Insight Agents

You can edit the configuration file of the vRealize Log Insight Agent to change the logging level.

Prerequisites
For the Log Insight Linux Agent:
- Log in as root or use sudo to run console commands.
- Log in to the Linux machine on which you installed the Log Insight Linux Agent, open a console and run `pgrep liagent` to verify that the VMware vRealize Log Insight Linux Agent is installed and running.

For the Log Insight Windows Agent:
- Log in to the Windows machine on which you installed the vRealize Log Insight Windows agent and start the Services manager to verify that the VMware vRealize Log Insight agent service is installed.

Procedure
1. Navigate to the folder containing the `liagent.ini` file.

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>/var/lib/loginsight-agent/</td>
</tr>
<tr>
<td>Windows</td>
<td>%ProgramData%\VMware\Log Insight Agent</td>
</tr>
</tbody>
</table>

2. Open the `liagent.ini` file in any text editor.

   ```ini
   [logging]
   ; The level of debug messages to enable: 0..2
   debug_level=1
   ```

   **NOTE** The higher the debug level, the higher the impact it has on the vRealize Log Insight Agent. The default and recommended value is 0. Debug level 1 provides more information and is recommended for troubleshooting of most issues. Debug level 2 provides detailed information. Use levels 1 and 2 only when requested by VMware Support.
4. Save and close the liagent.ini file.

The log debug level is changed.

**Administration UI Does Not Show Log Insight Agents**

Information about the Log Insight Agents instances does not appear on the Agents page of the Administration UI.

**Problem**

After you install the Log Insight Agents you do not see the Log Insight Agents in the Agents page of the Administration UI.

**Cause**

The most common causes are network connectivity problems or incorrect configuration of the Log Insight Agents in the liagent.ini file.

**Solution**

- Verify that the Windows or Linux system that the Log Insight Agents are installed on has connectivity to the vRealize Log Insight server.
- Verify that the Log Insight Agents use the cfapi protocol.
  
  When using the syslog protocol the UI does not show Log Insight Windows Agents.
- View the contents of the Log Insight Agents log files located in the following directories:
  - Windows - %ProgramData%\VMware\Log Insight Agent\log
  - Linux - /var/log/loginsight-agent/

  Look for log messages that contain the phrases Config transport error: Couldn't resolve host name and Resolver failed. No such host is known.

**Log Insight Agents Do Not Send Events**

Incorrect configuration can prevent the Log Insight Agents from forwarding events to the vRealize Log Insight server.

**Problem**

The Log Insight Agents instances appears on the Administration > Agent page but no events appear in Interactive Analytics page from the Log Insight Agents host names.

**Cause**

Incorrect configuration can prevent the Log Insight Agents from forwarding events to the vRealize Log Insight server.
Solution
- For the Log Insight Windows Agent, try the following.
  - View the contents of the Log Insight Windows Agent log files located at %ProgramData%\VMware\Log Insight Agent\log. Look for log messages related to channel configuration that contain the phrases Subscribed to channel CHANNEL_NAME. The default channel names are Application, System, and Security.
  - If a channel is not configured correctly, you might see log messages similar to Could not subscribe to channel CHANNEL_NAME events. Error Code: 15007. The specified channel could not be found. Check channel configuration. You might see an error code number other than 15007.
  - If a flat file collection channel is not configured correctly, you might see messages like Invalid settings were obtained for channel 'CHANNEL_NAME'. Channel 'CHANNEL_NAME' will stay dormant until properly configured.
- For both Log Insight Windows Agent and Log Insight Linux Agent, try the following.
  - If no flat file collection channel is configured, you might see messages similar to Cannot find section 'filelog' in the configuration. The flat file log collector will stay dormant until properly configured.

The contents of the Log Insight Agents log files are located in the following directories.
- Windows - %ProgramData%\VMware\Log Insight Agent\log
- Linux - /var/log/loginsight-agent/

What to do next
For more information about configuring the Log Insight Agents see “Configure the Log Insight Windows Agent After Installation,” on page 17 and “Configure the Log Insight Linux Agent,” on page 27.

Add an Outbound Exception Rule for the Log Insight Windows Agent
Define an exception rule for unblocking the Log Insight Windows Agent in the Windows firewall.
The procedure applies to Windows Server 2008 R2 and later, and to Windows 7 and later.

Prerequisites
- Verify that you have an administrator account or an account with administrative privileges.

Procedure
1. Select Start > Run.
2. Type wf.msc and click OK.
3. Right-click Outbound rules in the left pane and click New Rule.
4. Select Custom and follow the wizard to set the following options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program</td>
<td>liwinsvc.exe</td>
</tr>
<tr>
<td>Service</td>
<td>LogInsightAgentService</td>
</tr>
<tr>
<td>Protocol and Ports</td>
<td>TCP 9000 for cfapi and 514 for syslog</td>
</tr>
</tbody>
</table>
On the Specify the profiles for which this rule applies page, select the appropriate network type.

- Domain
- Public
- Private

**Note** You can select all network types to make sure that the exception rule is active regardless of the network type.

**What to do next**

Go to the Log Insight Windows Agent log directory `%ProgramData%\VMware\Log Insight Agent\log` and open the latest log file. If recent events contain the messages `Config transport error: Couldn't resolve host name and Resolver failed. No such host is known`, restart the Log Insight Windows Agent service and the Windows machine.

**Note** The Log Insight Windows Agent service can take up to 5 minutes to reconnect to the server.

---

**Allow Outbound Connections from the Log Insight Windows Agent in a Windows Firewall**

Configure Windows firewall settings to allow outbound connections of the Log Insight Windows Agent to the vRealize Log Insight server.

After you install and start the Log Insight Windows Agent service, the Windows domain or local firewall may restrict the connectivity to the target vRealize Log Insight server.

The procedure applies to Windows Server 2008 R2 and later, and to Windows 7 and later.

**Prerequisites**

- Verify that you have an administrator account or an account with administrative privileges.

**Procedure**

1. Select **Start > Run**.
2. Type `wf.msc` and click **OK**.
3. In the **Actions pane** click **Properties**.
4. On the **Domain Profile** tab, select **Allow(default)** from the **Outbound connections** drop-down menu. If the computer is not connected to a domain, you can select **Private Profile** or **Public Profile**, depending on the network type the computer is connected to.
5. Click **OK**.

**What to do next**

Define an unblocking exception rule for the Log Insight Windows Agent in the Windows firewall. See “Add an Outbound Exception Rule for the Log Insight Windows Agent,” on page 50.
Mass Deployment of the Log Insight Windows Agent is Not Successful

The mass deployment of the Log Insight Windows Agent is not successful on target machines.

**Problem**

After performing a mass deployment on Windows domain machines by using Group Policy Objects, the Log Insight Windows Agent fails to install as a local service.

**Cause**

Group policy settings might prevent the Log Insight Windows Agent from being installed correctly.

**Solution**

1. Edit the Group Policy Object (GPO) settings and redeploy the Log Insight Windows Agent agent.
   a. Right-click the GPO, click **Edit** and navigate to **Computer Configuration > Policies > Administrative Templates > System > Logon**.
   b. Enable the **Always wait for the network at computer startup and logon** policy.
   c. Navigate to **Computer Configuration > Policies > Administrative Templates > System > Group Policy**.
   d. Enable the **Startup policy processing wait time**, and set **Amount of time to wait (in seconds)** to 120.
2. Run the **gpupdate /force /boot** command on target machines.

Installation of RPM Package Update Fails

Attempts to install an RPM package update fail when you use the Linux GUI.

**Problem**

Installing or updating the Log Insight Linux Agent RPM package fails when you use the GUI in RHEL and SUSE Linux distributions. You might see the error message `PK_TMP_DIR|dir://var/tmp/TmpDir.MtqOPs] Repository already exists.`

**Cause**

The cache and repository list might not clean after you install applications.

**Solution**

1. Log in to the Linux system where the Log Insight Linux Agent RPM is installed and open a system console.
2. Run the following commands as a **root** user.
   ```bash
   sudo zypper rr 2
   sudo zypper rr 1
   sudo zypper clean -a
   sudo zypper ref
   ```
3. Double-click the Log Insight Linux Agent RPM package to install the update.
Log Insight Agents Reject Self-Signed Certificate

The Log Insight Agents reject self-signed certificate.

Problem

The Log Insight Agents reject self-signed certificate and cannot establish connection with the server.

Note

If you experience connection problems with the Agent, you can check the detailed logs by changing the debug level for the vRealize Log Insight Agent to 1. See “Define Log Details Level in the Log Insight Agents,” on page 48.

Cause

The messages you see in the vRealize Log Insight Agent log have specific reasons.

<table>
<thead>
<tr>
<th>Message</th>
<th>Cause</th>
</tr>
</thead>
</table>
| Rejecting peer self-signed certificate. Public key doesn't match previously stored certificate's key. | ■ This might happen when the Log Insight Server certificate is replaced.  
■ This might happen if the HA enabled in cluster environment is configured with different self-signed certificates on vRealize Log Insight nodes. |
| Rejecting peer self-signed certificate. Have a previously received certificate which is signed by trusted CA. | There is a CA-signed certificate stored at Agent side. |

Solution

◆ Verify whether your target host name is a trusted vRealize Log Insight instance, and then manually delete the previous certificate from vRealize Log Insight Agent cert directory.

■ For Log Insight Windows Agent, go to C:\ProgramData\VMware\Log Insight Agent\cert.
■ For Log Insight Linux Agent, go to /var/lib/loginsight-agent/cert.

Note

Some platforms might use nonstandard paths for storing trusted certificates. The Log Insight Agents have an option to configure the path to trusted certificates store by setting the ssl_ca_path=<fullpath> configuration parameter. Replace <fullpath> with the path to the trusted root certificates bundle file. See Configure the Log Insight Agents SSL Parameters.

vRealize Log Insight Server Rejects the Connection for Non-encrypted Traffic

The vRealize Log Insight Server rejects the connection with the Log Insight Agents when you try to send non-encrypted traffic.

Problem

When you attempt to use cfapi to send nonencrypted traffic, the vRealize Log Insight Server rejects your connection. The following error message appears in the Log Insight Agent log.

403 Forbidden.

Cause

vRealize Log Insight is configured to accept only SSL connections, but the Log Insight Agents are configured to use non-SSL connection.
Solution
You can configure vRealize Log Insight Server to accept non-SSL connections or configure the Log Insight Agents to send data through SSL cfapi protocol connection.

Procedure
1. Configure vRealize Log Insight Server to accept non-SSL connection.
   a. Click the configuration drop-down menu icon and select Administration.
   b. Under Configuration, click SSL.
   c. Under the API Server SSL header, deselect the Require SSL Connection check box.
   d. Click Save.
2. Configure the Log Insight Agents to send data through SSL cfapi protocol connection.
   a. Navigate to the folder containing the liagent.ini file.
      - **Operating system** | **Path**
        - Linux | /var/lib/loginsight-agent/
        - Windows | %ProgramData%\VMware\Log Insight Agent
   b. Open the liagent.ini file in any text editor.
   c. Change the ssl key in the [server] section of the liagent.ini file to yes and the protocol to cfapi.
      ```
      proto=cfapi
      ssl=yes
      ```
   d. Save and close the liagent.ini file.

Agent Service Fails on RPM-based systemd Systems Without Linux Standard Based Packages

The agent service fails if LSB packages are not installed on RPM-based platforms using the systemd system and service manager (for example, RHEL-7, SLES-12).

Problem
For RPM platforms where Linux Standard Base (LSB) packages are not installed, the Agent service liagentd fails to stop, start, or restart. You might see the error message

Stopping liagentd (via systemctl): Warning: Unit file of liagentd.service changed on disk, 'systemctl daemon-reload' recommended

Cause
The LSB packages are not installed on the RPM platforms.

Solution
1. Log in to the Linux system where the Log Insight Linux Agent RPM is installed and open a system console.
2. Run the following commands as a root user.
   ```
   systemctl daemon-reload
   ```

The liagentd service stop, start, and restart functionality is fixed.
Index

A
add firewall exception 50
agent, install with parameters 10
agent configuration 17, 27
agent service fails 54
agent configuration example 32
agent multiple upgrade 12
agent multiple deployment 12
agent not showing 49
agent overview 7
agent-side parsers 33
agent-side parsers, configure 34
agents
    configuring 17
    installing 9
    uninstalling 45
allow firewall connection 51
auto parser 44

C
centralized configuration 31
change the log debug level 48
CLF parser, integrate timestamp parser 37
collect events from log file 24
common options for parsers 35
configure agent 11
configure agents 17
CSV parsers 35

D
default configuration 17, 27
default agent settings 17, 27
deployment to multiple machines 11

E
effective agent configuration 32
event forwarding, forward events to Log Insight Windows Agent 26
events, collect from Windows event channel 21

F
flat file collection 24
forward Windows events 26

g
glossary 5
Group Policy Object 12

I
incorrect agent configuration 49
install agents 9
install with default configuration 9
intended audience 5

K
key/value parser 41

L
Linux agent
    collect events from log file 30
    flat file collection 30
    install bin package 15
    install deb package 14
    install rpm package 13
    overview 7
    set target server 28
    uninstall bin package 46
    uninstall deb package 46
    uninstall rpm package 45
Linux Agent 52, 54
Linux agent configuration 27
Linux agent support bundle 48
log debug level 48
log details level 48
Log Insight Agents 53
Log Insight Linux Agent 53
Log Insight Windows Agent 53

M
mass deployment 11
mass deployment fails 52
merge configurations 52
multiple agents configuration 31

N
non-encrypted traffic 53

O
outbound connection 51
outbound exception rule 50

P
parsers
 auto 44
 CLF (Apache) 37
 key/value 41
 KV 41
 Timestamp 43

R
reject self-signed certificate 53
rpm package update 52

S
self-signed certificate 53
server rejects the connection 53
set target server 19
SSL connection 53
support bundle 48

T
timestamp parser, auto 44
Timestamp parser 43
troubleshoot agent configuration 49
troubleshoot Log Insight Agents 47
troubleshoot Log Insight Linux Agent 47
troubleshoot Log Insight Windows Agent 47
troubleshooting
agent service fails 54
 rpm package update 52
troubleshooting agent 49, 52

U
uninstall agent 45
uninstalling agents 45

W
Windows events channel, add 21
Windows agent, .msi file download 9
Windows agent support bundle, support bundle 47
Windows event channel
 add filter 22
 event fields and operators 23