1

Configuring Encrypted Remote Logging

- Introducing encrypted remote logging
- Creating the remote encrypted logging configuration

Introducing encrypted remote logging

This document describes how to configure encrypted remote logging on the BIG-IP system. This version of the BIG-IP software includes a new version of system logging software named **syslog-ng**. You can configure **syslog-ng** to send BIG-IP system log information to a remote logging host using an encrypted network connection. To do this, you create a port-forwarding SSH tunnel to the remote logging host, and configure **syslog-ng** on the BIG-IP system to send log messages through the SSH tunnel.

Getting started with encrypted remote logging

Before you attempt to create this configuration, you must meet the following conditions:

- **On the BIG-IP system**
  You must have a console with root access to the BIG-IP system.
- **On the logging host**
  You must have a console with root access to the logging host, the IP address, or host name of the logging host.
- **Both systems**
  You must have both systems connected to the same subnetwork.

WARNING

You should attempt this configuration only if you understand the risks associated with making changes to service startup scripts.

Creating the remote encrypted logging configuration

When creating a remote encrypted logging configuration, you must complete the following tasks:

- Review the SSH syntax required to create this configuration.
- Create a unique SSH identity key to identify and authorize the BIG-IP system.
- Edit the **syslog-ng** service startup script to create and destroy the SSH tunnels.
- Edit the remote logging host to accept **syslog-ng** messages through the SSH tunnel.
- Copy the unique SSH identity key to the remote logging host and append it to the authorized key file.
- Verify the logging configuration and restart **syslog-ng**.

Reviewing the SSH syntax required to create this configuration

This configuration requires that the BIG-IP system is able to establish an SSH connection to the remote logging host. On the BIG-IP system, use the **ssh** command to create the tunnel. Figure 1.1 is an example of the syntax required to create an SSH tunnel.

```
$ ssh -L <local tunnel port>:<remote log hostname>:<remote
tunnel port> \
  <remote user>@<remote log hostname> \
  -nNCxf \
  -i <key identity file>
```

Figure 1.1 *Establish an SSH tunnel from the BIG-IP system to the logging host.*

Table 1.1 contains detailed descriptions of the **ssh** syntax elements shown in Figure 1.1.

SSH syntax	Description
<local tunnel port>	The port SSH listens on for connections in order to forward them to **<remote log hostname>:<remote tunnel port>**.
<remote log hostname>	The IP address or FQDN of the remote logging server.
<remote tunnel port>	The port to which you want the SSH daemon on the remote logging server to forward connections.

Table 1.1 *Detailed syntax elements for configuring SSH.*

SSH syntax	Description
<remote user>	The user name that **ssh** attempts to authenticate, as on **<remote log hostname>**.
<key identity file>	A file name from which the identity (private key) for authentication is read.

Table 1.1 *Detailed syntax elements for configuring SSH.*

Creating a unique SSH key to identify and authorize the BIG-IP system

After you have reviewed the **ssh** syntax, use the **ssh** command to create the encrypted tunnel on the BIG-IP system, you must create a unique key on the BIG-IP system. The unique key is used to identify and authorize the BIG-IP system to the remote logging host.

Use the following command to create the file **syslog_tunnel_ID** and **syslog_tunnel_ID.pub**.

```
$ ssh -b 2048 -f syslog_tunnel_ID -t rsa -N "" -P ""
```

Use the following command to make **syslog_tunnel_ID** readable only by the **root** account:

```
$ chmod 600 syslog_tunnel_ID
```

Use the following command to make the public portion of the unique SSH ID named **syslog_tunnel_ID.pub** readable by all accounts:

```
$ chmod 644 syslog_tunnel_ID.pub
```

Copy **syslog_tunnel_ID** and **syslog_tunnel_ID.pub** into **/var/ssh** with the following command:

```
$ cp syslog_tunnel_ID* /var/ssh
```

Editing the syslog-ng start script to open and close the encrypted tunnel

Next change the **syslog-ng** start script, **/etc/init.d/syslog-ng**, so the encrypted tunnel is opened when **syslog-ng** starts up and closed when the service is restarted or stopped.

Before you edit the **syslog-ng** start script, save a backup copy to the root directory. Use the following command to save the backup to the root directory:

```
$ cp /etc/init.d/syslog-ng /root/syslog-ng.backup
```

After you save a backup of the **syslog-ng**, edit the startup script **/etc/init.d/syslog-ng** to automatically create a SSH tunnels when **syslog-ng** is started, or closed when **syslog-ng** is restarted or stopped.

The example configuration in this document demonstrates how to create a tunnel to a host using the following IP addresses and ports:

- IP address of **10.0.0.100**
- Local tunnel port of **5140**
- Remote tunnel port of **5140**
- User name **logger** on host **10.0.0.100**.

Start by adding syntax below the line that reads **start)**. Figure 1.2 is an example of what the section of the **syslog-ng.conf** looks like after you add the new syntax. In this example, the syntax you need to add is shown with bold text).

```
start)
      ssh -L 5140:10.0.0.100:5140 \
      logger@10.0.0.100 -nNCxf \
      -i var/ssh/syslog_tunnel_ID
       echo -n "Starting $INIT_NAME: "
       daemon --check $INIT_PROG "$INIT_PROG $INIT_OPTS"
```

Figure 1.2 *The syntax to add below the start) line.*

Next, add syntax below the line that reads **stop)**. Figure 1.3 shows the syntax you need to add in bold text.

```
stop)
       for sshTunnel in \
          `ps -ewo "%p!%a" | \
           grep ssh | \
           grep syslog_tunnel_ID | \
           grep -v grep | \
           cut -f 1 -d !`; do
           if [ -n "$sshTunnel" -a $sshTunnel -gt 10 ]; then
               echo " -- Shutting down SSH tunnel with process $sshTunnel"
               kill -TERM $sshTunnel
           fi
       done
       echo -n "Stopping $INIT_NAME: "
```

Figure 1.3 *The syntax to add below the stop) line*

Editing syslog-ng to log messages on the remote logging host

After you add the syntax to open and close SSH tunnels, you can edit the **syslog-ng** configuration to log messages to the remote machine. To do this, you need to create source and filter configuration blocks based on the local environment.

Using the example IP addresses and ports used in the example in the previous section, you would edit the **syslog-ng.conf** file to look like the **syslog-ng.conf** in Figure 1.4.

```
# capture all messages
filter f_catchall {
   level(debug...emerg);
};

# send message to localhost through tcp port 5140
destination d_remoteLogTunnel {
   tcp("127.0.0.1" port(5140););
};

# Combine everything to actually perform logging
log {
   source(local);
   filter(f_catchall);
   destination(d_remoteLogTunnel);
};
```

Figure 1.4 *The syslog-ng.conf example configuration*

Copying the unique SSH identity to the remote logging host and appending it to the authorized keys file

After you have edited the **syslog-ng.conf** to log messages on the remote logging host, you must copy the unique SSH identity to the remote logging host. To do this, copy the **syslog_tunnel_ID.pub** to the remote syslog server, and append this key to the **authorized_keys** file found in the **.ssh** folder under the home directory of the user that you want to use to capture remote log messages.

```
$ cat syslog_tunnel_ID.pub >> ~logger/.ssh/authorized_keys
```

Note

The following instructions are given as examples. The actual process for setting up the new SSH key to be automatically authorized, and configuring the syslog service may be different.

Verify that the logging facility is configured and ready to receive syslog messages on the **<remote tunnel port>**. If the remote logging host uses **syslog-ng**, you need to add a source configuration block like the one in Figure 1.5:

```
source remote {
   tcp(ip(10.0.0.100) port(5140));
};
```

Figure 1.5 *Remote logging host source identification block.*

In addition to the source identification block, you also need to add filter, destination, and log configuration blocks to use the data from the source **remote** as required by your application.

Verifying the logging configuration and restarting syslog-ng

Finally, verify that the SSH connection is functional and restart the syslog-ng service.

To verify the configuration from a command line and restart the syslog-ng service

1. Log in as **root** to the BIG-IP system.
2. Make an SSH connection to the remote logging host using the new identity key you created.

   ```
   # ssh logger@10.0.0.100 -i /var/shh/syslog_tunnel_ID
   ```

   If everything is configured correctly, you should be able to get shell access to the remote logging host without being challenged for a password. By adding the new identity key to the remote host's **authorized_keys** file, the key is used to authenticate the BIG-IP system).
3. Exit from the SSH session to the BIG-IP system command line.
4. Restart the **syslog-ng** service by typing the following command:

   ```
   $ /etc/init.d/syslog-ng restart
   ```

   The BIG-IP system should now be sending log messages to your remote host.

1

Configuring Encrypted Remote Logging

- Introducing encrypted remote logging
- Creating the remote encrypted logging configuration

Introducing encrypted remote logging

This document describes how to configure encrypted remote logging on the BIG-IP system. This version of the BIG-IP software includes a new version of system logging software named **syslog-ng**. You can configure **syslog-ng** to send BIG-IP system log information to a remote logging host using an encrypted network connection. To do this, you create a port-forwarding SSH tunnel to the remote logging host, and configure **syslog-ng** on the BIG-IP system to send log messages through the SSH tunnel.

Getting started with encrypted remote logging

Before you attempt to create this configuration, you must meet the following conditions:

- **On the BIG-IP system**
  You must have a console with root access to the BIG-IP system.
- **On the logging host**
  You must have a console with root access to the logging host, the IP address, or host name of the logging host.
- **Both systems**
  You must have both systems connected to the same subnetwork.

WARNING

You should attempt this configuration only if you understand the risks associated with making changes to service startup scripts.

Creating the remote encrypted logging configuration

When creating a remote encrypted logging configuration, you must complete the following tasks:

- Review the SSH syntax required to create this configuration.
- Create a unique SSH identity key to identify and authorize the BIG-IP system.
- Edit the **syslog-ng** service startup script to create and destroy the SSH tunnels.
- Edit the remote logging host to accept **syslog-ng** messages through the SSH tunnel.
- Copy the unique SSH identity key to the remote logging host and append it to the authorized key file.
- Verify the logging configuration and restart **syslog-ng**.

Reviewing the SSH syntax required to create this configuration

This configuration requires that the BIG-IP system is able to establish an SSH connection to the remote logging host. On the BIG-IP system, use the **ssh** command to create the tunnel. Figure 1.1 is an example of the syntax required to create an SSH tunnel.

```
$ ssh -L <local tunnel port>:<remote log hostname>:<remote
tunnel port> \
  <remote user>@<remote log hostname> \
  -nNCxf \
  -i <key identity file>
```

Figure 1.1 *Establish an SSH tunnel from the BIG-IP system to the logging host.*

Table 1.1 contains detailed descriptions of the **ssh** syntax elements shown in Figure 1.1.

SSH syntax	Description
<local tunnel port>	The port SSH listens on for connections in order to forward them to **<remote log hostname>:<remote tunnel port>**.
<remote log hostname>	The IP address or FQDN of the remote logging server.
<remote tunnel port>	The port to which you want the SSH daemon on the remote logging server to forward connections.

Table 1.1 *Detailed syntax elements for configuring SSH.*

SSH syntax	Description
<remote user>	The user name that **ssh** attempts to authenticate, as on **<remote log hostname>**.
<key identity file>	A file name from which the identity (private key) for authentication is read.

Table 1.1 *Detailed syntax elements for configuring SSH.*

Creating a unique SSH key to identify and authorize the BIG-IP system

After you have reviewed the **ssh** syntax, use the **ssh** command to create the encrypted tunnel on the BIG-IP system, you must create a unique key on the BIG-IP system. The unique key is used to identify and authorize the BIG-IP system to the remote logging host.

Use the following command to create the file **syslog_tunnel_ID** and **syslog_tunnel_ID.pub**.

```
$ ssh -b 2048 -f syslog_tunnel_ID -t rsa -N "" -P ""
```

Use the following command to make **syslog_tunnel_ID** readable only by the **root** account:

```
$ chmod 600 syslog_tunnel_ID
```

Use the following command to make the public portion of the unique SSH ID named **syslog_tunnel_ID.pub** readable by all accounts:

```
$ chmod 644 syslog_tunnel_ID.pub
```

Copy **syslog_tunnel_ID** and **syslog_tunnel_ID.pub** into **/var/ssh** with the following command:

```
$ cp syslog_tunnel_ID* /var/ssh
```

Editing the syslog-ng start script to open and close the encrypted tunnel

Next change the **syslog-ng** start script, **/etc/init.d/syslog-ng**, so the encrypted tunnel is opened when **syslog-ng** starts up and closed when the service is restarted or stopped.

Before you edit the **syslog-ng** start script, save a backup copy to the root directory. Use the following command to save the backup to the root directory:

```
$ cp /etc/init.d/syslog-ng /root/syslog-ng.backup
```

After you save a backup of the **syslog-ng**, edit the startup script **/etc/init.d/syslog-ng** to automatically create a SSH tunnels when **syslog-ng** is started, or closed when **syslog-ng** is restarted or stopped.

The example configuration in this document demonstrates how to create a tunnel to a host using the following IP addresses and ports:

- IP address of **10.0.0.100**
- Local tunnel port of **5140**
- Remote tunnel port of **5140**
- User name **logger** on host **10.0.0.100**.

Start by adding syntax below the line that reads **start)**. Figure 1.2 is an example of what the section of the **syslog-ng.conf** looks like after you add the new syntax. In this example, the syntax you need to add is shown with bold text).

```
start)
      ssh -L 5140:10.0.0.100:5140 \
      logger@10.0.0.100 -nNCxf \
      -i var/ssh/syslog_tunnel_ID
       echo -n "Starting $INIT_NAME: "
       daemon --check $INIT_PROG "$INIT_PROG $INIT_OPTS"
```

Figure 1.2 *The syntax to add below the start) line.*

Next, add syntax below the line that reads **stop)**. Figure 1.3 shows the syntax you need to add in bold text.

```
stop)
       for sshTunnel in \
          `ps -ewo "%p!%a" | \
           grep ssh | \
           grep syslog_tunnel_ID | \
           grep -v grep | \
           cut -f 1 -d !`; do
           if [ -n "$sshTunnel" -a $sshTunnel -gt 10 ]; then
               echo " -- Shutting down SSH tunnel with process $sshTunnel"
               kill -TERM $sshTunnel
           fi
       done
       echo -n "Stopping $INIT_NAME: "
```

Figure 1.3 *The syntax to add below the stop) line*

Editing syslog-ng to log messages on the remote logging host

After you add the syntax to open and close SSH tunnels, you can edit the **syslog-ng** configuration to log messages to the remote machine. To do this, you need to create source and filter configuration blocks based on the local environment.

Using the example IP addresses and ports used in the example in the previous section, you would edit the **syslog-ng.conf** file to look like the **syslog-ng.conf** in Figure 1.4.

```
# capture all messages
filter f_catchall {
   level(debug...emerg);
};

# send message to localhost through tcp port 5140
destination d_remoteLogTunnel {
   tcp("127.0.0.1" port(5140););
};

# Combine everything to actually perform logging
log {
   source(local);
   filter(f_catchall);
   destination(d_remoteLogTunnel);
};
```

Figure 1.4 *The syslog-ng.conf example configuration*

Copying the unique SSH identity to the remote logging host and appending it to the authorized keys file

After you have edited the **syslog-ng.conf** to log messages on the remote logging host, you must copy the unique SSH identity to the remote logging host. To do this, copy the **syslog_tunnel_ID.pub** to the remote syslog server, and append this key to the **authorized_keys** file found in the **.ssh** folder under the home directory of the user that you want to use to capture remote log messages.

```
$ cat syslog_tunnel_ID.pub >> ~logger/.ssh/authorized_keys
```

Note

The following instructions are given as examples. The actual process for setting up the new SSH key to be automatically authorized, and configuring the syslog service may be different.

Verify that the logging facility is configured and ready to receive syslog messages on the **<remote tunnel port>**. If the remote logging host uses **syslog-ng**, you need to add a source configuration block like the one in Figure 1.5:

```
source remote {
   tcp(ip(10.0.0.100) port(5140));
};
```

Figure 1.5 *Remote logging host source identification block.*

In addition to the source identification block, you also need to add filter, destination, and log configuration blocks to use the data from the source **remote** as required by your application.

Verifying the logging configuration and restarting syslog-ng

Finally, verify that the SSH connection is functional and restart the syslog-ng service.

To verify the configuration from a command line and restart the syslog-ng service

1. Log in as **root** to the BIG-IP system.
2. Make an SSH connection to the remote logging host using the new identity key you created.

   ```
   # ssh logger@10.0.0.100 -i /var/shh/syslog_tunnel_ID
   ```

   If everything is configured correctly, you should be able to get shell access to the remote logging host without being challenged for a password. By adding the new identity key to the remote host's **authorized_keys** file, the key is used to authenticate the BIG-IP system).
3. Exit from the SSH session to the BIG-IP system command line.
4. Restart the **syslog-ng** service by typing the following command:

   ```
   $ /etc/init.d/syslog-ng restart
   ```

   The BIG-IP system should now be sending log messages to your remote host.