Bivio 6310-NC Common Criteria Administrative Guidance

Version 1.10

Nov 23, 2020

CONTENTS

1. INTRODUCTION

This document describes the procedures for setting up the Bivio 6310-NC Platform series of products running BiviOS 8.5.1, based on RHEL 8.2, for NIAP compliance, following the Collaborative Protective Profile for Network Devices (NDcPP).

The term “Bivio 6310-NC” refers to the following individual products:

Table 1: Bivio 6310-NC Naming Convention	
Part Number	Processor
Options with C11	Dual Intel Xeon Gold 6148, 2.4 GHz w/ 27Mb Cache
Options with C13	Dual Intel Xeon Silver 4110, 2.1 GHz w/ 11Mb Cache
Options with C15	Dual Intel Xeon Gold 6138, 2.0 GHz w/27Mb Cache
Options with C21	Dual Intel Xeon Silver 4215, 2.5Ghz with 11Mb cache
Options with C22	Dual Intel Xeon Silver 4214, 2.5Ghz with 11Mb cache
Options with C23	Dual Intel Xeon Silver 4208, 2.1Ghz with 11Mb cache
Options with C24	Dual Intel Xeon Gold 5222, 3.8 Ghz with 16.5 Mb cache
Options with C25	Dual Intel Xeon Gold 6242, 2.8Ghz with 22Mb cache
Options with C26	Dual Intel Xeon Gold 6252, 2.5Ghz with 35.75Mb cache
Options with C04	Intel Xeon D 1541, 2.1Ghz with 12MB cache
Part Number	Installed RAM
Options with M1	256GB DDR4-2666 memory
Options with M2	512GB DDR4-2666 memory
Options with M3	384GB DDR4-2666 memory
Options with M4	768GB DDR4-2666 memory
Options with M5	1536GB DDR4-2666 memory
Part Number	Installed Storage
Options with D1	2x 1TB SSD storage
Options with D2	2x 2TB SSD storage
Options with D3	4x 2TB SSD storage
Options with D4	8x 2TB SSD storage
Options with D5	4x 3.8TB SSD storage
Options with D6	8x 3.8TB SSD storage
Part Number	Installed NIC Interfaces
Options with N1	2x 10GbE Fiber interfaces and 4x 1GbE Copper interfaces
Options with N2	4x 10GbE Fiber interfaces and 4x 1GbE Copper interfaces
Options with N3	6x 10GbE Fiber interfaces and 2x 1GbE Copper interfaces
Options with N4	4x 10GbE Fiber interfaces and 2x 1GbE Copper interfaces

The option C04 above refers to the PacStar 451 chassis. This chassis has no configuration options and will use only the C04 designator.

The product is described in the following documents:

[1] Bivio 6310-NC Platform Hardware Installation and Configuration Guide

[2] BiviOS™ User Guide for the Bivio 6310-NC Platform

[3] Bivio 6310-NC Security Target

The following additional references are provided for informational purposes. The Administrator may want to refer to them on occasion, but it is not necessary as the operational procedures and the required information are part of this document.

[4] Red Hat Enterprise Linux 8 Security Guide

[5] The stunnel TLS Proxy

[6] The OpenSSL CSR Creation Utility

This document describes how the Bivio 6310-NC must be configured in order to conform with a CC evaluated configuration. Only the configuration items relevant for compliance are discussed in this document. Some familiarity with Linux and associated security features is assumed. If a feature is fully configured, and no other compliant configurations are possible, that feature will be mentioned but configuration options may not be discussed.

Note: Unless the configuration of the Bivio 6310-NC device follows this document, the resulting configuration will not be one of the CC evaluated configurations.

The Bivio 6310-NC product consists of a single processing unit that includes both a management entity and and application entity. These guidelines apply to the management entity, which controls access to the processor and the application. The application is not accessible except via the management entity. The management entity may be accessed over the network, or via a local serial console.

2. Operational Environment – IT Requirements

Operation of the product requires the following hardware / infrastructure to be present:

- Syslog server conformant to RFCs 5424 (Syslog over TCP), capable of receiving an SSH tunnel from the Bivio 6310-NC.
- A local console with an RS-232 port for use with the Bivio provided console cable.

The following software is required for secure operation of the product:

- Administrators will need an SSH Client conformant to RFCs 4251, 4252, 4253, 4254, and 6668.
    - The SSH client must support AES128-CBC and AES256-CBC encryption algorithms, using HMAC-SHA2-256 or HMAC-SHA2-512 integrity algorithms, and performing key exchange using Diffie-Hellman Group14-SHA1.
    - To perform public key authentication to the TOE, the SSH client must support SSH-RSA.
- The product also provides a TLS protected server capability, which requires a TLS client capable of negotiating one of the following cipher suites:
    - TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
    - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
    - The TLS client must support TLS v1.2

The TLS server offers a Telnet login service over TLSv1.2. Thus, the best way to utilize this service will be to use a TLS enabled Telnet client. If that is not available, a regular Telnet client can be used via a TLS proxy.

3. Operational Environment – Procedural/Policy Requirements

- The product is placed in a physically secure environment. Only authorized personnel are allowed physical access to the product, including the local (serial) console.
- The product does not contain any general-purpose computing capabilities. For example, there are no compilers available in the product.
- It is assumed that the protection of the traffic that traverses the Bivio 6310-NC is / will be covered by other security and assurance measures.
- The Administrator of the Bivio 6310-NC product is trusted to follow and apply the guidance document faithfully.
- The product firmware and software are updated by the Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
- The Administrator's credentials (private key) used to access the Bivio 6310-NC must be protected on any other platform on which they reside.
- The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.
- The Bivio 6310-NC Guidance document (this document) will be made available to the customer via the Bivio support portal. A copy of this document will also be provided in PDF format on optical media, which will be included with the shipment of the product.

4. Installation and Initial Configuration

When the Bivio 6310-NC is first received, it must be installed and made accessible. The steps are as follows:

- Unpack the hardware and rack mount it
- Assign an IP address to the management port
- Set the default router
- Set the DNS server(s)
- Set the time zone and time
- Configure the NTP service
- Change the system name as desired
- Customize the system prompt

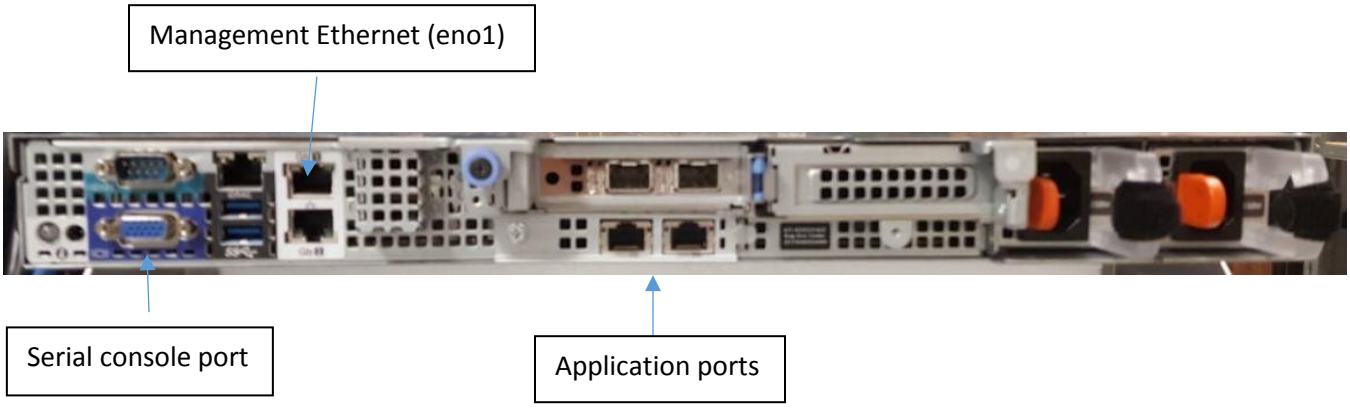
The "Bivio 6310-NC Platform Hardware Installation and Configuration Guide" [1] document is available from Bivio Support, and contains a description of the hardware, racking instructions, and initial configuration.

[1] has a diagram of the rear view of the system. The management Ethernet port and the local serial console port are indicated in this diagram. Other ports need not be accessed for the purposes of this document.

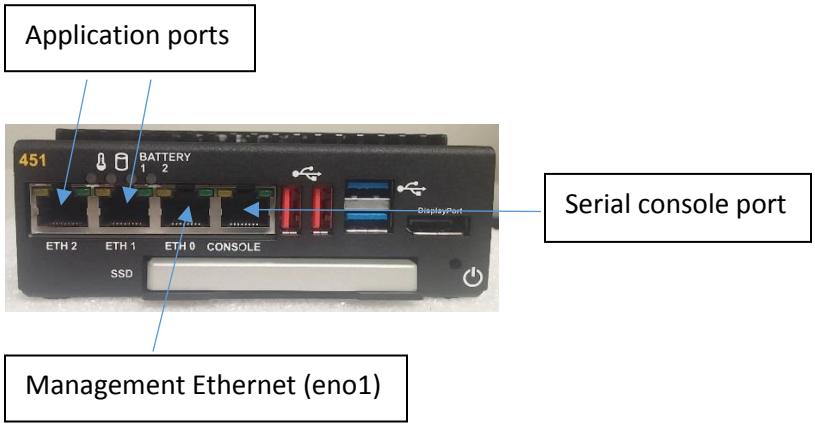
The customer should use the initial configuration instructions instructions provided in this section after physically installing the box in a rack or other suitable location, as described in the hardware installation document.

Initial configuration

There are two types of chassis, and they are both illustrated below. The first picture is representative of all models except the one with the C04 designator. The second picture below applies to the PacStar 451 chassis, which has the C04 model designator.



The above figure is a picture of the rear panel of the Bivio 6310-NC. The ports referenced in this document are indicated in the diagram.



The above figure is a picture of the front panel of the PacStar 451 chassis. The ports referenced in this document are indicated in the diagram. The power connecter is situated in the rear panel. Note that the external label for the Management Ethernet port appears as “ETH0”, but internal to the system, appears

with the same name as in the other models, namely, “eno1”. Thus, the configuration files for this interface have the same name in all models.

The following steps must be executed after the system has been rack mounted or installed as per the hardware guide.

1. Connect the power cables to the system.
2. Connect the provided serial console cable to the serial console port. The terminal settings for this connection are 115200 N/8/1.
3. Press the power button (in front of the system) to power up the system. Watch the system boot to ensure you have the correct terminal settings.
4. When the system has finished booting, the console should show the login prompt. Login as “root”, using the default password “root”.
5. Change the password using the command “passwd” and following the prompts. Logout after a suitable password has been set.
6. Login as “admin” using the default password “admin”. Change the password using the command “passwd”. The new password will need to be at least nine characters long, by default. Do not logout, as there are more steps to execute.
7. Edit the file `/etc/sysconfig/network-scripts/ifcfg-eno1` as follows:

```
sudo vi /etc/sysconfig/network-scripts/ifcfg-eno1
```

8. Set the management IP address, netmask, default gateway, DNS server, and DNS search domain. Save the file. The `ifcfg-eno1` file should look as follows (change the fields in bold as appropriate):

```
# Generated by dracut initrd
NAME=eno1
DEVICE=eno1
ONBOOT=yes
NETBOOT=no
UUID=3dcf20e8-bdb9-4416-9f59-a9641c3ea459
IPV6INIT=no
BOOTPROTO=none
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
IPADDR=192.168.120.165
PREFIX=24
GATEWAY=192.168.120.2
DNS1=192.168.2.1
DOMAIN=bivio.net
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
```

9. The application ports indicated on the rear panel are transparent and do not have IP addresses. These are visible only to the application. The unlabeled ports are not used for the default Bivio application.

10. Execute the following command to put the above changes into effect:

```
sudo service NetworkManager.service restart
```

At this point, an Ethernet cable may be connected from the management Ethernet port to the appropriate router or switch to make the box accessible to the Administrator via SSH. Do not logout since there are a few more things to do.

a. The host name (this includes the fully qualified domain name, or FQDN) may be set as follows (for example):

```
sudo hostnamectl set-hostname test.bivio.net
```

Upon the next login, the prompt will appear as follows:

```
admin@test ~$
```

Logging in and out of the system

When logging in locally via the serial console, or remotely via the SSH or TLS servers, the Administrator will be presented with a prompt for login name (“admin”) and then the password. The login name may be skipped if it has already been presented by the system, as is common for remote access methods such as SSH (and TLS in this case). Once the correct information has been presented, the Administrator will be presented with the login shell.

The Administrator may log out at any time by typing one of the commands “`logout`” or “`exit`”. This command can be used for both local and remote sessions over SSH or TLS.

Setting time

The local time on the system is accurate enough that no other time synchronization such as NTP needs to be enabled once time is set on the system. We recommend that the time be set at least once a year by the Administrator to ensure that it is accurate, if the NTP client is not used. If the Administrator does not want to use the NTP client for time synchronization, the NTP client (`chronyd`) may be disabled as follows:

```
sudo service chronyd stop
sudo chkconfig chronyd off
```

While logged in as “admin”, set the time zone and time as follows.

11. Use the `timedatectl` command to check the time zone and time.

```
timedatectl
```

12. If the output does not look right, get a list of time zones as follows, and ascertain the right time zone:

```
timedatectl list-timezones
```

13. If the correct time zone happens to be "America/Los_Angeles" (for example), set it as follows:

```
sudo timedatectl set-timezone America/Los_Angeles
```

14. Now you can logout of the system.

15. It is not necessary to reboot the box. All the above changes should be in effect, and the Administrator should be able to login to the system over SSH.

Enabling the NTP Client for Time Synchronization

By default, the NTP client should be disabled. If it is not desired to run the NTP client, follow the steps provided above to stop and disable the NTP client. If the NTP client is used, it must be used securely with symmetric key based authentication.

The NTP configuration is present in the file `/etc/chrony.conf`. The symmetric keys for authenticating the server are present in `/etc/chrony.keys`.

In order to configure the NTP servers, edit the `chrony.conf` file (as `root`) using the `vi` editor. Delete any existing `server` or `pool` directives in the file. To configure a set of NTP servers, add server directives at the top of the file as follows:

```
server 0.rhel.pool.ntp.org iburst key 1
server 1.rhel.pool.ntp.org iburst key 2
```

A `pool` directive may be used instead of the server directive(s) if server IP address are maintained as a pool and can change. The `pool` directive causes NTP to look up the server address dynamically. For example:

```
pool pool.ntp.org iburst key 5
```

The integer following the `key` keyword must be in the range 1 through (2^32)-1 and is the ID of the symmetric key that must be used with the server or pool. Keys are specified in the `/etc/chrony.keys` file as follows.

```
1 SHA1 HEX:1dc764e0791b11fa67efc7ecbc4b0d73f68a070c
2 SHA256 HEX: B2159C05D6A219673A3B7E896B6DE07F6A440995
```

The message digests allowed are SHA1, SHA256, and SHA512.

Note that the key and the ID must match configuration in the server to which the ID applies. The keys must be conveyed to the client in a secure manner, and the key file must be readable by root and the user `chrony` for the NTP client to function properly.

Keys can be generated using the chronyc command:

```
chronyc keygen <key id> <digest type> <key length>
```

If the key length is not specified, 160 bits is assumed. For example:

```
root@test ~$ chronyc keygen 5 SHA256
5 SHA256 HEX:0614C160D763746054F541917F2C207E084040E0
```

The line printed out (shown in bold) must be added to the `/etc/chrony.keys` file and securely conveyed to the server's configuration file.

After the NTP client is configured, it must be started for the configuration to take effect.

```
sudo chkconfig chronyd on
sudo service chronyd start
```

The command "chronyc tracking" can be used to verify that there is synchronization with the server. If synchronized, the "Reference ID" field will show the IP address of the server(s). If not synchronized, it will be set to zero.

```
root@test ~$ chronyc tracking
Reference ID    : C0A8787D (sw-odessa-1.bivio.net)
Stratum         : 4
Ref time (UTC)  : Mon Jun 29 19:57:49 2020
System time     : 0.000000029 seconds fast of NTP time
Last offset     : -0.000028382 seconds
RMS offset      : 0.000028382 seconds
Frequency       : 4.826 ppm slow
Residual freq   : -2.060 ppm
Skew            : 0.021 ppm
Root delay      : 0.084774576 seconds
Root dispersion : 0.002185735 seconds
Update interval : 2.0 seconds
Leap status     : Normal
```

The command "chronyc sources" can be used to see the time offsets from the NTP servers that are configured:

```
root@test ~$ chronyc sources
210 Number of sources = 4
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^* srcf-ntp.stanford.edu         2  10   377   504    +23us[  +91us] +/-   23ms
^- 13.86.101.172                 3  10   377   511  +4355us[+4423us] +/-   48ms
^- t2.time.bf1.yahoo.com         2  10   377   828  -1186us[-1126us] +/-   73ms
^- 50-205-244-111-static.hf>     2   9   377   289  +1198us[+1198us] +/-   69ms
```

Rebooting the system

The following steps describe the process for rebooting the system.

a) Login as "admin", either via SSH, or over the serial console, and type the command:

```
sudo -s shutdown -r now
```

b) The system should be back up within a few minutes. If there are problems, please contact Bivio support.

Power cycling the system

The system will need to be powered off sometimes, for example, when it needs to be moved or shipped. The following procedure describes how this can be done safely, and how to switch the power back on.

Powering off is done as follows:

a) Login as root into the system via the serial console and execute the command:

```
sudo -s shutdown -h now
```

b) Once the system is halted, it will be safe to power off the system by pulling out the power cables.

To power the system on again, wait at least 30 seconds after it is powered off, plug the power cables back in, and press the power switch on the front panel. The system should be up within a few minutes.

Once the initial configuration is done, the system will be accessible via the Ethernet management port and can be managed and operated remotely. The following sections detail the administrative procedures, and CC conformant configuration options where available.

Note: The local (serial) console is only accessible by authorized personnel. As such, most normal activities will be conducted remotely on the system. However, all administration procedures may be performed over the local serial console, by the Authorized Administrator, if necessary (i.e. an emergency recovery procedure). Administrative procedures are identical whether performed locally or remotely.

The login banner may be set separately for the local and remote consoles, as described later in this document (see section “Login warning banner”). By default, the login banner is the same for both local and remote consoles. The administrator may change the banners and put in specific markers or text so that the local console can be distinguished from a remote console while logging into the system.

If the administrator is already logged in, the `tty` command may be used to distinguish the local console from a remote console. On the local console, the `tty` command produces the following output:

```
admin@test ~$ tty
/dev/ttyS0
```

where `ttyS0` represents the local serial port. On remote consoles, the `tty` command produces output that has the following form:

```
admin@test ~$ tty
/dev/pts/2
```

where the pseudo terminal number (2 in the above example) may vary.

5. Configuration Items for CC Compliance

Once the initial configuration is complete, some additional items need to be configured by the Administrator for CC compliance. These are as follows.

1. Audit parameters (if necessary)
2. Password length (if necessary)
3. SSH host keys (if necessary)
4. The SSH tunnel for conveying audit logs to the remote syslog server
5. The TLS login server

6. Parameters for account locking if password retries exceed the maximum allowed number of retries

Cryptographic engine parameters are all preconfigured at the factory and should not be changed by the Administrator.

Note: The parameters described in this document should be set as per the guidance herein in order to maintain CC compliance.

6. THE SYSTEM ADMINISTRATOR

Once initial configuration is complete, the "root" user account need not be used any longer. In fact, root login is not permitted from remote locations. Root login is permitted over the local serial console, which is only physically accessible by authorized personnel.

It is the duty of the System Administrator (also known as the Authorized Administrator, Security Administrator, or just the Administrator) to manage the security of the Bivio 6310-NC as a product, and as a component of the larger operational environment. The sections that follow detail how the Administrator can accomplish this function.

All the operational tasks and day-to-day administration can be done by logging in as the Administrator, using the "admin" user account. To that end, the "root" user password should be stored securely, and never used for operational purposes. The Administrator ("admin") can effectively perform all administrative tasks without having to use the root password, by prefixing commands with the string "`sudo`". The Administrator will only be prompted for his/her own password for authentication. For example, in order search for an audit log related to login, the Administrator may run the following command:

```
sudo ausearch -m user_login,user_start -i
```

and enter his/her own password when prompted for it. The Administrator will not be prompted for a password for the next 5 minutes, by default. Some commands will require the "-s" parameter, and these are documented for specific commands in this document.

If the Administrator wants to stay in the "sudo" shell for longer than 5 minutes (to avoid typing in the password too often), the following command can be used:

```
sudo -s
```

Note that the Administrator is not assuming the identity of "root" by using the "sudo" command. In particular, the administrator will not be able to view the system's secret keys. The Administrator must exit the "sudo" shell (by typing "`exit`") as soon as the tasks are accomplished.

7. CONFIGURING LOGIN RETRIES AND ACCOUNT LOCKING

The Bivio 6310-NC requires only the single administrator `admin` for operation. The Administrator will perform almost all administrative activities by logging in over an SSH or TLS connection from a remote location. Thus, login activity from remote locations over SSH or TLS using password authentication is

governed by some security parameters that are configurable, and these are described in the rest of this section.

In the process of logging in over SSH or TLS, the Administrator may inadvertently enter an incorrect password. Upon authentication failure, the Administrator will retry by entering the password again. The maximum number of times the Administrator may retry logging in is a configurable parameter. If the number of attempts to enter the correct password exceeds this parameter, the Administrator's account will be locked for a configurable period.

There are two lines in the file `/etc/pam.d/password-auth` which contain the following string:

```
pam_faillock.so preauth silent fail_interval=900 unlock_time=600 deny=3
```

The maximum number of retries is determined by the `deny` parameter which is set to `3` by default. The `fail_interval` parameter, which is set to 900 seconds (15 minutes), means that if there are three unsuccessful login attempts with no intervening successful logins within this interval, the account will be locked.

Once the account is locked, the amount of time for which it will remain locked is determined by the unlock_time parameter, which by default is set to 600 seconds (10 minutes). The `unlock_time` parameter can also be set to the string `never`, which means that the account can only be unlocked by the Administrator by logging in over the local serial console.

Using the `vi` editor, the Administrator may edit the `/etc/pam.d/password-auth` file to change one or both of these parameters as desired. Both occurrences of each of these values must be changed to the respective new values.

```
sudo vi /etc/pam.d/password-auth
```

The new values will take effect as soon as the file is saved.

The Administrator may unlock the account by logging into the system via the local serial console and executing the following command:

```
sudo faillock --user admin --reset
```

Note that the Administrator account will never be locked when logging in over the local serial console.

8. COMMON TASKS

If you need to contact Bivio Support for anything, you will need to provide the software version. This is stored in the file `/etc/NRDIST`:

```
admin@test ~$ cat /etc/NRDIST
V: Version 8.5.1 (Build 202006181129)
```

Much finer detail regarding versions of individual software packages may be obtained with the following command:

```
admin@test ~$ rpm -qa
```

The above command is seldom required but could be useful in some cases.

A system name, which is different from the host name, and has only local significance, can be set as follows:

```
admin@test ~$ sudo bvspcfgset /bivio/system/id/sysname Eng
```

Upon logging out and logging back in, the prompt will appear as follows:

```
[Eng] admin@test ~$
```

The local time on the system can be set using the `timedatectl` command. For example:

```
sudo timedatectl set-time "2020-8-19 13:32:00"
```

This command will not allow you to set time if NTP is active (i.e. if the `chronyd` daemon is running).

Alternatively, the `date` command may be used to set time. The manual page for `date` provides all the information needed. For example:

```
admin@test ~$ sudo date -s "Tue Jun 30 10:35:13 PDT 2020"
```

9. Security Audit Data Generation

A key feature of CC compliance is the ability to audit all security relevant activities performed on the system. In this section, the configuration items relevant to auditing are described, and examples of each type of audit record are provided for the Administrator's reference.

A full description of the Linux auditing mechanism is provided in [4] (this is for informational purposes only and is not required reading).

Audit information is sent to two repositories: the local audit log (/var/log/audit/audit.log), and a remote (external) syslog server via an SSH tunnel. Only one external syslog server is supported.

The audit sub-system makes every attempt to create specific logs that are easy to search. However, sometimes it is difficult, or even impossible, to capture exactly what was changed. In order to address this, every administrator command typed into the login shell is also captured. Thus, albeit cumbersome, every action can be tracked in detail, if necessary, by examining the keyboard activity of the Administrator.

The subject identity is determined in a context sensitive manner and varies depending on the audit log record type. If the "`auid`" field is set to a non-root account name, this is the identity. If not, the identity is determined by an "`acct=`" field in the record, or an "`id=`" phrase in the message body.

Sometimes, the "`auid`" field appears with the value "`unset`". This indicates that the audit log was created by a system process that does not have a user login context (such as a system daemon or the kernel).

The relevant configuration options for the audit system are described below.

Storage space for audit data

The disk space available for audit logs is basically all the storage space available on the audit log partition, which is bounded by 50GB. However, the maximum size of a log file is restricted by the `max_log_file` parameter (see below, default is 6MB) and the number of log files stored is restricted by the `num_logs` parameter (see below, default is 5). Thus the total amount of space used for audit logs will not exceed `max_log_file*num_logs` (default is 30MB). However, the disk could fill up for other reasons, and not leave enough room for audit logs. Audit logging will be adversely affected if the total amount of disk space available falls below 30MB. The `space_left` parameter is used to configure a threshold for generating a warning to the administrator. This is set to a value sufficiently above the space required for audit logs (see below, default is 75 MB) so the administrator has enough time to react before the space left falls below required space for audit logs.

When the space required for audit data storage is starting too low (below a specified threshold), a warning is issued to the System Administrator as an audit record. The "`space_left`" parameter in `/etc/audit/auditd.conf` specifies the threshold, and is set by default to 75 MB:

```
space_left = 75
```

Use the `vi` editor to edit the file `/etc/audit/auditd.conf`, and change this parameter to the desired value:

```
sudo vi /etc/audit/auditd.conf
```

Save the file and quit `vi` when done with editing.

The default values shown above for `max_log_file`, `num_logs`, and `space_left` are the minimally acceptable values and should suffice in most cases. If they need to be changed, it is important to remember that the maximum disk space used for audit logs will be equal to the product of `max_log_file` and `num_logs`. The administrator should ensure that the maximum space used is well within the total storage space available to the processor as described above.

Audit log rotation

By default, audit logs are rotated when the size of `/var/log/audit/audit.log` reaches 6 MB, and up to 5 rotated files are kept.

The "`max_log_file`" parameter in `/etc/audit/auditd.conf` specifies the maximum size allowed for audit log files, and then it is rotated by adding a suffix in the range [0 – 99]. By default, this is set to 6 (MB), and can be changed by the Administrator.

The "`max_log_file_action`" parameter determines what the system will do when an audit log file reaches the maximum size. This is set to "`rotate`" and must not be changed.

The "`num_logs`" parameter determines the number of rotated files that are kept; older files are deleted. By default, this is set to 5 (files) and can be changed by the Administrator.

```
max_log_file = 6
num_logs = 5
```

Use the `vi` editor to edit the file `/etc/audit/auditd.conf`, and change these parameters as desired:

```
sudo vi /etc/audit/auditd.conf
```

Save the file and quit `vi` when done with editing.

For the changes to take effect, the audit daemon should be restarted as follows:

```
sudo service auditd restart
```

Note: CC Evaluation Activities only cover the audit parameters described above. Other parameters are not covered in this CC Evaluation.

Audit storage space running low – warning

When the disk space left in the audit log partition is lower than the value of the "`space_left`" parameter in `/etc/audit/auditd.conf`, an audit record is created as a warning to the Administrator. However, when this condition occurs, the audit log does not make it into `/var/log/audit/audit.log`. Instead, it is sent to syslog (`/var/log/messages`). Once this condition is reached (space left in audit log partition falls below `space_left` parameter) audit logs will be sent to syslog, and forwarded on to the remote audit log server, but not sent to the local audit log, since it is nearly full.

Once space is restored in the audit log partition, the audit server will need to be restarted with the following command for audit logs to be sent to the local audit log repository again:

```
sudo service auditd restart
```

The audit record created in `/var/log/messages` for the warning message looks as follows, and is also forwarded to the remote audit log server:

```
Jun 17 09:51:27 test audisp-syslog[1460]: node=test.bivio.net type=USER
msg=audit(1592412687.343:4524): pid=4344 uid=0 auid=4294967295
ses=4294967295 subj=system_u:system_r:auditd_t:s0 msg='text=Warning:
audit log storage space is running low exe="/etc/audit/auditctlbin"
hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
```

Audit log entry format

The different audit record types and fields are described in detail in [4]. For convenience, the event fields and audit record types that are significant for the purposes of this document are listed below.

Event Field	Explanation
`acct`	Records a user's account name.
`addr`	Records the IPv4 or IPv6 address. This field usually follows a `hostname` field and contains the address the host name resolves to.
`auid`	Records the Audit user ID. This ID is assigned to a user upon login and is inherited by every process even when the user's identity changes (for example, by switching user accounts with "`su - john`").
`cmd`	Records the entire command line that is executed.
`comm`	Records the command that is executed.

cwd	Records the path to the directory in which a system call was invoked.
data	Records data associated with TTY records.
egid	Records the effective group ID of the user who started the analyzed process.
euid	Records the effective user ID of the user who started the analyzed process.
exe	Records the path to the executable that was used to invoke the analyzed process.
exit	Records the exit code returned by a system call.
family	Records the type of address protocol that was used, either IPv4 or IPv6.
gid	Records the group ID.
hostname	Records the host name.
key	Records the user defined string associated with a rule that generated a particular event in the Audit log.
msg	Records a time stamp and a unique ID of a record, or various event specific *<name>=<value>* pairs provided by the kernel or user space
pid	Records the Process ID (PID).
ppid	Records the Parent Process ID (PID).
res	Records the result of the operation that triggered the Audit event.
result	Records the result of the operation that triggered the Audit event.
success	Records whether a system call was successful or failed.
suid	Records the set user ID of the user who started the analyzed process.
syscall	Records the type of the system call that was sent to the kernel.
terminal	Records the terminal name (without /dev/).
tty	Records the name of the controlling terminal. The value (none) is used if the process has no controlling terminal.
uid	Records the real user ID of the user who started the analyzed process.

Audit Record Type	Explanation
ANOM_RBAC_INTEGRITY_FAIL	Triggered when a Role-Based Access Control (RBAC) file integrity test failure is detected.
CONFIG_CHANGE	Triggered when the Audit system configuration is modified.
CRYPTO_KEY_USER	Triggered to record the cryptographic key identifier used for cryptographic purposes.
CRYPTO_SESSION	Triggered to record parameters set during a TLS session establishment.
DAEMON_END	Triggered when a daemon is successfully stopped.
DAEMON_START	Triggered when the auditd daemon is started.
SERVICE_START	Triggered when a service is started.
SERVICE_STOP	Triggered when a service is stopped.
SYSCALL	Triggered to record a system call to the kernel.
TTY	Triggered when TTY input was sent to an administrative process.
USER	Triggered in application specific format.
USER_ACCT	Triggered when a user-space user account is modified.
USER_AUTH	Triggered when a user-space authentication attempt is detected.

USER_CHAUTHTOK	Triggered when a user account attribute is modified.
USER_END	Triggered when a user-space session is terminated.
USER_ERR	Triggered when a user account state error is detected.
USER_LOGIN	Triggered when a user logs in.
USER_LOGOUT	Triggered when a user logs out.
USER_START	Triggered when a user-space session is started.
USER_TTY	Triggered when an explanatory message about TTY input to an administrative process is sent from user-space by the root user
USER_CMD	Triggered when the Administrator types in any command into the login shell using the “sudo” command

Searching audit logs

In the remainder of this section, examples of different types of audit records are provided. The two utilities commonly used for searching audit records are `ausearch` and `aureport`. These are described in the Linux man pages on the system. Manual pages may be viewed by typing the following commands:

```
man ausearch

man aureport
```

For example, to search of all instances of a service being started, the following command can be used:

```
sudo ausearch -m service_start
```

The `grep` utility can be used to narrow search so that the record contains a specific string of interest. For example,

```
sudo ausearch -m service_start | grep auditd
```

will yield audit records for all events where the `auditd` daemon was started. For searching console input for execution of a specific command or changes to a file (using vi for example), it is recommended that the “-i” parameter is used with `ausearch`, for example:

```
sudo ausearch -m user_cmd -i
```

This will make the characters typed in more readable (otherwise numeric format will be used for certain text fields in the records). This also makes the timestamp attached to the `msg` event field human readable.

Once some records are listed as a result of a search, the event ID of a record can be used to view all the audit logs associated with that event. For example, if the search:

```
sudo ausearch -m user -i | grep PASSED
```

yields:

```
node=test.bivio.net type=USER msg=audit(01/12/2018 15:31:47.110:195) :
pid=2985 uid=root auid=unset ses=unset
```

```
subj=system_u:system_r:auditctl_t:s0 msg='SSL: All tests PASSED
exe=/usr/sbin/auditctl hostname=? addr=? terminal=? res=success'
```

then the event ID associated with it, namely 195, can be used to search for all the audit logs that were created as part of this event, as follows:

```
sudo ausearch -a 195 -i
```

The following is a listing of examples of all the relevant audit logs generated by the system. Note that these examples only illustrate the form of the log. Exact content will vary depending on the individual instantiations.

Starting the audit daemon

Records created:

```
node=test.bivio.net type=PROCTITLE msg=audit(06/15/2020
07:56:39.524:17125) : proctitle=sudo service auditd start

node=test.bivio.net type=SYSCALL msg=audit(06/15/2020
07:56:39.524:17125) : arch=x86_64 syscall=openat success=yes exit=3
a0=0xffffff9c a1=0x56264ead53e0 a2=O_RDONLY a3=0x0 items=1 ppid=4276
pid=4280 auid=admin uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts0 ses=6 comm=service
exe=/usr/bin/bash subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
key=successful-access

node=test.bivio.net type=SERVICE_START msg=audit(06/15/2020
07:56:39.793:17555) : pid=1 uid=root auid=unset ses=unset
subj=system_u:system_r:init_t:s0 msg='unit=auditd comm=systemd
exe=/usr/lib/systemd/systemd hostname=? addr=? terminal=? res=success'
```

Notes:

Notice that the first log above shows the command that was executed to start the audit daemon. The second log, which is related to the first because they have the same event ID (17125), shows that the audited user ID (auid) is `admin` and the real user ID is root. The third log shows that the audit daemon was started successfully. In the third log, the auid field is unset, which means that this log was created by a system daemon running with uid equal to root, as indicated in the log.

Stopping the audit daemon

Records created:

```
node=test.bivio.net type=PROCTITLE msg=audit(06/15/2020
13:11:49.342:27830) : proctitle=sudo service auditd stop

node=test.bivio.net type=SYSCALL msg=audit(06/15/2020
13:11:49.342:27830) : arch=x86_64 syscall=openat success=yes exit=3
a0=0xffffff9c a1=0x55923e69f3e0 a2=O_RDONLY a3=0x0 items=1 ppid=5204
pid=5207 auid=admin uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts0 ses=18 comm=service
exe=/usr/bin/bash subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
key=successful-access
```

```
node=test.bivio.net type=SERVICE_STOP msg=audit(06/15/2020
13:06:41.335:25300) : pid=1 uid=root auid=unset ses=unset
subj=system_u:system_r:init_t:s0 msg='unit=auditd comm=systemd
exe=/usr/lib/systemd/systemd hostname=? addr=? terminal=? res=success'
```

Notes:

Using the same reasoning as for the records created when starting the audit daemon we can infer that the audit daemon was stopped by the administrator. The third record above may not always be created, due to a race condition (the audit daemon sometimes stops before this record can be written to the audit log). In this case the following command may be used to ensure that the audit daemon has actually exited:

```
ps -ef | grep auditd
```

When the audit daemon is running the output will contain a line such as the following:

```
root        5356       1  5 13:30 ?        00:00:00 /sbin/auditd
```

Admin Login / Initiation of trusted path

Records created for remote login:

```
node=test.bivio.net type=USER_AUTH msg=audit(06/15/2020
13:46:30.713:32363) : pid=5467 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=pam_faillock,pam_unix acct=admin exe=/usr/sbin/sshd
hostname=192.168.2.10 addr=192.168.2.10 terminal=ssh res=success'

node=test.bivio.net type=USER_START msg=audit(06/15/2020
13:46:30.801:32422) : pid=5467 uid=root auid=admin ses=24
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:session_open
grantors=pam_selinux,pam_loginuid,pam_selinux,pam_namespace,pam_keyinit
,pam_keyinit,pam_limits,pam_unix,pam_tty_audit,pam_umask,pam_lastlog
acct=admin exe=/usr/sbin/sshd hostname=192.168.2.10 addr=192.168.2.10
terminal=ssh res=success'

node=test.bivio.net type=USER_LOGIN msg=audit(06/15/2020
13:46:30.858:32474) : pid=5467 uid=root auid=admin ses=24
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.2.10 terminal=/dev/pts/0
res=success'

node=test.bivio.net type=USER_START msg=audit(06/15/2020
13:46:30.858:32475) : pid=5467 uid=root auid=admin ses=24
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.2.10 terminal=/dev/pts/0
res=success'
```

Records created for local login (over serial console):

```
node=test.bivio.net type=USER_AUTH msg=audit(06/15/2020
16:45:00.612:32942) : pid=2528 uid=root auid=unset ses=unset
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023
msg='op=PAM:authentication grantors=pam_unix acct=admin
```

```
exe=/usr/bin/login hostname=test.bivio.net addr=? terminal=ttyS0
res=success'

node=test.bivio.net type=USER_START msg=audit(06/15/2020
16:45:00.688:32988) : pid=2528 uid=root auid=admin ses=28
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023
msg='op=PAM:session_open
grantors=pam_selinux,pam_loginuid,pam_console,pam_selinux,pam_namespace
,pam_keyinit,pam_keyinit,pam_limits,pam_unix,pam_tty_audit,pam_umask,pa
m_lastlog acct=admin exe=/usr/bin/login hostname=test.bivio.net addr=?
terminal=ttyS0 res=success'

node=test.bivio.net type=USER_LOGIN msg=audit(06/15/2020
16:45:00.689:32996) : pid=2528 uid=root auid=admin ses=28
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023 msg='op=login
id=admin exe=/usr/bin/login hostname=test.bivio.net addr=?
terminal=ttyS0 res=success'
```

Notes:

The above shows that the Administrator logged in successfully, and records the name and address of the host from which the Administrator logged in. When the audit records are created, the process that gets audited is running as `root` while attempting to authenticate the user. Thus, `uid` is set to `root` while `auid` is unset or set to `admin`.

Records created for failed attempt to login (over local console):

```
node=test.bivio.net type=USER_AUTH msg=audit(06/15/2020
16:50:43.526:33377) : pid=5823 uid=root auid=unset ses=unset
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023
msg='op=PAM:authentication grantors=? acct=admin exe=/usr/bin/login
hostname=test.bivio.net addr=? terminal=ttyS0 res=failed'

node=test.bivio.net type=USER_LOGIN msg=audit(06/15/2020
16:50:45.563:33378) : pid=5823 uid=root auid=unset ses=unset
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023 msg='op=login
id=admin exe=/usr/bin/login hostname=test.bivio.net addr=?
terminal=ttyS0 res=failed'
```

Records created for successful login using public key authentication:

```
node=test.bivio.net type=USER_AUTH msg=audit(06/15/2020
23:00:04.960:48583) : pid=7898 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=pubkey_auth
grantors=auth-key acct=admin exe=/usr/sbin/sshd hostname=?
addr=192.168.120.164 terminal=? res=success'

node=test.bivio.net type=USER_START msg=audit(06/15/2020
23:00:05.002:48629) : pid=7898 uid=root auid=admin ses=58
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:session_open
grantors=pam_selinux,pam_loginuid,pam_selinux,pam_namespace,pam_keyinit
,pam_keyinit,pam_limits,pam_systemd,pam_unix,pam_tty_audit,pam_umask,pa
m_lastlog acct=admin exe=/usr/sbin/sshd hostname=192.168.120.164
addr=192.168.120.164 terminal=ssh res=success'
```

```
node=test.bivio.net type=USER_LOGIN msg=audit(06/15/2020
23:00:05.057:48675) : pid=7898 uid=root auid=admin ses=58
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.120.164 terminal=/dev/pts/2
res=success'

node=test.bivio.net type=USER_START msg=audit(06/15/2020
23:00:05.057:48676) : pid=7898 uid=root auid=admin ses=58
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.120.164 terminal=/dev/pts/2
res=success'
```

Records created for failed login using public key authentication:

```
node=test.bivio.net type=USER_AUTH msg=audit(06/15/2020
23:08:16.314:49854) : pid=8008 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=pubkey acct=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.120.164 terminal=ssh
res=failed'

node=test.bivio.net type=USER_ERR msg=audit(06/15/2020
23:08:16.321:49856) : pid=8008 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:bad_ident
grantors=? acct=? exe=/usr/sbin/sshd hostname=192.168.120.164
addr=192.168.120.164 terminal=ssh res=failed'

node=test.bivio.net type=USER_LOGIN msg=audit(06/15/2020
23:08:16.322:49858) : pid=8008 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login acct=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.120.164 terminal=ssh
res=failed'
```

Admin logout / User initiated termination / Termination of trusted path

Records created for remote user:

```
node=test.bivio.net type=USER_END msg=audit(06/15/2020
23:15:00.803:50520) : pid=8025 uid=root auid=admin ses=59
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=? terminal=/dev/pts/2 res=success'

node=test.bivio.net type=USER_LOGOUT msg=audit(06/15/2020
23:15:00.804:50521) : pid=8025 uid=root auid=admin ses=59
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=? terminal=/dev/pts/2 res=success'

node=test.bivio.net type=USER_END msg=audit(06/15/2020
23:15:00.809:50528) : pid=8025 uid=root auid=admin ses=59
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:session_close
grantors=pam_selinux,pam_loginuid,pam_selinux,pam_namespace,pam_keyinit
,pam_keyinit,pam_limits,pam_systemd,pam_unix,pam_tty_audit,pam_umask,pa
m_lastlog acct=admin exe=/usr/sbin/sshd hostname=192.168.120.164
addr=192.168.120.164 terminal=ssh res=success'
```

Records created for local user (over serial console):

```
node=test.bivio.net type=USER_END msg=audit(06/15/2020
23:17:43.489:50960) : pid=8067 uid=root auid=admin ses=60
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023
msg='op=PAM:session_close
grantors=pam_selinux,pam_loginuid,pam_console,pam_selinux,pam_namespace
,pam_keyinit,pam_keyinit,pam_limits,pam_systemd,pam_unix,pam_tty_audit,
pam_umask,pam_lastlog acct=admin exe=/usr/bin/login
hostname=test.bivio.net addr=? terminal=ttyS0 res=success'
```

Notes:

The above shows a successful logout by the Administrator.

Security related information

Security related information resides in various files, for example, `/etc/ssh/sshd_config`. If this file is modified, for example, audit logs are generated as shown below. Similar records are created for other files when modified using the `vi` editor. Only the relevant records are shown.

Records created:

```
node=test.bivio.net type=PROCTITLE msg=audit(06/16/2020
08:29:24.378:55103) : proctitle=sudo vi sshd_config

node=test.bivio.net type=PATH msg=audit(06/16/2020 08:29:24.378:55103)
: item=1 name=sshd_config inode=543974 dev=fd:00 mode=file,600
ouid=root ogid=root rdev=00:00 obj=sysadm_u:object_r:etc_t:s0
nametype=CREATE cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
cap_frootid=0

node=test.bivio.net type=SYSCALL msg=audit(06/16/2020
08:29:24.378:55103) : arch=x86_64 syscall=openat success=yes exit=5
a0=0xffffff9c a1=0x563b42164b20 a2=O_WRONLY|O_CREAT a3=0x180 items=2
ppid=9211 pid=9214 auid=admin uid=root gid=root euid=root suid=root
fsuid=root egid=root sgid=root fsgid=root tty=pts0 ses=66 comm=vi
exe=/usr/bin/vi subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
key=successful-create
```

Resetting a user's password

Records created:

```
node=test.bivio.net type=USER_CHAUTHTOK msg=audit(06/16/2020
09:05:44.376:56736) : pid=9342 uid=admin auid=admin ses=66
subj=sysadm_u:sysadm_r:passwd_t:s0-s0:c0.c1023 msg='op=PAM:chauthtok
grantors=pam_pwquality,pam_unix acct=admin exe=/usr/bin/passwd
hostname=test.bivio.net addr=? terminal=pts/0 res=success'
```

Notes:

The above shows that the `admin` account's password was successfully changed by the Administrator.

Starting a service

Records created:

```
node=test.bivio.net type=PROCTITLE msg=audit(06/16/2020
10:06:38.080:64438) : proctitle=sudo service chronyd start

node=test.bivio.net type=SYSCALL msg=audit(06/16/2020
10:06:38.080:64438) : arch=x86_64 syscall=openat success=yes exit=3
a0=0xffffff9c a1=0x7fe9dfdc9f57 a2=O_RDONLY|O_CLOEXEC a3=0x0 items=1
ppid=9865 pid=9868 auid=admin uid=root gid=root euid=root suid=root
fsuid=root egid=root sgid=root fsgid=root tty=pts0 ses=66
comm=systemctl exe=/usr/bin/systemctl
subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023 key=successful-access

node=test.bivio.net type=SERVICE_START msg=audit(06/16/2020
10:06:38.189:64488) : pid=1 uid=root auid=unset ses=unset
subj=system_u:system_r:init_t:s0 msg='unit=chronyd comm=systemd
exe=/usr/lib/systemd/systemd hostname=? addr=? terminal=? res=success'
```

Notes:

The first record shows the NTP service (`chronyd`) being started. The second record, related to the first via the same event ID, shows that the audited user is `admin`. The third record shows that `chronyd` was started by `systemd`, which is a system program running as root, so the `auid` field is unset.

Stopping a service

Records created:

```
node=test.bivio.net type=PROCTITLE msg=audit(06/16/2020
10:38:07.636:70660) : proctitle=sudo service chronyd stop

node=test.bivio.net type=SYSCALL msg=audit(06/16/2020
10:38:07.636:70660) : arch=x86_64 syscall=openat success=yes exit=3
a0=0xffffff9c a1=0x5589822a83e0 a2=O_RDONLY a3=0x0 items=1 ppid=10101
pid=10105 auid=admin uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts0 ses=70 comm=service
exe=/usr/bin/bash subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
key=successful-access

node=test.bivio.net type=SERVICE_STOP msg=audit(06/16/2020
10:38:07.770:70885) : pid=1 uid=root auid=unset ses=unset
subj=system_u:system_r:init_t:s0 msg='unit=chronyd comm=systemd
exe=/usr/lib/systemd/systemd hostname=? addr=? terminal=? res=success'
```

The first record shows the NTP service (`chronyd`) being stopped. The second record, related to the first via the same event ID, shows that the audited user is `admin`. The third record shows that `chronyd` was stopped by `systemd`, which is a system program running as root, so the `auid` field is unset.

Capture all Administrator commands

This is done in order to ensure that nothing is missed on the system. Whenever a direct method to search for a specific event is available (such as a user logging in), that should be used. This is a catch-all method that is used as a last resort when direct methods are not available. The following log is created when the **date** command is used by the administrator to set time.

Records created:

```
node=test.bivio.net type=USER_CMD msg=audit(06/16/2020
14:23:30.826:3114) : pid=2625 uid=admin auid=admin ses=1
subj=sysadm_u:sysadm_r:sysadm_sudo_t:s0-s0:c0.c1023
msg='cwd=/home/admin cmd=date -s Tue Jun 16 14:23:30 PDT 2020
exe=/usr/bin/sudo terminal=pts/0 res=success'
```

Sometimes, the administrator assumes the root identity by using one of the following commands: `su`, `sudo -s`, or `sudo -i`. In this situation, the following records will be created (among others).

```
node=test.bivio.net type=PROCTITLE msg=audit(06/25/2020
14:17:49.123:16274) : proctitle=date -s Thu Jun 25 14:17:49 PDT
2020
```

```
node=test.bivio.net type=EXECVE msg=audit(06/25/2020
14:17:49.123:16274) : argc=3 a0=date a1=-s a2=Thu Jun 25 14:17:49
PDT 2020
```

```
node=test.bivio.net type=SYSCALL msg=audit(06/25/2020
14:17:49.123:16274) : arch=x86_64 syscall=execve success=yes
exit=0 a0=0x555555a28c20 a1=0x555555a2a700 a2=0x5555559cf2d0
a3=0x8 items=2 ppid=139290 pid=140057 auid=admin uid=root
gid=root euid=root suid=root fsuid=root egid=root sgid=root
fsgid=root tty=pts1 ses=41 comm=date exe=/usr/bin/date
subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023 key=(null)
```

Finally, the audit log can be examined for all the characters typed in by the administrator using the command "`sudo aureport --tty`". The output in this case does not resemble the usual audit log, and can be very hard to examine:

```
TTY Report
===============================================
# date time event auid term sess comm data
===============================================
1. 06/28/2020 14:42:57 60279 1000 pts1 82 .vi ":q",<ret>
2. 06/28/2020 15:12:16 73978 0 pts0 84 yum "y",<nl>
3. 06/28/2020 15:13:44 73985 0 pts0 84 .vi ":q",<ret>
4. 06/28/2020 15:19:03 74511 1000 pts3 86 bash "exit",<ret>
5. 06/28/2020 15:19:15 75016 1000 pts3 87 bash <^D>
6. 06/28/2020 15:21:26 75551 1000 pts3 88 bash "exit",<ret>
```

The full buffer that contains these characters is not output in real time. Once the admin logs out, the whole buffer will be output.

Failure to establish an SSH session (client)

The only SSH client running on the system is the SSH tunnel client which is set up to transport audit logs to a remote audit server. SSH clients normally do not have permission to create audit records, since they do not run as root as a rule. However, the client is forced by the system to output debug information when the SSH tunnel is initiated. These logs are added to the local audit logs. The message type created is "USER".

Records created (only relevant ones are shown):

```
node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.127:5789) :
pid=5392 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Starting
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.168:5734) :
pid=5341 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: OpenSSH_8.0p1, OpenSSL
1.1.1c FIPS  28 May 2019 exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.174:5738) :
pid=5345 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Connecting to
192.168.120.125 [192.168.120.125] port 22. exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.176:5739) :
pid=5346 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Connection established.
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.302:5773) :
pid=5380 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Next authentication
method: publickey exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.311:5776) :
pid=5383 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: No more authentication
methods to try. exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.314:5777) :
pid=5384 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: root@192.168.120.125:
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:13:23.679:5810) :
pid=5416 uid=root auid=root ses=9
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

```
msg='text=ssh-tunnel-client 192.168.120.125: Exited
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'
```

Note:

Although there is no explicit “Reason for failure” mentioned above, the logs clearly indicate failure because public key authentication failed, and there are no other authentication methods to try. The “`res=success`” only indicates that the audit log was created successfully for the SSH client. The actual reason is embedded in the `msg` field in this case.

The initiator is identified as root, which is the user id used by SSH tunnel initiator, and the target IP address is also identified, in the above logs.

Successful SSH rekey (client)

Records created (only relevant records are shown):

```
node=test.bivio.net type=USER msg=audit(06/17/2020 11:58:37.319:5953) :
pid=5607 uid=root auid=root ses=14
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: rekeying in progress
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:58:37.353:5965) :
pid=5619 uid=root auid=root ses=14
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: ssh_set_newkeys: rekeying
out, input 5168 bytes 191 blocks, output 30760 bytes 1888 blocks
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:58:37.358:5968) :
pid=5622 uid=root auid=root ses=14
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: SSH2_MSG_NEWKEYS sent
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:58:37.361:5969) :
pid=5623 uid=root auid=root ses=14
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: expecting SSH2_MSG_NEWKEYS
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 11:58:37.365:5970) :
pid=5624 uid=root auid=root ses=14
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: SSH2_MSG_NEWKEYS received
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'
```

Notes:

This indicates a successful rekey attempt with host 192.168.120.125. Note that the SSH client logs have all the relevant information embedded in the message body.

Failure to establish an SSH session (server)

Many audit records are created when user authentication fails. One possible scenario showing a failure due to an invalid password for a login attempt by the Administrator, is shown here.

Records created (only relevant ones shown):

```
node=test.bivio.net type=USER_LOGIN msg=audit(06/17/2020
12:53:16.669:6254) : pid=5961 uid=root auid=root ses=16
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=root
exe=/usr/sbin/sshd hostname=? addr=192.168.2.10 terminal=/dev/pts/1
res=success'

node=test.bivio.net type=USER_START msg=audit(06/17/2020
12:53:16.669:6255) : pid=5961 uid=root auid=root ses=16
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=root
exe=/usr/sbin/sshd hostname=? addr=192.168.2.10 terminal=/dev/pts/1
res=success'

node=test.bivio.net type=USER_AUTH msg=audit(06/17/2020
12:54:47.685:6263) : pid=5991 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=192.168.2.10
addr=192.168.2.10 terminal=ssh res=failed'

node=test.bivio.net type=USER_AUTH msg=audit(06/17/2020
12:54:53.397:6264) : pid=5991 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=192.168.2.10
addr=192.168.2.10 terminal=ssh res=failed'

node=test.bivio.net type=USER_AUTH msg=audit(06/17/2020
12:54:59.373:6267) : pid=5991 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=192.168.2.10
addr=192.168.2.10 terminal=ssh res=failed'

node=test.bivio.net type=USER_LOGIN msg=audit(06/17/2020
12:55:01.981:6270) : pid=5991 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login acct=admin
exe=/usr/sbin/sshd hostname=? addr=192.168.2.10 terminal=ssh
res=failed'
```

Notes:

The above shows a failed attempt from a remote host to login as `admin` due to invalid password being entered incorrectly three times. The message body shows that password authentication was used. Because the number of attempts exceeded the maximum allowed (three), the account was locked.

Locking of user account

The following records are created when the `admin` account is locked after the number of unsuccessful login attempts exceeds the configured threshold.

```
node=test.bivio.net type=RESP_ACCT_LOCK msg=audit(06/17/2020
12:54:59.373:6266) : pid=5991 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='pam_faillock
uid=root  exe=/usr/sbin/sshd hostname=? addr=? terminal=?
res=success'
```

Although the above record does not contain a user ID, we can infer the details from the logs surrounding this log, which are basically the logs shown above for the failed authentication attempt.

Unlocking of user account

The following records are created when the `admin` account is automatically unlocked upon the expiry of a timeout.

```
node=test.bivio.net type=RESP_ACCT_UNLOCK_TIMED
msg=audit(1592429220.585:7646): pid=6513 uid=0 auid=4294967295
ses=4294967295 subj=system_u:system_r:sshd_t:s0-s0:c0.c1023
msg='pam_faillock uid=1000  exe="/usr/sbin/sshd"
hostname=192.168.2.10 addr=192.168.2.10 terminal=ssh
res=success'^]UID="root" AUID="unset" UID="admin"
```

The following records are created when the `admin` account is unlocked by the Administrator by logging in via the local serial console.

```
node=test.bivio.net type=USER_MGMT
msg=audit(1592428095.633:7604): pid=6457 uid=0 auid=0 ses=5
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='op=faillock-reset id=1000 exe="/usr/sbin/faillock"
hostname=test.bivio.net addr=? terminal=ttyS0 res=success'
UID="root" AUID="root" ID="admin"
```

Successful SSH rekey (server)

The following records are created when a rekey event occurs. The way to detect that this is a rekey event is to check if this occurs after a session is created, but before it ends.

Records created:

```
node=test.bivio.net type=CRYPTO_SESSION msg=audit(06/17/2020
15:30:51.416:10336) : pid=6998 uid=root auid=admin ses=36
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=start
direction=from-server cipher=aes128-cbc ksize=128 mac=hmac-sha2-256
pfs=diffie-hellman-group14-sha1 spid=7011 suid=admin rport=43326
laddr=192.168.120.165 lport=22  exe=/usr/sbin/sshd hostname=?
addr=192.168.2.10 terminal=? res=success'

node=test.bivio.net type=CRYPTO_SESSION msg=audit(06/17/2020
15:30:51.417:10337) : pid=6998 uid=root auid=admin ses=36
```

```
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=start
direction=from-client cipher=aes128-cbc ksize=128 mac=hmac-sha2-256
pfs=diffie-hellman-group14-sha1 spid=7011 suid=admin rport=43326
laddr=192.168.120.165 lport=22  exe=/usr/sbin/sshd hostname=?
addr=192.168.2.10 terminal=? res=success'

node=test.bivio.net type=CRYPTO_KEY_USER msg=audit(06/17/2020
15:30:51.427:10338) : pid=6998 uid=root auid=admin ses=36
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=destroy
kind=session fp=? direction=from-client spid=7011 suid=admin
rport=43326 laddr=192.168.120.165 lport=22  exe=/usr/sbin/sshd
hostname=? addr=192.168.2.10 terminal=? res=success'

node=test.bivio.net type=CRYPTO_KEY_USER msg=audit(06/17/2020
15:30:51.431:10339) : pid=6998 uid=root auid=admin ses=36
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=destroy
kind=session fp=? direction=from-server spid=7011 suid=admin
rport=43326 laddr=192.168.120.165 lport=22  exe=/usr/sbin/sshd
hostname=? addr=192.168.2.10 terminal=? res=success'
```

Note that a time based rekey event may not occur exactly when the time interval expires. The system is designed such that the rekey event will be triggered by any traffic after the rekey interval has expired.

Successful authentication for a login session over TLS

The bolded lines in the first two audit records shown below indicate the IP address of the remote host from where the login attempt is made. After the TLS connection is made successfully, the TLS server launches a local SSH session with the user's (in this case, the `admin` user) credentials. Thus, the audit records following the first two records below resemble a local SSH login with password authentication. Note that only the relevant records are shown below, for both TLS connection and the corresponding SSH login.

Records created:

```
node=test.bivio.net type=USER msg=audit(06/17/2020 22:36:16.003:10733)
: pid=9874 uid=root auid=root ses=45
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.17 22:36:16 LOG5[5]: Service [tls-login]
accepted connection from 192.168.120.125:34888 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 22:36:16.015:10735)
: pid=9876 uid=root auid=root ses=45
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.17 22:36:16 LOG5[5]: s_connect: connected
127.0.0.1:8275 exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'

node=test.bivio.net type=USER_START msg=audit(06/17/2020
22:36:25.857:10799) : pid=9892 uid=root auid=admin ses=50
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:session_open
grantors=pam_selinux,pam_loginuid,pam_selinux,pam_namespace,pam_keyinit
,pam_keyinit,pam_limits,pam_unix,pam_tty_audit,pam_umask,pam_lastlog
```

```
acct=admin exe=/usr/sbin/sshd hostname=127.0.0.1 addr=127.0.0.1
terminal=ssh res=success'

node=test.bivio.net type=USER_LOGIN msg=audit(06/17/2020
22:36:25.917:10846) : pid=9892 uid=root auid=admin ses=50
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=127.0.0.1 terminal=/dev/pts/4
res=success'
```

Termination of a login session over TLS

The following are the relevant records for a user (admin) logging off from a TLS session, resulting in the user logging out as well as terminating the TLS connection.

Records created:

```
node=test.bivio.net type=USER_LOGOUT msg=audit(06/17/2020
23:05:25.060:11454) : pid=10167 uid=root auid=admin ses=55
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login id=admin
exe=/usr/sbin/sshd hostname=? addr=? terminal=/dev/pts/4 res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 23:05:25.067:11465)
: pid=10225 uid=root auid=root ses=54
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=User admin from localhost has logged out
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/3
res=success'

node=test.bivio.net type=USER msg=audit(06/17/2020 23:05:25.074:11467)
: pid=10229 uid=root auid=root ses=53
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.17 23:05:25 LOG5[0]: Connection closed:
487 byte(s) sent to TLS, 93 byte(s) sent to socket
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'
```

Failed authentication for a login session over TLS

The following are the relevant audit records for a failed login attempt over TLS. This audit trail shows three failed password attempts, and the termination of the TLS connection. Because the number of attempts exceeded the maximum allowed, the account was locked.

Records created:

```
node=test.bivio.net type=USER msg=audit(06/18/2020 12:29:56.086:11589)
: pid=11127 uid=root auid=root ses=57
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.18 12:29:56 LOG5[0]: Service [tls-login]
accepted connection from 192.168.120.125:57292 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/18/2020 12:29:59.732:11594)
: pid=11142 uid=root auid=root ses=58
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

```
msg='text=User admin logging in from localhost exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/3 res=success'

node=test.bivio.net type=USER_AUTH msg=audit(06/18/2020
12:30:03.855:11599) : pid=11145 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=127.0.0.1
addr=127.0.0.1 terminal=ssh res=failed'

node=test.bivio.net type=USER_AUTH msg=audit(06/18/2020
12:30:08.877:11600) : pid=11145 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=127.0.0.1
addr=127.0.0.1 terminal=ssh res=failed'

node=test.bivio.net type=USER_AUTH msg=audit(06/18/2020
12:30:13.808:11603) : pid=11145 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:authentication
grantors=? acct=admin exe=/usr/sbin/sshd hostname=127.0.0.1
addr=127.0.0.1 terminal=ssh res=failed'

node=test.bivio.net type=USER_LOGIN msg=audit(06/18/2020
12:30:16.417:11606) : pid=11145 uid=root auid=unset ses=unset
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login acct=admin
exe=/usr/sbin/sshd hostname=? addr=127.0.0.1 terminal=ssh res=failed'

node=test.bivio.net type=USER msg=audit(06/18/2020 12:30:16.418:11607)
: pid=11155 uid=root auid=root ses=58
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=User admin from localhost failed to login
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/3
res=success'

node=test.bivio.net type=USER msg=audit(06/18/2020 12:30:16.424:11608)
: pid=11157 uid=root auid=root ses=57
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.18 12:30:16 LOG5[0]: Connection closed:
351 byte(s) sent to TLS, 112 byte(s) sent to socket
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'
```

Failure to establish a TLS session

A TLS session may fail even before it gets to user authentication, if a connection cannot be negotiated. The relevant records are as follows.

Records created:

```
node=test.bivio.net type=USER msg=audit(06/18/2020 12:39:24.889:11624)
: pid=11205 uid=root auid=root ses=57
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.18 12:39:24 LOG5[2]: Service [tls-login]
accepted connection from 192.168.120.125:58196 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'
```

```
node=test.bivio.net type=USER msg=audit(06/18/2020 12:39:24.893:11625)
: pid=11206 uid=root auid=root ses=57
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.18 12:39:24 LOG3[2]: SSL_accept:
1420910A: error:1420910A:SSL
routines:tls_early_post_process_client_hello:wrong ssl version
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/18/2020 12:39:24.896:11626)
: pid=11207 uid=root auid=root ses=57
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=TLS-server: 2020.06.18 12:39:24 LOG5[2]: Connection reset: 0
byte(s) sent to TLS, 0 byte(s) sent to socket exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'
```

The above records were created when an TLSv1.1 connection was attempted to the TLS server. Similar records will be created for other errors and will contain explanatory text as above.

All use of the identification and authentication mechanism

Audit logs are created when a user is being identified and authenticated. For Administrator login, the authentication methods are password based and public key based. Once the Administrator is identified and authenticated, his/her credentials are checked every time the Administrator performs an action. For example, if the user searches for certain audit records, the credentials are verified to ensure that the user has permission to do so. The actual action is logged along with the user's identity, but no identification and authentication logs are created at this time since they were created at login time.

Note that the TLS server is not required to validate the remote user's certificate. The remote user must provide a password in order to be allowed to login.

Unsuccessful attempt to validate a certificate (TLS server)

Records created:

```
node=test.bivio.net type=USER msg=audit(06/19/2020 11:20:51.246:12161)
: pid=15230 uid=root auid=root ses=71
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SHA1
Fingerprint=74:EA:AB:61:8D:BE:D6:95:66:FA:41:E0:A5:67:5D:18:B6:A6:9E:91
- verification/CA failed: C = US, ST = CA, L = Pleasanton, O = Bivio,
CN = Example4|error 10 at 0 depth lookup: certificate has expired|error
/opt/TLS-server/stunnel.crt: verification failed|
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 11:08:54.850:12132)
: pid=14918 uid=root auid=root ses=71
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SHA1
Fingerprint=B2:3D:4B:BD:FF:31:64:8B:38:C9:A3:06:59:18:71:E1:97:6D:F3:11
- verification/CRL failed: C = US, ST = CA, L = Pleasanton, O = Bivio,
CN = Example3|error 23 at 0 depth lookup: certificate revoked|error
```

```
/opt/TLS-server/stunnel.crt: verification failed|
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 11:15:06.194:12146)
: pid=15061 uid=root auid=root ses=71
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SHA1
Fingerprint=D3:76:12:1C:07:1D:7E:46:1B:C2:DA:68:AD:E0:97:AF:E5:72:69:BC
- verification/CA failed: C = US, ST = California, L = Pleasanton, O =
Bivio Networks, OU = Engineering, CN = Server|error 7 at 0 depth
lookup: certificate signature
failure|140321238951744:error:04091068:rsa routines:int_rsa_verify:bad
signature:crypto/rsa/rsa_sign.c:228:|140321238951744:error:0D0C5006:asn
1 encoding routines:ASN1_item_verify:EVP
lib:crypto/asn1/a_verify.c:179:|error /opt/TLS-server/stunnel.crt:
verification failed| exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 11:25:59.038:12167)
: pid=15308 uid=root auid=root ses=71
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SHA1
Fingerprint=D2:57:F6:F5:9A:B8:E3:02:AE:1D:72:B0:B2:4E:F8:62:E8:61:FF:18
- verification/CA failed: C = US, ST = California, L = Pleasanton, O =
Bivio, CN = Server2|error 20 at 0 depth lookup: unable to get local
issuer certificate|error /opt/TLS-server/stunnel.crt: verification
failed| exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 12:06:51.850:12228)
: pid=15628 uid=root auid=root ses=74
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SHA1
Fingerprint=D2:57:F6:F5:9A:B8:E3:02:AE:1D:72:B0:B2:4E:F8:62:E8:61:FF:18
- verification/CRL failed: C = US, ST = California, L = Pleasanton, O =
Bivio, CN = Server2|error 8 at 0 depth lookup: CRL signature
failure|139935702542144:error:0407008A:rsa
routines:RSA_padding_check_PKCS1_type_1:invalid
padding:crypto/rsa/rsa_pk1.c:67:|139935702542144:error:04067072:rsa
routines:rsa_ossl_public_decrypt:padding check
failed:crypto/rsa/rsa_ossl.c:652:|139935702542144:error:0D0C5006:asn1
encoding routines:ASN1_item_verify:EVP
lib:crypto/asn1/a_verify.c:179:|error /opt/TLS-server/stunnel.crt:
verification failed| exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/0 res=success'
```

Note that certificates are identified by their SHA1 fingerprint during verification.

The first message above was issued when the server certificate (`stunnel.crt`) was expired. The second message was issued when the server certificate could not be validated against the CRL list. The third message was issued for a damaged certificate which could not be parsed. These checks were done when the certificate was being imported by the Administrator. The fourth message above was issued

when the server certificate could not be validated against the CA list. The fifth message above was issued when the downloaded CRL was corrupted.

Other similar error messages can result when validating certificates. All such error messages are listed below.

```
unable to get issuer certificate
unable to get certificate CRL
unable to decrypt certificate's signature
unable to decrypt CRL's signature
unable to decode issuer public key
certificate signature failure
CRL signature failure
certificate is not yet valid
CRL is not yet valid
certificate has expired
CRL has expired
format error in certificate's notBefore field
format error in certificate's notAfter field
format error in CRL's lastUpdate field
format error in CRL's nextUpdate field
out of memory
self signed certificate
self signed certificate in certificate chain
unable to get local issuer certificate
unable to verify the first certificate
certificate revoked
invalid CA certificate
invalid non-CA certificate (has CA markings)
path length constraint exceeded
proxy path length constraint exceeded
proxy certificates not allowed, please set the appropriate flag
unsupported certificate purpose
certificate not trusted
```

```
certificate rejected
application verification failure
subject issuer mismatch
authority and subject key identifier mismatch
authority and issuer serial number mismatch
key usage does not include certificate signing
unable to get CRL issuer certificate
unhandled critical extension
key usage does not include CRL signing
key usage does not include digital signature
unhandled critical CRL extension
invalid or inconsistent certificate extension
invalid or inconsistent certificate policy extension
no explicit policy
Different CRL scope
Unsupported extension feature
RFC 3779 resource not subset of parent's resources
permitted subtree violation
excluded subtree violation
name constraints minimum and maximum not supported
unsupported name constraint type
unsupported or invalid name constraint syntax
unsupported or invalid name syntax
CRL path validation error
```

Software integrity checking

Record created upon success:

```
node=test.bivio.net type=USER msg=audit(06/19/2020 14:40:20.520:12365) :
pid=16998 uid=root auid=root ses=79
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=Software integrity test: No problems found exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'
```

Records created upon failure:

```
node=test.bivio.net type=ANOM_RBAC_INTEGRITY_FAIL msg=audit(06/19/2020
14:36:54.611:12361) : pid=16969 uid=root auid=root ses=79
```

```
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='added=94550
removed=0 changed=0 exe=/usr/sbin/aide hostname=test.bivio.net addr=?
terminal=pts/0 res=failed'

node=test.bivio.net type=USER msg=audit(06/19/2020 14:37:52.802:12362) :
pid=16981 uid=root auid=root ses=79
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=Software integrity test: Failed, please check
/var/log/aide/aide.log exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/0 res=success'
```

Notes:

When software integrity checking fails, the cause of the failure (i.e. which files did not match the database) must be determined by examining the file `/var/log/aide/aide.log`.

Modification of the behavior of the handling of audit data, and modification of the behavior of the TSF

Modification of the behavior of the handling of audit data is not supported. Any change to the parameters in the `/etc/audit/auditd.conf` file is limited to those described in this document.

Modification of the behavior of the TSF involves changing one of the following files:

```
/etc/sysconfig/network-scripts/ifcfg-eno1

/etc/audit/auditd.conf

/opt/ssh-tunnel-client/ssh-tunnel-client.conf

/etc/security/pwquality.conf

/etc/ssh/sshd_config

/etc/ssh/ssh_config

/etc/ssh/ssh_host_rsa_key.pub

/etc/issue

/etc/issue.net

/etc/aide.conf

/opt/TLS-server/cert.cnf

/opt/TLS-server/stunnel.conf

/opt/TLS-server/ca.crt

/opt/TLS-server/stunnel.crt

/opt/ssh-tunnel-client/.ssh/ssh_config

/opt/ssh-tunnel-client/.ssh/id_rsa.pub

/opt/ssh-tunnel-client/.ssh/known_hosts

/opt/ssh-tunnel-client/ssh-tunnel-client.conf
```

```
/etc/bashrc
/etc/passwd
/etc/shadow
/etc/group
/etc/gshadow
/etc/sudoers
/etc/chrony.conf
/etc/chrony.keys
```

A modification of any of the files listed above by the Administrator will generate audit logs for the changes in the file. Specifically, the lines logged will be generated by the `diff` command:

```
diff <orginal file> <modified file>
```

For example, changing the `num_logs` entry in the `/etc/audit/auditd.conf` file with the command:

```
sudo vi /etc/audit/auditd.conf
```

will generate the following audit logs:

```
node=test.bivio.net type=USER msg=audit(06/19/2020 15:12:45.210:13407)
: pid=17389 uid=root auid=admin ses=82
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/audit/auditd.conf: Fri Jun 19 15:12:45 PDT 2020:
modified by admin running with uid=root exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 15:12:45.218:13435)
: pid=17392 uid=root auid=admin ses=82
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/audit/auditd.conf: Fri Jun 19 15:12:45 PDT 2020: 12c12
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 15:12:45.222:13444)
: pid=17393 uid=root auid=admin ses=82
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/audit/auditd.conf: Fri Jun 19 15:12:45 PDT 2020: <
max_log_file = 8 exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/19/2020 15:12:45.226:13453)
: pid=17394 uid=root auid=admin ses=82
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/audit/auditd.conf: Fri Jun 19 15:12:45 PDT 2020: ---
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'
```

```
node=test.bivio.net type=USER msg=audit(06/19/2020 15:12:45.230:13462)
: pid=17395 uid=root auid=admin ses=82
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/audit/auditd.conf: Fri Jun 19 15:12:45 PDT 2020: >
max_log_file = 6 exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'
```

The behavior of the TSF can also be affected by other configuration changes or starting/stopping of services. These will create audit logs as described earlier in the section titled “Security related information”. In order to determine exactly what changed, the USER_CMD logs (the Administrator’s command input into login shell) must be examined. Note that USER_CMD logs can only indicate the commands typed in, and not the result of any operation.

The behavior of the audit functionality is not configurable when the local audit storage space is full.

Trusted updates

Upon initiation of the update, the following record is created:

```
node=test.bivio.net type=USER msg=audit(06/24/2020 16:00:00.251:4470) :
pid=72036 uid=root auid=admin ses=24
subj=sysadm_u:sysadm_r:rpm_script_t:s0-s0:c0.c1023 msg='text=software
update: module=BiviOSPatch-bv-8.5.1-101.el7.x86_64.rpm action=initiate
result=success exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'
```

Upon successful update, the following record is created:

```
node=test.bivio.net type=USER msg=audit(06/24/2020 16:00:00.251:4470) :
pid=72036 uid=root auid=admin ses=24
subj=sysadm_u:sysadm_r:rpm_script_t:s0-s0:c0.c1023 msg='text=software
update: module=BiviOSPatch-bv-8.5.1-101.el7.x86_64.rpm action=ACCEPT
result=success exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'
```

If the update fails, the following record is created:

```
node=test.bivio.net type=USER msg=audit(06/24/2020 16:00:00.251:4470) :
pid=72036 uid=root auid=admin ses=24
subj=sysadm_u:sysadm_r:rpm_script_t:s0-s0:c0.c1023 msg='text=software
update: module=BiviOSPatch-bv-8.5.1-101.el7.x86_64.rpm action=update
result=failed exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'
```

Changes to time

Records created:

```
node=test.bivio.net type=USER_CMD msg=audit(06/19/2020
15:33:24.485:13818) : pid=17429 uid=admin auid=admin ses=82
subj=sysadm_u:sysadm_r:sysadm_sudo_t:s0-s0:c0.c1023
msg='cwd=/home/admin cmd=date -s Fri Jun 19 15:33:30 PDT 2020
exe=/usr/bin/sudo terminal=pts/0 res=success'
```

Notes:

The above two log is due to the Administrator who used the “date” command to change the time. The current time is the log timestamp. The new time is indicated in the arguments to the “date” command.

Termination of local session by session locking mechanism

The following records are created when the Administrator’s login is terminated due to the expiry of the session timeout.

```
node=test.bivio.net type=USER_END msg=audit(06/19/2020
16:37:58.603:16556) : pid=11038 uid=root auid=admin ses=88
subj=system_u:system_r:local_login_t:s0-s0:c0.c1023
msg='op=PAM:session_close
grantors=pam_selinux,pam_loginuid,pam_console,pam_selinux,pam_nam
espace,pam_keyinit,pam_keyinit,pam_limits,pam_systemd,pam_unix,pa
m_tty_audit,pam_umask,pam_lastlog acct=admin exe=/usr/bin/login
hostname=test.bivio.net addr=? terminal=ttyS0 res=success'
```

Termination of remote session by session locking mechanism

Records created:

```
node=test.bivio.net type=USER_END msg=audit(06/19/2020
16:45:16.168:17097) : pid=17989 uid=root auid=admin ses=93
subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:session_close
grantors=pam_selinux,pam_loginuid,pam_selinux,pam_namespace,pam_keyinit
,pam_keyinit,pam_limits,pam_unix,pam_tty_audit,pam_umask,pam_lastlog
acct=admin exe=/usr/sbin/sshd hostname=192.168.2.10 addr=192.168.2.10
terminal=ssh res=success'
```

Notes:

This log indicates the termination of the Administrator’s login due to expiry of the session timeout.

Initiation of trusted channel (SSH client)

Records created (only relevant records are shown):

```
node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.192:17333)
: pid=19959 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Starting
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.278:17338)
: pid=19967 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Connecting to
192.168.120.125 [192.168.120.125] port 22. exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.280:17339)
: pid=19968 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

```
msg='text=ssh-tunnel-client 192.168.120.125: Connection established.
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.308:17345)
: pid=19974 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Authenticating to
192.168.120.125:22 as 'root' exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.513:17374)
: pid=20005 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Next authentication
method: publickey exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.532:17377)
: pid=20008 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Authentication succeeded
(publickey). exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.536:17378)
: pid=20009 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Authenticated to
192.168.120.125 ([192.168.120.125]:22). exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.539:17379)
: pid=20010 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Local connections to
localhost:7750 forwarded to remote address localhost:514
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 11:47:10.549:17382)
: pid=20013 uid=root auid=root ses=103
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Local forwarding listening
on 127.0.0.1 port 7750. exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 12:43:27.236:17383)
: pid=20223 uid=root auid=admin ses=106
subj=sysadm_u:sysadm_r:sysadm_sudo_t:s0-s0:c0.c1023 msg='text=Started
SSH tunnel client to 192.168.120.125:514 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

The last record that shows that the administrator started the trusted channel may occur anywhere in the above sequence of records, and not necessarily always at the end.

Termination of trusted channel (SSH client)

Records created:

```
node=test.bivio.net type=USER msg=audit(06/20/2020 12:47:52.073:18252)
: pid=20235 uid=root auid=admin ses=106
subj=sysadm_u:sysadm_r:sysadm_sudo_t:s0-s0:c0.c1023 msg='text=Stopped
SSH tunnel client to 192.168.120.125:514 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

The above record shows that the Administrator terminated the trusted channel.

Failure of the trusted channel (public key)

The following records are created when the trusted channel fails due to public-key based authentication failure. In this case, the failure occurred because the server did not have the client's authenticated public key. Note that connections are always initiated by the client side, that is, from the Bivio 6310-NC system. This can be seen from the bolded audit log in the list below, which shows that the client is **connecting to** the server.

```
node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:08.686:19351)
: pid=20410 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Starting
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:08.760:19352)
: pid=20414 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: OpenSSH_8.0p1, OpenSSL
1.1.1c FIPS  28 May 2019 exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:08.768:19356)
: pid=20418 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Connecting to
192.168.120.125 [192.168.120.125] port 22. exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:08.770:19357)
: pid=20419 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Connection established.
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:08.997:19392)
: pid=20456 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Next authentication
method: publickey exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'
```

```
node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:09.004:19395)
: pid=20459 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: No more authentication
methods to try. exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:09.007:19396)
: pid=20460 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: root@192.168.120.125:
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'

node=test.bivio.net type=USER msg=audit(06/20/2020 13:33:09.011:19397)
: pid=20461 uid=root auid=root ses=108
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=ssh-tunnel-client 192.168.120.125: Exited
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/0
res=success'
```

Running cryptographic tests (on the OpenSSL cryptography suite)

The following record is created upon success:

```
node=test.bivio.net type=USER msg=audit(06/20/2020 21:27:35.378:19778)
: pid=96038 uid=root auid=root ses=120
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SSL: All tests PASSED exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/0 res=success'
```

The following record is created upon failure:

```
node=test.bivio.net type=USER msg=audit(06/20/2020 21:33:21.227:19778)
: pid=96038 uid=root auid=root ses=120
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=SSL: Some tests FAILED; please check /var/log/ssltests.log
for details exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/0 res=success'
```

Modification, deletion, generation/import of cryptographic keys

The following records are created whenever keys are deleted using the `zeroize-keyfile` command. When the "`-d`" parameter is used, the key is zeroized and deleted, as indicated in the second record below.

```
node=test.bivio.net type=USER msg=audit(06/21/2020 15:23:35.453:19890)
: pid=97386 uid=root auid=admin ses=126
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=Key file /opt/TLS-server/stunnel.key zeroized
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'
```

```
node=test.bivio.net type=USER msg=audit(06/21/2020 15:22:31.217:19889)
: pid=97369 uid=root auid=admin ses=126
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='text=Key file /opt/TLS-server/testcerts/www.example2.com.key.pem
zeroized and deleted exe=/usr/sbin/auditctl hostname=test.bivio.net
addr=? terminal=pts/1 res=success'
```

Deleting keys using the `rm` command creates the following audit records (among others).

```
node=test.bivio.net type=PROCTITLE msg=audit(06/21/2020
15:39:26.424:21999) : proctitle=sudo rm stunnel.crt

node=test.bivio.net type=PATH msg=audit(06/21/2020 15:39:26.424:21999)
: item=1 name=stunnel.crt inode=137367307 dev=fd:00 mode=file,640
ouid=root ogid=root rdev=00:00 obj=sysadm_u:object_r:usr_t:s0
nametype=DELETE cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
cap_frootid=0

node=test.bivio.net type=PATH msg=audit(06/21/2020 15:39:26.424:21999)
: item=0 name=/opt/TLS-server inode=137367474 dev=fd:00 mode=dir,755
ouid=root ogid=root rdev=00:00 obj=system_u:object_r:usr_t:s0
nametype=PARENT cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
cap_frootid=0

node=test.bivio.net type=CWD msg=audit(06/21/2020 15:39:26.424:21999) :
cwd=/opt/TLS-server

node=test.bivio.net type=SYSCALL msg=audit(06/21/2020
15:39:26.424:21999) : arch=x86_64 syscall=unlinkat success=yes exit=0
a0=0xffffff9c a1=0x55eac2d3a640 a2=0x0 a3=0x0 items=2 ppid=97579
pid=97582 auid=admin uid=root gid=root euid=root suid=root fsuid=root
egid=root sgid=root fsgid=root tty=pts1 ses=127 comm=rm exe=/usr/bin/rm
subj=sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023 key=successful-delete
```

Importing a signed certificate creates the following audit log, in addition to other validation related logs:

```
node=test.bivio.net type=USER msg=audit(06/21/2020 15:52:00.941:24440)
: pid=97807 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Imported
certificate /opt/TLS-server/stunnel.crt exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

Importing a CA certificate bundle creates the following audit log, in addition to other validation related logs:

```
node=test.bivio.net type=USER msg=audit(06/21/2020 15:59:04.726:26263)
: pid=97936 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Imported CA
certificates /opt/TLS-server/ca.crt exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

Generating a key and a certificate request for the TLS login server creates the following audit logs, among others:

```
node=test.bivio.net type=USER msg=audit(06/21/2020 16:04:33.988:27246)
: pid=98020 uid=root auid=admin ses=127
```

```
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
key /opt/TLS-server/stunnel.key exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 16:04:56.132:27302)
: pid=98023 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
certificate request /opt/TLS-server/req.pem exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

Similar messages are created when a self-signed certificate is generated, and the last message in the sequence looks as follows:

```
node=test.bivio.net type=USER msg=audit(06/21/2020 16:10:36.633:27962)
: pid=98057 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
self-signed certificate /opt/TLS-server/stunnel.crt
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'
```

The following audit logs are created when the SSH host keys (public and private) are regenerated using the command `ssh-host-keygen`:

```
node=test.bivio.net type=USER msg=audit(06/21/2020 16:49:50.523:31855)
: pid=98478 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Key file
/etc/ssh/ssh_host_rsa_key zeroized and deleted exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 16:49:50.529:31882)
: pid=98480 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Deleted SSH
host public key /etc/ssh/ssh_host_rsa_key.pub exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 16:49:53.888:31983)
: pid=98485 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
key /etc/ssh/ssh_host_rsa_key exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 16:49:53.892:31992)
: pid=98486 uid=root auid=admin ses=127
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
public key /etc/ssh/ssh_host_rsa_key.pub exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

The following logs are created when the `ssh-tunnel-keygen` utility is used to generate public and private keys for the SSH tunnel client.

```
node=test.bivio.net type=USER msg=audit(06/21/2020 19:16:10.327:1587) :
pid=2739 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Key file
/opt/ssh-tunnel-client/.ssh/id_rsa zeroized and deleted
```

```
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 19:16:10.338:1614) :
pid=2741 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Deleted key
/opt/ssh-tunnel-client/.ssh/id_rsa.pub exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 19:16:13.936:1680) :
pid=2744 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
key /opt/ssh-tunnel-client/.ssh/id_rsa exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020 19:16:13.938:1689) :
pid=2745 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023 msg='text=Generated
public key /opt/ssh-tunnel-client/.ssh/id_rsa.pub
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=? terminal=pts/1
res=success'
```

Configuration of a new time server and removal of an existing time server

Time servers for the NTP client (chronyd) are stored in `/etc/chrony.conf`. Configuration of new time servers and removal of existing servers are accomplished by using `vi` to edit this file, make the necessary changes, and then saving the file. The changes between the old and new versions of the file will be written to the audit log files.

The following logs are created when a new server is added to the list of current servers.

```
node=test.bivio.net type=USER msg=audit(06/21/2020
19:38:05.527:8341) : pid=3069 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:38:05 PDT 2020:
modified by admin running with uid=root exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020
19:38:05.535:8369) : pid=3072 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:38:05 PDT 2020: 5a6
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020
19:38:05.539:8378) : pid=3073 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:38:05 PDT 2020: >
server2 192.168.120.155 iburst key 4 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

The following logs are created when an existing server is removed from the list of current servers.

```
node=test.bivio.net type=USER msg=audit(06/21/2020
19:34:30.986:6929) : pid=3007 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:34:30 PDT 2020:
modified by admin running with uid=root exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020
19:34:30.994:6957) : pid=3010 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:34:30 PDT 2020: 6d5
exe=/usr/sbin/auditctl hostname=test.bivio.net addr=?
terminal=pts/1 res=success'

node=test.bivio.net type=USER msg=audit(06/21/2020
19:34:30.999:6966) : pid=3011 uid=root auid=admin ses=6
subj=sysadm_u:sysadm_r:auditctl_t:s0-s0:c0.c1023
msg='text=/etc/chrony.conf: Sun Jun 21 19:34:30 PDT 2020: <
server2 192.168.120.155 iburst key 4 exe=/usr/sbin/auditctl
hostname=test.bivio.net addr=? terminal=pts/1 res=success'
```

Similar logs are created when /etc/chrony.keys is modified.

10. CONFIGURING THE SSH TUNNEL CLIENT FOR TRANSPORTING AUDIT LOGS TO REMOTE SERVER

The SSH tunnel client is used to send audit records generated locally on the Bivio 6310-NC to a remote syslog server over a secure connection. The process works as follows. Upon boot up, the Bivio 6310-NC will connect to the remote syslog server via the SSH tunnel client. Audit records generated on the Bivio 6310-NC are stored locally, and simultaneously forwarded to the SSH tunnel client, which will convey the records securely to the remote syslog server. The local store is not “cleared” per se, but space will be reclaimed by the process of rotating logs and deleting older archived log files.

The SSH tunnel client is installed in `/opt/ssh-tunnel-client`. The cryptographic engine parameters (ciphers, MACs) for the SSH tunnel client are already configured and need not be set by the Administrator. The Administrator only has to create the keys required for setting up the tunnel, as described below.

The SSH connection between the Bivio 6310-NC and the remote server needs to be mutually authenticated. Public key based authentication is preferred since password based authentication will require the connection to be made interactively, and not automatically at boot up time.

The following steps describe how to generate a public/private key pair for the tunnel client on the local system to be able to connect to the remote server using public key based authentication.

1. Delete any existing keys before generating new keys. The private key must be zeroized before it is deleted. The public key can be deleted using the Linux “rm” command.

```
sudo zeroize-keyfile –d /opt/ssh-tunnel-client/.ssh/id_rsa
```

```
sudo rm /opt/ssh-tunnel-client/.ssh/id_rsa.pub
```

2. Create a public/private key pair for the tunnel client in `/opt/ssh-tunnel-client/.ssh`:

   ```
   sudo /opt/ssh-tunnel-client/ssh-tunnel-keygen
   ```

   Just hit "Enter" when prompted for a passphrase (do not enter a passphrase).

3. Copy the public key file (`/opt/ssh-tunnel-client/.ssh/id_rsa.pub`) to the appropriate location on the remote server (if it is a Linux server, the location would most likely be `~<user-name>/.ssh/authorized_keys`, where `<user-name>` is the name of the user on the remote system, and the public key is appended to the end of the `authorized_keys` file). The secure copy utility (`scp`) can be used to copy the public key to the remote system securely. The private key stays on the Bivio 6310-NC.

4. Modify the values of the variables in the `/opt/ssh-tunnel-client/ssh-tunnel-client.conf` file appropriately:
   a. `ENABLED` (set to "yes")
   b. `USERNAME` (set to <user-name>, the name of the user on the remote system)
   c. `REMOTE_IP_ADDRESS` (set to IP address of remote server)
   d. `REMOTE_SYSLOG_PORT` (port at which remote syslog server is listening)

   Note: These fields will already be present in the file. Using the `vi` editor, edit the file `/opt/ssh-tunnel-client/ssh-tunnel-client.conf` to set the desired values for these parameters:

   ```
   sudo vi /opt/ssh-tunnel-client/ssh-tunnel-client.conf
   ```

   Save the file and quit `vi` when done with editing.

5. In order to create the `/opt/ssh-tunnel-client/.ssh/known_hosts` file, execute the script `/opt/ssh-tunnel-client/ssh-tunnel-client` as shown. Enter the required passwords and "`yes`" to the prompt which displays the remote server's public key fingerprint. This will set up the `/opt/ssh-tunnel-client/.ssh/known_hosts` file correctly with the remote host's public key. Now terminate it using by typing a `ctrl-c` character into the terminal.

   ```
   admin@test ssh-tunnel-client$ sudo ./ssh-tunnel-client
   [sudo] password for admin:
   The authenticity of host '192.168.120.154 (192.168.120.154)'
   can't be established.
   RSA key fingerprint is
   93:e8:4f:b9:4e:cc:84:4b:7f:ff:22:81:61:25:35:b2.
   Are you sure you want to continue connecting (yes/no)? yes
   admin@test ssh-tunnel-client$
   ```

6. Execute the script `/opt/ssh-tunnel-client/start-tunnel-client` for manually starting the client for the first time, after it has been configured. The tunnel will then be

automatically started whenever the system boots up, if `ENABLED` is set to "yes". **The tunnel will be monitored by the system and restarted if it stops.**

```
admin@test ~$ cd /opt/ssh-tunnel-client
admin@test ssh-tunnel-client$ sudo ./start-tunnel-client
```

7. Execute the script `/opt/ssh-tunnel-client/stop-tunnel-client` for manually stopping the client. Once this is done, the client will not start up even when the system is rebooted, unless it is manually started up again.

```
admin@test ~$ cd /opt/ssh-tunnel-client
admin@test ssh-tunnel-client$ sudo ./stop-tunnel-client
```

Note that the SSH configuration by the SSH tunnel client resides in the file `/opt/ssh-tunnel-client/.ssh/ssh_config`. The differences from the default values for SSH clients in general are as follows:

```
Host *
IdentityFile /opt/ssh-tunnel-client/.ssh/id_rsa
UserKnownHostsFile /opt/ssh-tunnel-client/.ssh/known_hosts
RekeyLimit 1G 1h
Ciphers aes128-cbc,aes256-cbc
HostKeyAlgorithms ssh-rsa
KexAlgorithms diffie-hellman-group14-sha1
MACs hmac-sha2-256,hmac-sha2-512
PasswordAuthentication no
```

Other identity files are not permitted. Also note that the `none` option for MAC is not allowed.

In order to run the SSH tunnel client using password authentication, the following steps must be used instead of the above procedure.

1. Edit the file `/opt/ssh-tunnel-client/.ssh/ssh_config` using the `vi` editor and set the value of the "`PasswordAuthentication`" variable to "`yes`", and the value of "`PubkeyAuthentication`" to "`no`". Save the file and quit the editor.
2. When password authentication is used, the SSH tunnel client cannot be started automatically as the password must be keyed in. Type the following command into a login shell/console to start the SSH tunnel client:

```
admin@test ~$ sudo /opt/ssh-tunnel-client/ssh-tunnel-client
```

3. If the file `/opt/ssh-tunnel-client/.ssh/known_hosts` does not exist, or does not have a key for the remote server, the remote key fingerprint will be displayed, and a prompt will be displayed to the Administrator. To continue, type "yes". If the key is already present, this prompt will not be displayed.
4. Enter the password for the remote server when prompted for it (this may be after the prompt for the Administrator's password).

5. If authenticated properly, there will not be any output and the SSH client will be running at this time. The SSH tunnel client may be stopped by entering a ctrl-C (^C) character at any time.
6. In order to let the SSH tunnel client run in the background so the Administrator can log off, enter the ^Z character, and type the command "bg" at the shell prompt, as shown below.

```
admin@test ~$ sudo /opt/ssh-tunnel-client/ssh-tunnel-client
root@192.168.120.154's password:
^Z
[1]+  Stopped                 sudo /opt/ssh-tunnel-client/ssh-
tunnel-client
admin@test ~$ bg
[1]+ sudo /opt/ssh-tunnel-client/ssh-tunnel-client &
admin@test ~$
```

7. To stop the SSH tunnel running in the background, the Administrator should login and execute the following command. The message printed out by the command should be ignored.

```
admin@test ~$ sudo /opt/ssh-tunnel-client/stop-tunnel-client
Application ssh-tunnel-client is already stopped.
admin@test ~$
```

Mitigating Against CVE-2020-14145

CVE-2020-14145 describes a scenario where an attack is possible when connecting to a malicious "man-in-the-middle" server. This scenario applies when the client tries to connect to the server before the server's public key has been locally cached in the `/opt/ssh-tunnel-client/.ssh/known_hosts` file. This can be mitigated by the following actions on the administrator's part.

1. If possible, the administrator should populate the `known_hosts` file manually with a known good host key (the host key can be copied to this file).
2. If the key was cached by the client side, the administrator should ensure, by using means external to the system, that the host key stored in the `known_hosts` file is good. The RSA key fingerprint that is printed out may be used for this purpose.
3. Finally, the administrator must connect only to servers with known good host keys.

11. ADMIN LOGIN AUTHENTICATION

Two methods for user authentication are supported: password and public key certificates (SSH-RSA).

In both cases, the administrator will need to login using an SSH client program. The SSH client can be either a command line utility (like the `ssh` program in Linux), or it may be a GUI. In any case, the parameters needed are the same: a username, the IP address of the remote node, and a password if password authentication is used. A password is not required if public key authentication is used.

If the SSH client is a command line utility, as in Linux, it is usually invoked as follows:

```
ssh <username>@<ip_address>
```

where `<username>` is the name of the user on the remote node, and `<ip_address>` is the IP address of the remote node.

If password authentication is used, the SSH client utility will display a prompt as follows:

```
password:
```

Once the user enters the correct password, the login will be successful. If public key authentication is used, and the SSH client is configured to use the correct public key, the login will be successful without a password prompt being displayed.

If the SSH client is GUI based, the parameters <username>, <ip_address>, and the password will need to be filled into the appropriate fields displayed by the client. If public key authentication is used, and the SSH client is configured to use the correct public key, the password field will usually not be displayed, and the login will be successful if the public key has been configured correctly.

Password based authentication

A user password is a sequence of characters. Any printable character is a valid character for use in a password.

Password quality parameters, including the minimum password length, are set in the file /etc/security/pwquality.conf. The parameter minlen (the minimum password length) must be set to the desired value by the Administrator, as follows. Edit the parameter file with the command:

```
sudo vi /etc/security/pwquality.conf
```

Save the file and quit vi when done with editing.

Minimum password length shall be configurable to between 9 and 128 characters. To set the minimum password length to 16, ensure that the file has the following line in it (either change an existing line, or add a new one):

```
minlen = 16
```

In order for the minimum password setting to work as expected, the following parameters in /etc/security/pwquality.conf must be set explicitly to zero, as shown below:

```
dcredit = 0
ucredit = 0
lcredit = 0
ocredit = 0
```

Password hashes are stored in /etc/shadow using the SHA512 hashing algorithm.

Note: CC Evaluation Activities only cover the minlen parameter; effects of changing other parameters are not covered in the CC Evaluation.

Strong passwords

The Administrator must use strong passwords in order to maximize security. A strong password is one that has three or more of the following attributes:

Has at least 9 characters

Includes numbers, symbols, upper and lowercase alphabets

Isn't a dictionary word or combination of dictionary words

Doesn't rely on obvious substitutions (such as 0 for 0, 1 for l, and so on)

Public key based authentication

Public key based authentication is turned on by default and will be attempted before password based authentication.

If there is a need to turn off password based authentication and use only public key based authentication, the `PasswordAuthentication` parameter in `/etc/ssh/sshd_config` must be set to "`no`" (default value is "`yes`"). Using the "vi" editor, edit the file `/etc/ssh/sshd_config` as follows:

```
sudo vi /etc/ssh/sshd_config
```

Locate the keyword `PasswordAuthentication` and change its value to "no".

Save the file and quit `vi`. Restart the SSH server as follows:

```
sudo service sshd restart
```

Note: Password authentication can be turned on again (without impacting public key based authentication) by setting the the keyword `PasswordAuthentication` back to "yes" using the above procedure.

Note: Do not change the value of the `PasswordAuthentication` parameter until a public key has been installed for the Administrator, and the login procedure has been tested. This is described below.

Only use of SSH-RSA type public keys, with private key lengths of 2048 bits (or higher), is permitted for CC conformance. Public and private keys may be generated on any system.

Below is a procedure for using the Bivio 6310-NC system itself for key generation (stated here only for illustrative purposes; private keys are best generated on the system where they will be used):

1. While logged in as `admin`, use the `ssh-keygen` utility to generate a public/private key pair
   ```
   ssh-keygen –b 2048
   ```
2. This will create the public key (`id_rsa.pub`) and the private key (`id_rsa`) in `~/.ssh` directory.
3. Copy the public key to the "`authorized_keys`" file in the same directory
   ```
   ssh-copy-id localhost
   ```
4. Transport the public and private keys to the client system securely (using `scp`), to the appropriate locations on the system used for logging in
5. Zeroize and delete the private key on the Bivio 6310-NC system as follows
   ```
   zeroize-keyfile -d ~/.ssh/id_rsa
   ```

The next time the Administrator logs in, a password will not be required.

If the keys are generated on a remote system (which is the normal case), only the public key needs to be copied to the Bivio 6310-NC. Using a secure copy utility on the remote system, copy the public key to the following file on the Bivio 6310-NC:

```
~admin/.ssh/authorized_keys
```

Note that the `.ssh` directory should have permissions set to `700`, and the `authorized_keys` file must have permissions set to `600`.

Login session timeout

User login sessions are automatically timed out, by default, after 10 minutes. This is configured in the global file `/etc/bashrc`. The variable that controls the timeout is `TMOUT` and is set, by default, to 600 (seconds). To change this value, edit the file `/etc/bashrc` using the `vi` editor, set the `TMOUT` variable to the desired value in seconds, save the file, and exit the editor.

```
sudo vi /etc/bashrc
```

The new session timeout will take effect in subsequent login sessions after the user logs out and logs in again.

Login warning banner

For local login over the serial console, the local login warning banner is set in the file `/etc/issue`. To change the local login warning banner, edit the file `/etc/issue` using the vi editor as follows:

```
sudo vi /etc/issue
```

For remote login over SSH and TLS, the remote login warning banner is set in the file `/etc/issue.net`. To change the remote login warning banner, edit the file `/etc/issue.net` using the vi editor as follows:

```
sudo vi /etc/issue.net
```

Save the file and quit `vi` when done with editing.

Changing the Administrator's password

The Administrator may change his/her password using the `passwd` utility, as follows:

```
admin@test ~$ passwd
Changing password for user admin.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
admin@test ~$
```

12. SSH SERVER CONFIGURATION

The SSH server configuration resides in `/etc/ssh/sshd_config`. As shipped from the factory, the Bivio 6310-NC is configured with cryptographic engine parameters that are CC conformant. They should not be changed by the Administrator. For reference, the changes to `/etc/ssh/sshd_config` from its default values are as follows.

Host keys other than RSA must be commented out:

```
HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

The following parameters must be set as follows:

```
RekeyLimit 1G 1H
Ciphers aes128-cbc,aes256-cbc
KexAlgorithms diffie-hellman-group14-sha1
MACs hmac-sha2-256,hmac-sha2-512
PermitRootLogin no
HostKeyAlgorithms rsa-sha2-256,rsa-sha2-512,ssh-rsa
PubkeyAcceptedKeyTypes ssh-rsa,rsa-sha2-256,rsa-sha2-512
```

Note that the `none` MAC algorithm is not allowed for the SSH server.

The Administrator may wish to change the host key, which is described below. Note that this is not necessary under normal circumstances.

CC Evaluation Activities Coverage

The CC Evaluation Activities cover the following parameter settings (which are already set at the factory). Effects of changing other parameters are not covered by the CC Evaluation Activities.

Protocol version:	2
Public key authentication:	SSH-RSA
Encryption algorithms:	aes128-cbc, aes256-cbc
Key exchange:	Diffie-Hellman-Group14-SHA1
Integrity algorithms:	hmac-sha2-256, hmac-sha2-512
Rekey limit:	1 GB of data, or 1 Hour
Host key algorithms:	ssh-rsa, rsa-sha2-256, rsa-sha2-512
Allowed client public key types:	ssh-rsa, rsa-sha2-256, rsa-sha2-512

Generating new server keys

SSH server keys are stored in the directory `/etc/ssh`. **The `ssh-host-keygen` utility must be used to generate new RSA keys, namely, the key pair `ssh_host_rsa_key` and `ssh_host_rsa_key.pub`.** Before generating new keys, the Administrator must destroy the old private key file with the following command:

```
sudo zeroize-keyfile -d /etc/ssh/ssh_host_rsa_key
```

```
sudo rm /etc/ssh/ssh_host_rsa_key.pub
```

The public key may be deleted as usual using the Linux `rm` command. The new key pair can then be generated as follows:

```
sudo -s ssh-host-keygen
```

Just hit Enter if prompted for a password – do not enter a passphrase. This will generate the key files `/etc/ssh/ssh_host_rsa_key` and `/etc/ssh/ssh_host_rsa_key.pub`, which are respectively the private and public keys for the host.

This is best done from the serial console since the SSH server must then be restarted as follows:

```
sudo service sshd restart
```

13. SSH CLIENT CONFIGURATION

The SSH client configuration resides in `/etc/ssh/ssh_config`. As shipped from the factory, the Bivio 6310-NC is configured with cryptographic engine parameters that are CC conformant. For reference, the changes to `/etc/ssh/ssh_config` from its default values are as follows: the following additional lines have been added:

```
Host *
IdentityFile ~/.ssh/id_rsa
RekeyLimit 1G 1H
Ciphers aes128-cbc,aes256-cbc
HostKeyAlgorithms ssh-rsa
KexAlgorithms diffie-hellman-group14-sha1
MACs hmac-sha2-256,hmac-sha2-512
```

Note that the `none` MAC algorithm is not allowed for the SSH client.

CC Evaluation Activities Coverage

The CC Evaluation Activities cover the following parameter settings (which are already set at the factory). Effects of changing other parameters are not covered by the CC Evaluation Activities.

Protocol version:	2
Public key authentication:	SSH-RSA
Encryption algorithms:	aes128-cbc, aes256-cbc
Key exchange:	Diffie-Hellman-Group14-SHA1
Integrity algorithms:	hmac-sha2-256, hmac-sha2-512
Rekey limit:	1 GB of data, or 1 Hour

14. TLS SERVER CONFIGURATION

The TLS server resides in `/opt/TLS-server`. It is implemented using the `stunnel` proxy, which proxies for a login server that provides a Telnet type login service over TLSv1.2.

The unencrypted server listens on port 8275 for tcp connections, within local host only. The TLS proxy, `stunnel`, is configured to accept incoming TLS version 1.2 connections on port 27777 and connects to the login server over tcp port 8275 within the local host.

The ciphers supported by the TLS server (from the OpenSSL list of supported ciphers) are:

```
AES128-SHA
ECDHE-RSA-AES256-GCM-SHA384
```

The elliptic curve used (from the OpenSSL list of supported curves) is:

```
prime256v1
```

The `stunnel` proxy is configured to present a server certificate to the client, when the client makes a connection. As such, it needs to be configured with an X509 public key certificate, a private key and a CA list. The CRL list will be downloaded as needed using the CDP field (if present) in the certificate being validated. The client program is not requested to present a public key certificate and will be permitted to connect without presenting a certificate. The client's login password will be authenticated by the server.

The validation of the server certificate is done against the configured CA list and any CRL indicated in the certificate as a CRL Distribution Point (CDP). If the server certificate contains a CDP, the CRL will be downloaded from the CDP when the server certificate is imported, and when the server is started. The downloaded CRL will overwrite any existing CRL list. The CDP is specified using syntax similar to the one shown below, for example:

```
URI:http://192.168.120.1/install/certs/crls/crl.pem
```

Currently, `http` is the only protocol supported. The host name can be specified as a fully qualified name (FQN) if DNS is configured, otherwise an IP address as shown above must be used. If the host is accessible but the file is not present, it is treated as an error and the import operation will fail. If the host is not accessible due to a network problem, any import operation will fail (if the certificate is being imported) and attempts to start the server will fail. If a CDP is not present in the certificate, the certificate will be considered valid.

The CA list is also validated when it is imported. The CA list must be imported before the server certificate is imported, since the validation of the server certificates depends on the CA list. Each certificate in the CA list is validated in the same way as the server certificate. The CA list is also validated whenever the server certificate is validated, although they are both imported separately.

The `stunnel` proxy and its configuration file are described in detail in [5] (not required reading), but there are no configuration parameters for the Administrator to set since the requisite values are set at the factory. Thus, some of the following details (in the rest of this section) are for informational purposes only, for the Administrator to gain an understanding of how the TLS server works.

The configuration file is:

```
/opt/TLS-server/stunnel.conf
```

The public key certificate, in PEM format, is:

```
/opt/TLS-server/stunnel.crt
```

The private key file is:

```
/opt/TLS-server/stunnel.key
```

The CRL list is overwritten when CRLs are downloaded from the specified CDPs. As such, this can be ignored by the administrator. PEM and DER formats are supported for CRLs.

```
/opt/TLS-server/crl.crt
```

The trusted CA certificate list is:

```
/opt/TLS-server/ca.crt
```

This file contains all the trusted CA certificates, concatenated into a single file.

Thus, the relevant parameters in the configuration file are set as follows in the factory, and the Administrator does not need to change anything. The content of /opt/TLS-server/stunnel.conf should be as follows:

```
foreground = yes
debug = notice

[tls-login]
accept = 27777
options = NO_SESSION_RESUMPTION_ON_RENEGOTIATION
options = NO_TICKET
connect = 8275
ciphers = AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384
curve = prime256v1
cert = /opt/TLS-server/stunnel.crt
key = /opt/TLS-server/stunnel.key
sslVersion = TLSv1.2
requireCert = no
CAfile = /opt/TLS-server/ca.crt
;CRLfile = /opt/TLS-server/crl.crt
```

Note that comments start with a ";" character. The `ca.crt` file must be supplied by the Administrator, and the `crl.crt` file is created as needed from CDPs.

Rekeying the TLS server

The procedure for generating a new private key and a certificate request is described in the next section. The existing private key must be zeroized as follows, before it is replaced by the new private key:

```
sudo zeroize-keyfile -d /opt/TLS-server/stunnel.key

sudo rm /opt/TLS-server/stunnel.crt
```

The old public key certificate may just be overwritten, or deleted using the Linux "`rm`" command as shown above. These are done automatically when the CSR (Certificate Signing Request) is created.

Note on first login via the TLS server

In order to login via TLS, the user must first establish a TLS v1.2 connection to port 27777. Once connected, the Telnet protocol may be used over this connection securely to initiate a login session. As described earlier, the best way to utilize this service will be to use a TLS enabled Telnet client. If that is not available, a regular Telnet client can be used via a TLS proxy.

When logging in via the TLS server for the first time, a dialogue similar to logging in via SSH will be presented. That is, the host's public key signature will be presented to the user, and the user will be asked to accept or reject it. After the first login, just as in SSH, this dialogue will not be presented for future logins via the TLS server.

CC Evaluation Activities Coverage

The only `ciphers` and `curve` parameters covered are from the above configuration. These are not to be changed by the Administrator.

15. MAKING A CERTIFICATE REQUEST AND INSTALLING FILES

Certificate request creation is described in [6], but this is not required reading for the Administrator since the procedure is described in this document.

The following command illustrates how a private key and a certificate request are generated for the TLS-server. Before issuing the command, the Administrator should have the following information handy:

- Country Name (two letter code, such as "US")
- State or Province (such as "California")
- Locality Name (such as "Pleasanton")
- Organization Name (for example, "Bivio Networks Inc")
- Organizational Unit Name (for example, "Engineering")
- Common Name (for example, "www.bivio.net")
- Email address (for example, myemail@bivio.net)

The commands to be executed are:

```
cd /opt/TLS-server

admin@test TLS-server$ sudo ./tls-server-csr-gen
```

An interactive session that creates a CSR is shown below:

```
Generating RSA private key, 2048 bit long modulus

.....................................................................................
.............+++

.....................................................................+++

e is 65537 (0x10001)

You are about to be asked to enter information that will be incorporated
```

```
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [US]:
State or Province Name [California]:
Locality Name []:.
Organization Name [Bivio Networks Inc]:
Organizational Unit Name []:.
Common Name []:www.bivio.net
Email Address []:it@bivio.net
```

This will create an unencrypted private key (`stunnel.key`), and a certificate request (`req.pem`). (Note: The file `/opt/TLS-server/cert.cnf` contains the certificate configuration and need not be edited by the Administrator.) The key file can now be used as the TLS server's private key, and the request file can be shipped to a CA to be signed. The signed certificate can then be used as the TLS server's public key certificate. The Bivio 6310-NC is not intended to be used as a CA.

The above procedure installs the private key in the correct location.

The signed certificate needs to be copied securely (from a remote system) to:

`/opt/TLS-server/stunnel.crt`.

A CA certificate list should be copied securely to:

`/opt/TLS-server/ca.crt`

Note: A "certificate list" is simply a list of concatenated certificates, separated by a newline character between adjacent certificates.

Note: Copying files to the system from a remote system is done in two steps, in order to prevent permission related problems. For example, the signed "`stunnel.crt`" file should first be copied securely to the Administrator's home directory. From there, it can then be copied to the correct location as follows:

```
admin@test ~$ sudo cp stunnel.crt /opt/TLS-server/stunnel.crt
```

The `tls-server-crt-gen` utility is provided to generate a private key and a self-signed certificate, for quick testing. This creates the files `stunnel.crt` and `stunnel.key`, and is invoked as follows:

```
/opt/TLS-server/tls-server-crt-gen
```

As before, prior to generating keys, the existing private key (if any) should be zeroized and deleted, and the public key should be just deleted (using "`rm`"). This is automatically done when `tls-server-crt-gen` is invoked.

Importing a server certificate

A signed CSR (the server certificate) is imported into the TLS server by executing the following commands, after the certificate has been copied into `/opt/TLS-server/stunnel.crt`:

```
cd /opt/TLS-server
sudo ./tls-import-cert
```

The process of importing the certificate automatically validates the certificate against the CA list (`ca.crt`) and any CRL specified in a CDP. A successful validation looks as follows (this may vary a lot depending on the certificates):

```
root@test TLS-server$ ./tls-import-cert

SHA1 Fingerprint=F6:75:2A:66:D3:77:29:22:7B:60:3D:F2:94:44:81:A5:2A:28:99:4C -
verification/CA successful

getcrl: CDP not present in /opt/TLS-server/tmp-crt

SHA1 Fingerprint=70:37:01:18:53:CD:C5:97:E4:36:B0:D4:46:E2:08:F3:47:1E:80:85 -
verification/CA successful

getcrl: CDP not present in /opt/TLS-server/tmp-crt

SHA1 Fingerprint=24:68:5F:DF:0E:B9:C9:8F:F4:2C:18:E9:E2:F2:FC:06:B6:D0:E9:B2 -
verification/CA successful

Downloading http://192.168.120.1/install/rh/extras/crl.pem ...

--2017-04-24 16:38:50--  http://192.168.120.1/install/rh/extras/crl.pem

Connecting to 192.168.120.1:80... connected.

HTTP request sent, awaiting response... 200 OK

Length: 1198 (1.2K) [text/plain]

Saving to: '/opt/TLS-server/tmpcrl'


100%[=============================================================================
==========>] 1,198       --.-K/s   in 0s


2017-04-24 16:38:50 (59.4 MB/s) - '/opt/TLS-server/tmpcrl' saved [1198/1198]


getcrl: successfully downloaded http://192.168.120.1/install/rh/extras/crl.pem

SHA1 Fingerprint=24:68:5F:DF:0E:B9:C9:8F:F4:2C:18:E9:E2:F2:FC:06:B6:D0:E9:B2 -
verification/CRL successful

Imported certificate /opt/TLS-server/stunnel.crt

root@test TLS-server$
```

For good measure, the server will also perform these validation steps every time the server is started. The results of the validation are audited, with appropriate error messages in case of failure.

Importing the CA list

The CA list should be imported before the server certificate is imported. The CA list is imported using the following commands, after it has been copied into the file: `/opt/TLS-server/ca.crt`.

```
cd /opt/TLS-server
sudo ./tls-import-ca
```

A successful outcome will look similar to the output shown above for importing the server certificate. Note that during the validation/verification process, certificates are referred to by their SHA1 fingerprints (since the file `ca.crt` may contain multiple certificates).

Before starting the TLS server for the first time

The following steps (described in detail earlier) should be executed by the administrator before starting the TLS server for the first time.

- Create or obtain the CA certificate bundle (`/opt/TLS-server/ca.crt`) and import it
- Generate a CSR (this will also install the server's private key)
- Import the signed CSR (`/opt/TLS-server/stunnel.crt`)

Starting and stopping the TLS server

The script `/opt/TLS-server/tls-login-ctl` can be used to start and stop the server as follows:

```
sudo /opt/TLS-server/tls-login-ctl start    (starts the server)

sudo /opt/TLS-server/tls-login-ctl stop     (stops the server)
```

The TLS server must be started manually every time the system is restarted (rebooted).

16. CRYPTOGRAPHIC TESTS

There is a full suite of cryptographic tests for OpenSSL in `/opt/ssl`. When the system is started (or rebooted), these tests are run automatically, and the results are placed in `/var/log/ssltests.log`. An audit log is created at the end of the tests, which indicates whether the tests succeeded or failed.

The audit logs (described earlier) must be examined after the system is rebooted to check whether the cryptographic tests passed or failed.

The tests can also be invoked manually by the Administrator as follows:

```
sudo ssl-test
```

If the audit log indicates failure (or the test itself, if manually run), the file `/var/log/ssltests.log` should be inspected for the failure reason.

The tests cover the following algorithms (among others), as used by the Bivio 6310-NC platform:

```
RSA 2048-bit in accordance to FIPS 186-4
ECC  p-256
FFC 2048-bit
RSA key establishment
EC key establishment
FF key establishment
AES in 128 and 256 bit, in CBC and GCM modes
RSA Digital Signature 2048 bit
SHA-1, SHA-256, SHA-384, SHA-512
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512
Random number generation tests
```

17. SOFTWARE INTEGRITY CHECKING

Software integrity checking on the Bivio 6310-NC is done using the AIDE package, which is installed and configured on the system by default. This is fully described in the man pages for `aide` and `aide.conf`. There are no administrator settable parameters, however. These tests are run when the system boots up.

The aide utility may also be invoked manually to check software integrity as follows:

```
sudo -s aide -c /etc/aide.conf --check
```

If any problems are discovered, an audit log is created. In this case, the log file `/var/log/aide/aide.log` should be examined to see the exact cause of the problem:

```
sudo vi /var/log/aide/aide.log
```

Once the problem is determined and rectified, the Administrator should manually update the database used for integrity checking by running the following command:

```
sudo -s aide -c /etc/aide.conf --init
sudo rm /var/lib/aide/aide.db.gz
sudo mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

Otherwise, the same error logs will be created every time software integrity checks are run.

An audit log is also created when the test runs successfully (as shown earlier). Thus, the audit logs indicate that the software integrity test did run during boot time, and also indicates whether the check was successful or not.

18. SELF-TESTS

When the system boots up, it performs a number of self-tests. These are as follows.

RDRAND tests

These are hardware tests performed within the Intel processor core and ensure that the system can generate high quality entropy. If these tests fail, the processer will not boot up. Bivio Support will need to be contacted for replacing the hardware.

Memory Tests

These are performed at very early stages of boot up, before the operating system gets loaded. If these tests fail, it is indicative of a hardware problem, and the system will not boot up. Bivio Support will need to be contacted for replacing either the memory, or the full system, depending on the failure diagnosis and support plan.

Cryptographic Tests

These are software tests, with a very low probability of failing, and may take up to five minutes to complete after the system boots up. Audit logs are created for successful as well as failed tests. If the audit logs indicate failure, the Administrator should immediately quarantine the system, and contact Bivio Support. Diagnosis will depend on the contents of the log file where the test results are stored.

Software Integrity Tests

These tests may take up to five minutes to complete after the system has booted up. Audit logs are created for successful and failed outcomes. If failure is indicated, the Administrator should immediately check the report in `/var/log/aide/aide.log`. If the failure is due changes that were done administratively and/or expected, procedures for updating the integrity database should be followed. If not, the system should be quarantined, and the audit logs should be examined for clues as to who made the changes and why.

19. SOFTWARE UPDATES

Software updates are always done as a patch to the system. The patch is provided as an RPM. Typically, a patch RPM may have a name as follows:

```
BiviOSPatch-8.5.1-101.el6.x86_64.rpm
```

The sequence number for the patch, starting at 101, will be incremented sequentially for new patches. A SHA-256 check sum will also be provided, with the file name:

```
BiviOSPatch-8.5.1-101.sum
```

Downloading the update

In order to obtain a patch, the Administrator must first contact Bivio Support (1-925-924-8606) and request an account on the Bivio support portal. When the Administrator logs in, patches will be available under a “Downloads” section, along with the release notes for the patch.

The Administrator must download the patch (over HTTPS connection to support portal) and the checksum file onto a local machine (for example, the Administrator’s workstation or PC). These files then must be securely copied to the Bivio 6310-NC (preferably, into a directory called “patches” in the

Administrator's home directory). The SHA-256 checksum must then be determined by running the command:

```
sha256sum BiviOSPatch-8.5.1-101.el6.x86_64.rpm
```

The output will look as follows (checksum will be different for actual patch):

```
15127ad93522209167bb8b37cd3b47b37439086833adfcc0748680cc565e7e34
/home/admin/patches/BiviOSPatch-8.5.1-101.el6.x86_64.rpm
```

The Administrator should verify that the checksum matches the contents of the downloaded checksum file. If the checksum does not match, the Administrator should contact Bivio support immediately.

Installing the update

If the checksum is okay, the update should be installed using the command:

```
sudo -s rpm -Uvh BiviOSPatch-8.5.1-101.el6.x86_64.rpm
```

An audit log will be created that indicates the success or failure of the update. If the log indicates failure, the file `/var/log/bivio/patch/intall.log` must be examined to determine the cause of the failure.

If the update is successful, the software release information in `/etc/NRDIST` will be updated to indicate the latest patch that was installed.

After installing the patch (or a set of patches), unless specified otherwise in the release notes for the patch, the processor should be rebooted using the command:

```
sudo -s shutdown -r now
```

20. CHECKING FOR CC COMPLIANCE

The Administrator can check whether the Bivio 6310-NC is configured to be CC compliant or not, by issuing the following command:

```
sudo bv_check_config
```

The output for a valid configuration will be:

```
Success – Configuration is CC compliant
```

The output for a non-compliant configuration will indicate failure, and the first non-compliant item that was encountered.

```
Failure – Configuration is not CC compliant. Reason: SSH tunnel
client not installed
```

Note that this test does not run through all the configuration items, so it cannot be depended on for 100% accuracy. Detailed configuration settings are not checked. Only major units are checked, such as whether a required item has been installed or activated.

21. SERVICES THAT MAY BE STARTED OR STOPPED BY THE ADMINISTRATOR

The administrator can start or stop the following services:

- The audit service (auditd)
- The syslog service (rsyslog)
- The NTP daemon (chronyd)
- The SSH login service (sshd)
- The SSH tunnel client (described previously)
- The TLS login service (described previously)

The SSH tunnel client and the TLS login service must be started and stopped as described previously. The other services may be started and stopped using the usual Linux method:

```
sudo service <service_name> [start | stop | status]
```

22. MANAGING THE BIVIOS APPLICATION

The procedures in [1] and [2] will properly configure the system to run the factory installed Bivio application named "`zbridgeapp_DPDK`". The procedures in [2] can be used to start, stop, and monitor it. A "cheat sheet" will be provided upon request if necessary, if detailed knowledge of the application processor is not desirable/essential for the purpose to which the product is applied.

The application can be started by the Administrator as follows:

```
sudo nrsp start zbridgeapp_DPDK
```

The application can be stopped by the Administrator as follows:

```
sudo nrsp stop zbridgeapp_DPDK
```

Other application related procedures are described in [2], as mentioned earlier.