Apple Inc.

Apple macOS Catalina 10.15 Common Criteria Configuration Guide

September 18, 2020
v1.7

Prepared By:
Acumen Security
2400 Research Blvd Suite 395
Rockville, MD, 20850
www.acumensecurity.net

Prepared for:
Apple Inc.
One Apple Park Way
MS 927-1CPS
Cupertino, CA-95014
www.apple.com

Table of Contents

Revision History

Version	Date	Description
0.1	April 8, 2020	Initial draft
1.7	September 18, 2020	Updated based on validator feedback.

1. Purpose of this document

This Common Criteria guidance document contains configuration information needed to configure and administer Apple macOS Catalina 10.15. The Apple macOS Catalina 10.15 conforms to the Protection Profile for General Purpose Operating Systems Version 4.2.1 (OS PP v4.2.1). The information contained in this document is intended for Administrators who would be responsible for the configuration and management of the Apple macOS Catalina 10.15.

1.1. TOE Overview

The Apple macOS Catalina 10.15 (herein referred to as the TOE) is a Unix-based graphical operating system that runs on Mac mini, MacBook Air, MacBook Pro, Mac Pro and iPad which include the T2 chip. macOS core is a POSIX compliant operating system built on top of the XNU kernel with standard Unix facilities available from the command line interface. It satisfies all the criterion to meet the Protection Profile for General Purpose Operating Systems Version 4.2.1.

1.2. TOE Description

The TOE includes the operating system macOS Catalina 10.15 (Build 19A583) and the security processor (T2) (SEPOS build 17P572). The T2 is a System on Chip (SoC) that acts as a security processor for the host platforms. The T2 includes several coprocessors, including the Secure Enclave Processor (SEP), that performs security-related functionality.

The TOE is evaluated on:

Device	Reference	Model	Family	Processor
iMac Pro	iMac Pro1,1	A1862	iMac Pro, Late 2017	Intel Xeon W-2140B, Intel Xeon W-2150B, Intel Xeon W-2170B, Intel Xeon W-2191B
Mac mini	Macmini8,1	A1993	2018	Intel Core i5-8500B, Intel Core i7-8700B
MacBook Air	MacBookAir8,1	A1932	Late 2018	Intel Core i5-8210Y
MacBook Air	MacBookAir8,2	A1932	2019	Intel Core i5-8210Y
Mac Pro	Mac Pro7,1	A1991	2019	Intel Xeon W-3223
Mac Pro	Mac Pro7,2	A1991	2019	Intel Xeon W-3235
Mac Pro	Mac Pro7,3	A1991	2019	Intel Xeon W-3245
Mac Pro	Mac Pro7,4	A1991	2019	Intel Xeon W-3265M
Mac Pro	Mac Pro7,5	A1991	2019	Intel Xeon W-3275M

MacBook Pro	MacBookPro15,2	A1989	Mid 2018, 13-inch (Touch Bar)	Intel Core i5-8279U, Intel Core i5-8259U
MacBook Pro	MacBookPro15,2	A1989	2019, 13-inch (Touch Bar)	Intel Core i5-8279U
MacBook Pro	MacBookPro15,4	A2159	2019 13-inch (Touch Bar, 2TB 3)	Intel Core i5-8257U
MacBook Pro	MacBookPro15,1	A1990	Mid 2018, 15-inch (Touch Bar)	Intel Core i7-8750H
MacBook Pro	MacBookPro15,1	A1990	2019, 15-inch (Touch Bar)	Intel Core i7-9750H
MacBook Pro	MacBookPro15,2	A1989	Mid 2018, 13-inch (Touch Bar)	Intel Core i7-8559U
MacBook Pro	MacBookPro15,2	A1989	2019, 13-inch (Touch Bar)	Intel Core i7-8569U
MacBook Pro	MacBookPro15,3	A1990	Mid 2018, 15-inch (Touch Bar)	Intel Core i7-8850H
MacBook Pro	MacBookPro15,4	A2159	2019 13-inch (Touch Bar, 2TB 3)	Intel Core i7-8557U
MacBook Pro	MacBookPro16,1	A2141	2019, 16-inch	Intel Core i7-9750H
MacBook Pro	MacBookPro15,1	A1990	Mid 2018, 15-inch (Touch Bar)	Intel Core i9-8950HK
MacBook Pro	MacBookPro15,1	A1990	2019, 15-inch (Touch Bar)	Intel Core i9-9880H, Intel Core I9-9980HK
MacBook Pro	MacBookPro15,3	A1990	Mid 2018, 15-inch (Touch Bar)	Intel Core i9-8950HK
MacBook Pro	MacBookPro15,3	A1990	2019, 15-inch (Touch Bar)	Intel Core i9-9880H
MacBook Pro	MacBookPro16,1	A2141	2019, 16-inch	Intel Core i9-9880H, Intel Core I9-9980HK
MacBook Air	MacBook Air9,1	A2179	2020	Intel Core I5-1030NG7 Intel Core I7-1060NG7

MacBook Pro	MacBook Pro16,3	A2289	2020, 13-inch	Intel Core I5-8257U, Intel Core I7-8557U,
MacBook Pro	MacBook Pro16,2	A2251	2020, 13-inch	Intel Core I5-1037NG7 Intel CoreI7-1068NG7

1.3. Assumptions

The following Assumptions are for the Operational Environment:

Assumptions	Operational Environment
A.PLATFORM	The OS relies upon a trustworthy computing platform for its execution. This underlying platform is out of scope of this PP.
A.PROPER_USER	The user of the OS is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy. At the same time, malicious software could act as the user, so requirements which confine malicious subjects are still in scope.
A.PROPER_ADMIN	The administrator of the OS is not careless, willfully negligent or hostile, and administers the OS within compliance of the applied enterprise security policy.

Table 1 Assumptions on the Operational Environment

1.4. TOE Delivery

The evaluated TOE is delivered as a pre-installed operating system on an Apple Mac with a T2 security coprocessor. All these Mac devices have the latest macOS version that is available for installation at the time of manufacturing. Digitally signed and verifiable updates to the TOE can be downloaded from: https://support.apple.com/downloads/macos. The evaluated TOE hardware models were shipped to Acumen Security CCTL located at 2400 Research Blvd, Suite #395, Rockville, MD 20850. The hardware models are securely kept at the CCTL location.

2. TOE Self Tests

The TOE is designed to perform all required self-tests without the need for any TOE configuration changes. If any of the self-tests fail, the TOE will immediately shutdown and prevent any attempted use of the unverified TOE.

2.1. Cryptographic Module Tests

FIPS 140-2 requires that all five cryptographic modules within the TOE perform the FIPS Power-On-Self-Tests (FIPSPOST) to ensure the integrity of the modules. HMAC-SHA-256 is used as an approved algorithm for the integrity test of the modules. This first step is invoked by the iBoot process upon power-up.

2.2. Cryptographic Algorithm Tests

FIPS 140-2 also requires that all five cryptographic modules perform self-tests to ensure the correctness of the cryptographic functionality upon power-up. In addition, the DRBG performs the required continuous health verification.

If the integrity and self-tests succeed for all five modules, the modules are made available to their calling services without reloading and the startup of the TOE continues. If any of the self-tests fail, the TOE will immediately shutdown.

The FIPSPOST of the TOE performs the following Cryptographic Algorithm Tests:

Algorithm	Modes	Test
Triple-DES	CBC	KAT (Known Answer Test), Separate encryption/decryption operations are performed.
AES implementations selected by the module for the corresponding environment AES-128	CBC, XTS, GCM	KAT
DRBG (CTR_DRBG and HMAC_DRBG; tested separately)	N/A	KAT
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512	N/A	KAT

ECDSA	Signature Generation, Signature Verification.	Pair-wise consistency test
RSA	Signature Generation, Signature Verification.	KAT

Table 2 Cryptographic algorithms

2.2. Software/Firmware Integrity Tests

All other TOE firmware and software integrity tests are performed using digital signature verification prior to any execution of the corresponding code.

When a Mac computer with the Apple T2 Security Chip is turned on, the chip executes code from read-only memory known as Boot ROM. The Boot ROM code contains the Apple Root CA public key, which is used to verify that the iBoot bootloader is signed by Apple's private key before allowing it to load. This immutable code, referred to as the hardware root of trust, is laid down during chip fabrication and is audited for vulnerabilities and implicitly trusted. This is the first step in the chain of trust. iBoot verifies the kernel and kernel extension code on the T2 chip, which subsequently verifies the Intel UEFI firmware. The UEFI firmware and the associated signature are initially available only to the T2 chip.

2.3. Critical Function Tests

No other critical function test is performed on power up.

3. Prerequisites for Installation

The User should ensure that compatible hardware is available before installing the TOE. The TOE cannot be installed on non-Apple products nor it can be installed as a virtual instance. The TOE can be installed on any one or all the above hardware platforms; refer Section 1.2 for a complete list of supported hardware platforms.

3.1. Disable Excluded Functionality

The following interfaces are not included as part of the evaluated configuration.

Functions	Exclusion discussion
Two-Factor Authentication	Two-factor authentication is an extra layer of security for an Apple ID used in the Apple store, iCloud and other Apple services. It is designed to enhance the security on these on-line Apple accounts. This feature is outside the scope of the evaluation.

Functions	Exclusion discussion
Bonjour	Bonjour is Apple's standards-based, zero configuration network protocol that lets devices find services on a network. This feature is outside the scope of the evaluation.
VPN Split Tunnel	VPN split tunnel is not included in the evaluation and must be disabled in the mobile device configurations meeting the requirements of this CC evaluation.
Siri Interface	The Siri interface supports some commands related to configuration settings. This feature is not included in the evaluation and must be disabled in the mobile device configurations that meet the requirements of this CC evaluation.

Table 3 Excluded functionality

The instructions below will enable the reader to disable the above functionalities:

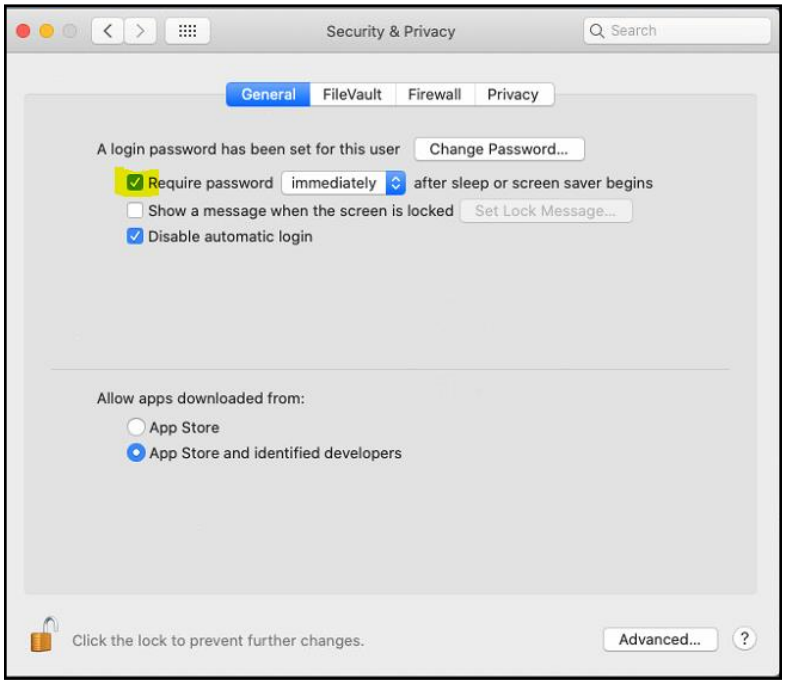
1. **Two-Factor Authentication**: If Two factor authentication is already enabled, it cannot be turned off. Latest versions of macOS require this extra level of security, which is designed to protect the user's information. However, if the user has recently updated their account to the latest version of macOS then they can unenroll within two weeks of enrollment. Follow the steps below to unenroll from Two factor authentication:
    a. Open enrollment confirmation email and click the link to return to your previous security settings.
    b. Note that turning off Two factor authentication makes the user's account less secure and the user cannot use features that require high security.
2. **Bonjour**: Bonjour is out of scope and is not an interfering service. No instructions are necessary.
3. **VPN Split Tunnel**:
    a. Open Apple Menu,
    b. Click on System Preferences,
    c. Click on Network -> <VPN-Identifier> -> Advanced and
    d. Check "Send all traffic over VPN connection".
4. **Siri Interface**:
    a. Open Apple Menu,
    b. Click on System Preferences and
    c. Click on Siri -> Uncheck "Enable Ask Siri".

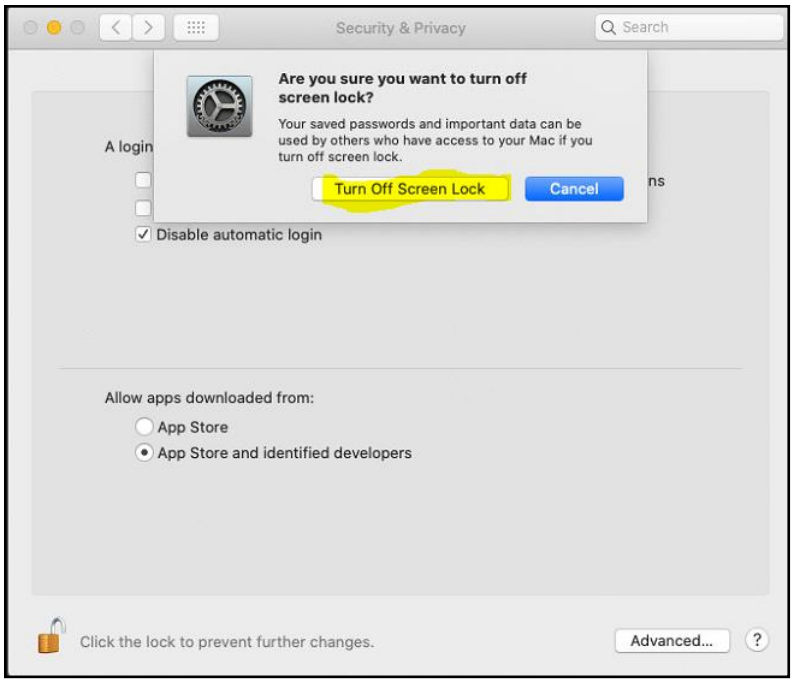
3.2. TOE Management Functions

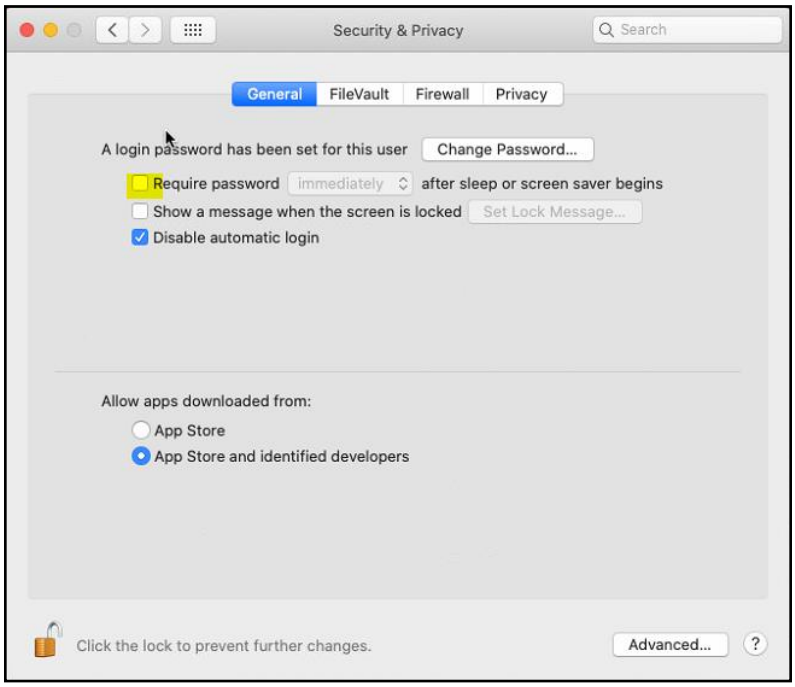
The TOE supports the following roles: Administrator and User. A user can Enable/disable screen lock, Configure screen lock inactivity timeout and Enable/disable Bluetooth interface.

The TOE administrator can configure management functions on the TOE as explained below:

1. Enable/Disable Screen-lock
    a. Open Apple Menu
    b. Click on System Preferences -> Security & Privacy,
    c. Click on General,
    d. Uncheck "Require password immediately after sleep or screen save changes".
    e. Click on "Turn Off Screen Lock".

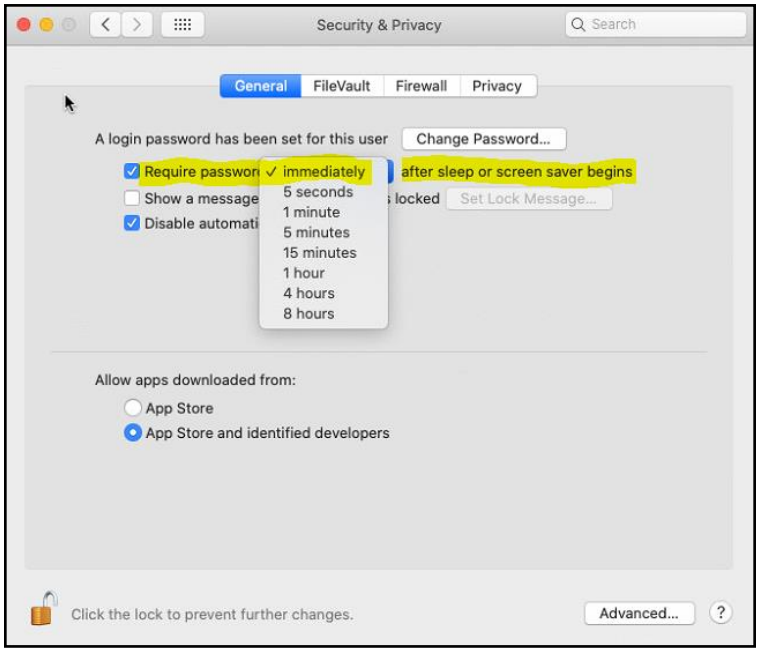






2. Configure screen-lock inactivity timeout
    a. Open Apple Menu
    b. Click on System Preferences -> Security & Privacy,
    c. Click on General,
    d. At the bottom left, click on “Click the lock to make changes”. The TOE will prompt the user to enter their administrator user account password.

e. Apart from “immediately”, the user can select different time values such as 5 seconds, 15 minutes or 8 hours (maximum). For a full list of supported time values, refer to the screenshot below.



3. Configure local audit storage capacity

a. Execute: sudo su

b. Navigate to /etc/security/

c. Execute: vi audit_control

d. Change the **filesz:10M**, for 10 MB file size.

e. Save the file and exit.

```
sh-3.2# nano audit_control
sh-3.2# cat audit_control
#
# $P4: //depot/projects/trustedbsd/openbsm/etc/audit_control#8 $
#
dir:/var/audit
flags:fr
minfree:5
naflags:fr
policy:cnt,argv
filesz:10M
expire-after:2G
superuser-set-sflags-mask:has_authenticated,has_console_access
superuser-clear-sflags-mask:has_authenticated,has_console_access
member-set-sflags-mask:
member-clear-sflags-mask:has_authenticated
sh-3.2#
```

f. Audit Log:

```
sh-3.2# auditreduce 20200827095123.20200827095356 | praudit -lx | grep audit_control
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:51:35 2020" msec=" + 781 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>audi
t_control</path><path>/private/etc/security/audit_control</path><attribute mode="100400" uid="root" gid="wheel" fsid="16777220" nodeid="1610810" device="0" /><subject a
udit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="1442" sid="621" tid="50546 192.168.254.19" /><return errval="success" retval="3" /><identity signe
r-type="1" signing-id="com.apple.cat" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xa1932f44abcf2d56e899db7f3e5f1931be34b7ca" /></record>
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:52:02 2020" msec=" + 306 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/pri
vate/etc/security/audit_control</path><path>/private/etc/security/audit_control</path><attribute mode="100400" uid="root" gid="wheel" fsid="16777220" nodeid="1610810" d
evice="0" /><subject audit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="1443" sid="621" tid="50546 192.168.254.19" /><return errval="success" retval
="3" /><identity signer-type="1" signing-id="com.apple.nano" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xbe96c0d33be7480a270dce14c6cb8c9337697
7c3" /></record>
```

4. Configure minimum password length

The TOE supports a password length of minimum 8 characters including Letters (Uppercase and Lowercase), Numbers and Special Symbols. Refer section “12 Password Policy” of this document for detailed instructions.

- Example:

```
admin@apple-icelake-i7-1060ng7 ~ % sudo su
Password:
sh-3.2# pwpolicy getaccountpolicies > temp.xml
sh-3.2# vi temp.xml
```

```
sh-3.2# pwpolicy setaccountpolicies temp.xml
Setting global account policies
```

```
Getting global account policies
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Prop
ertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>policyCategoryPasswordContent</key>
        <array>
                <dict>
                        <key>policyContent</key>
                        <string>policyAttributePassword matches '^$|.{8,}+</string>
                        <key>policyContentDescription</key>
                        <dict>
                                <key>ar</key>
```

- Audit Log:

```
<record version="11" event="close(2)" modifier="0" time="Wed Aug 26 14:29:23 2020" msec=" + 430 msec" ><argument arg-num="2" value="0x5" desc="fd" /><path>/private/etc/
master.passwd</path><attribute mode="100600" uid="root" gid="wheel" fsid="16777221" nodeid="1610954" device="0" /><subject audit-uid="admin" uid="root" gid="wheel" ruid
="root" rgid="wheel" pid="6972" sid="5697" tid="14590 192.168.254.93" /><return errval="success" retval="0" /><identity signer-type="1" signing-id="com.apple.praudit" s
igning-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xe4bc4db7975a8074a03ae4f19d4548831b42270b" /></record>
```

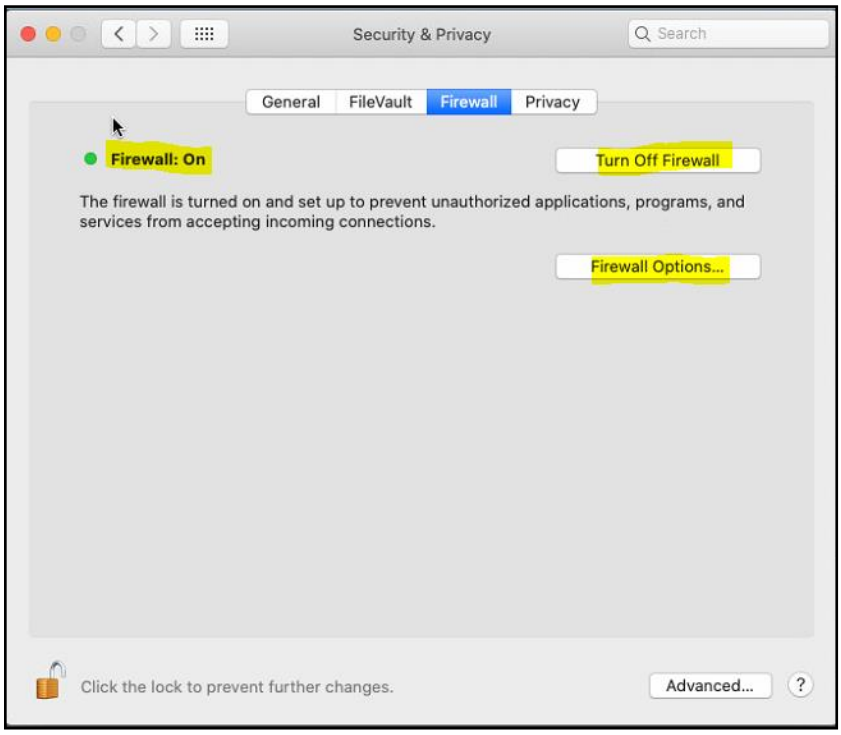
5. Configure lockout policy for unsuccessful authentication attempts through timeout between attempts
    a. Refer section “12 Password Policy” of this document for detailed instructions.
    b. Example:

```
sh-3.2# pwpolicy -u user1 -setpolicy "maxFailedLoginAttempts=3"
Setting policy for user1
sh-3.2#
```

    c. Audit Log:

<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 06:39:09 2020" msec=" + 635 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/private/var/db/dslocal/nodes/Default/users/user1.plist</path><path>/private/var/db/dslocal/nodes/Default/users/user1.plist</path><attribute mode="100600" uid="root" gid="wheel" fsid="16777220" nodeid="1795213" device="0" /><subject audit-uid="-1" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="129" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="6" /><identity signer-type="1" signing-id="com.apple.opendirectoryd" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xf3b54985df56cc3462de45d4cf1b7b30fd1a2a89" /></record>
<record version="11" event="open(2) - read,write,creat" modifier="0" time="Thu Aug 27 06:39:12 2020" msec=" + 114 msec" ><argument arg-num="3" value="0x180" desc="mode" /><argument arg-num="2" value="0xa02" desc="flags" /><path>/private/var/db/dslocal/nodes/Default/users/.tmp.user1.plist</path><subject audit-uid="-1" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="129" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="6" /><identity signer-type="1" signing-id="com.apple.opendirectoryd" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xf3b54985df56cc3462de45d4cf1b7b30fd1a2a89" /></record>

6. Configure host-based firewall
    a. Open Apple Menu,
    b. Click on System Preferences -> Security & Privacy
    c. At the bottom left, click on “Click the lock to make changes”. The TOE will prompt the user to enter their administrator user account password.
    d. Click on “Turn on Firewall” or “Turn Off Firewall”.
    e. When the firewall is On, click on “Firewall Options…” for additional firewall configuration options.



    f. Audit Log:

<record version="11" event="close(2)" modifier="0" time="Wed Aug 26 13:01:28 2020" msec=" + 539 msec" ><argument arg-num="2" value="0x9" desc="fd" /><path>/usr/libexec/ApplicationFirewall/socketfilterfw</path><attribute mode="100755" uid="root" gid="wheel" fsid="16777221" nodeid="520115" device="0" /><subject audit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="168" sid="100040" tid="50331650 0.0.0.0" /><return errval="success" retval="0" /><identity signer-type="1" signing-id="com.apple.authd" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0x714f6005ca8309dee510bfe85b908174be6bf7a2" /></record>

7. Configure name/address of audit/logging server to which to send audit/logging records
    - **Prerequisites:**
        - A user account with administrator privileges.
        - A Syslog server running and configured to receive the audit events.

- To redirect logs in macOS to a syslog server, it is necessary to edit the file syslog.conf located in:

  /etc/syslog.conf

- Then add the following line (replace it with the IP of your syslog server):

  . @syslog_server_ip

- To customize the port number (e.g. 514) use @syslog_server_ip:514 instead.
- This change should be instant, if not the syslog service has to be restarted typing the following command in Terminal app:
- launchctl start com.apple.syslogd
- To verify that the configuration was successful, perform a privilege elevation, by typing "sudo su" in the Terminal app:

```
cert@mac-mini-i5-8500B ~ % sudo su
Password:
sh-3.2#
```

- The audit record should be sent and received by the remote audit server:

```
2020-08-27T14:57:15-04:00 mac-mini-i5-8500B sudo[3075]: getgrouplist_2 called triggering group enumeration
```

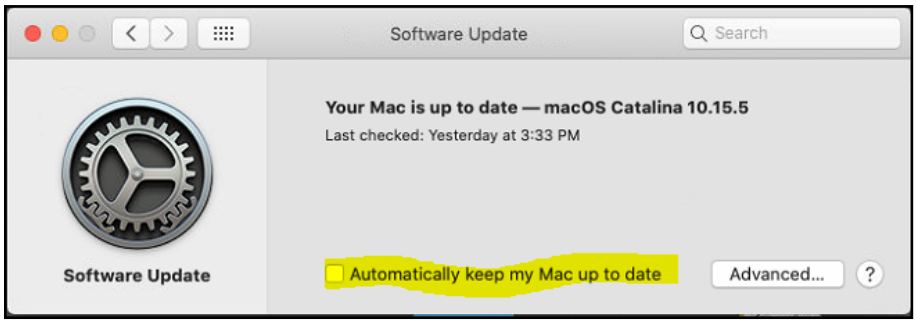
8. Configure audit rules
   a. Refer section "14 Auditing" for detailed instructions.
   b. Example:

```
sh-3.2# nano audit_control
sh-3.2# cat audit_control
#
# $P4: //depot/projects/trustedbsd/openbsm/etc/audit_control#8 $
#
dir:/var/audit
flags:fr
minfree:5
naflags:fr
policy:cnt,argv
filesz:10M
expire-after:2G
superuser-set-sflags-mask:has_authenticated,has_console_access
superuser-clear-sflags-mask:has_authenticated,has_console_access
member-set-sflags-mask:
member-clear-sflags-mask:has_authenticated
sh-3.2#
```

   c. Audit Log:

```
sh-3.2# auditreduce 20200827095123.20200827095356 | praudit -lx | grep audit_control
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:51:35 2020" msec=" + 781 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>audi
t_control</path><path>/private/etc/security/audit_control</path><attribute mode="100400" uid="root" gid="wheel" fsid="16777220" nodeid="1610810" device="0" /><subject a
udit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="1442" sid="621" tid="50546 192.168.254.19" /><return errval="success" retval="3" /><identity signe
r-type="1" signing-id="com.apple.cat" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xa1932f44abcf2d56e899db7f3e5f1931be34b7ca" /></record>
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:52:02 2020" msec=" + 306 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/pri
vate/etc/security/audit_control</path><path>/private/etc/security/audit_control</path><attribute mode="100400" uid="root" gid="wheel" fsid="16777220" nodeid="1610810" d
evice="0" /><subject audit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="1443" sid="621" tid="50546 192.168.254.19" /><return errval="success" retval
="3" /><identity signer-type="1" signing-id="com.apple.nano" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xbe96c0d33be7480a270dce14c6cb8c93376977
7c3" /></record>
```

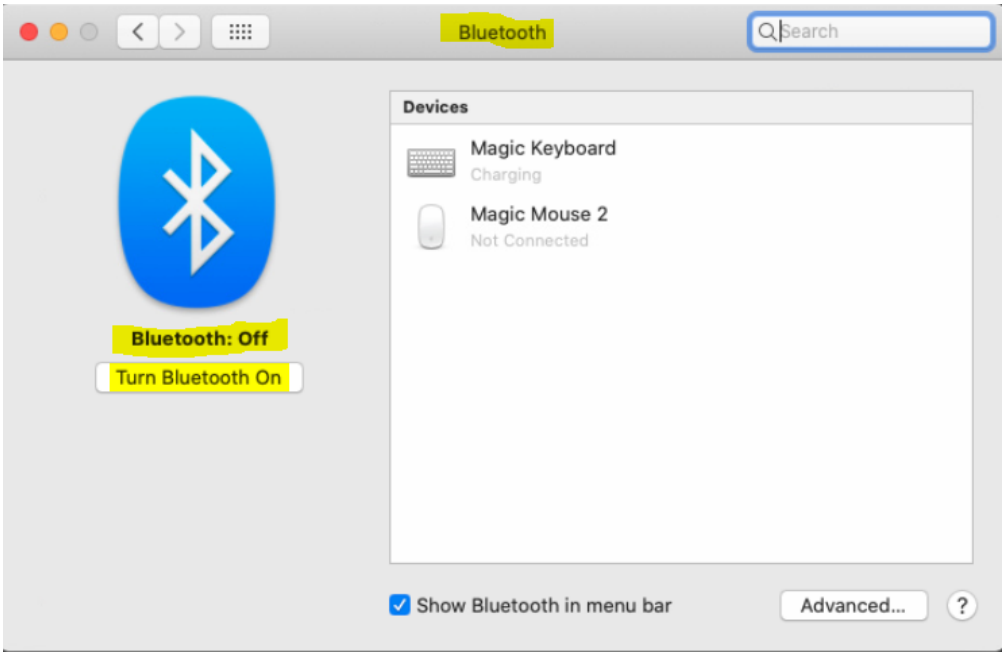
9. Enable/Disable automatic software update
    a. Open Apple Menu,
    b. Click on System Preferences,
    c. Click on Software Update,
    d. Uncheck “Automatically keep my Mac up to date”.



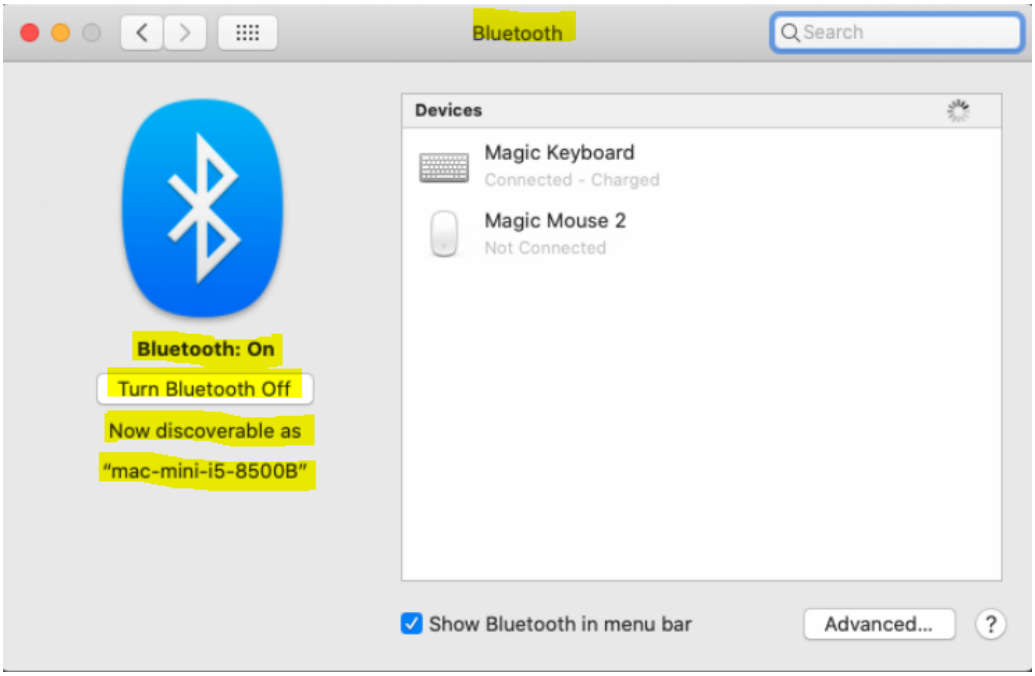
    e. Audit Log:


```
<record version="11" event="open(2) - read,write" modifier="0" time="Thu Aug 27 03:51:11 2020" msec=" + 547 msec" ><argument arg-num="2" value="0x2" desc="flags" /><path>/dev/dtracehelper</path><path>/dev/dtracehelper</path><attribute mode="20666" uid="root" gid="wheel" fsid="1709924350" nodeid="591" device="419430400" /><subject audit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="783" sid="621" tid="50546 192.168.254.19" /><return errval="success" retval="3" /><identity signer-type="1" signing-id="com.apple.softwareupdate" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0x31d8ff2a3c03648b1283114c4859a86bd7fddf84" /></record>
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 03:51:11 2020" msec=" + 557 msec" ><argument arg-num="2" value="0x1100004" desc="flags" /><path>/usr/sbin</path><path>/usr/sbin</path><attribute mode="40755" uid="root" gid="wheel" fsid="16777220" nodeid="5320" device="0" /><subject audit-uid="-1" uid="_softwareupdate" gid="_softwareupdate" ruid="_softwareupdate" rgid="_softwareupdate" pid="580" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="5" /><identity signer-type="1" signing-id="com.apple.softwareupdated" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0x8c08583c37245cd270a1eb07cf401cd70487ea7c" /></record>
```


10. Configure Bluetooth
    a. Open System Preferences,
    b. Click on Bluetooth



c. If the Bluetooth is Off (as shown in the screenshot above), then click on “Turn Bluetooth On” to enable Bluetooth.

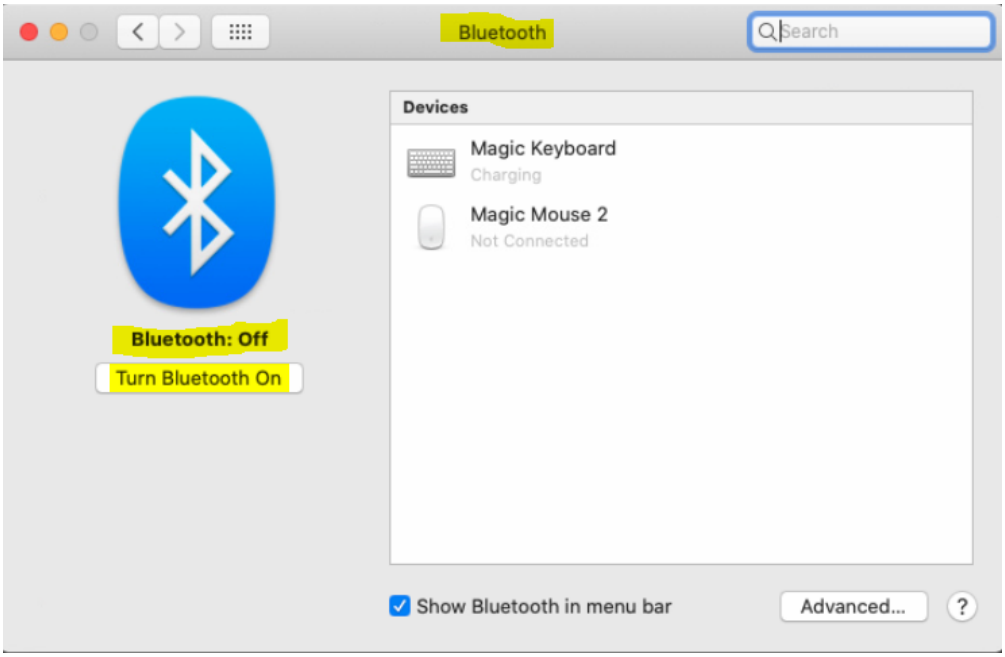


d. Audit Log:


```
<record version="11" event="open(2) - read" modifier="0" time="Wed Sep 16 12:17:48 2020" msec=" + 535 msec" ><argument arg-num="2" value="0x0" desc="
flags" /><path>/Library/Preferences/com.apple.Bluetooth.plist</path><path>/Library/Preferences/com.apple.Bluetooth.plist</path><attribute mode="10064
4" uid="root" gid="wheel" fsid="16777220" nodeid="2209663" device="0" /><subject audit-uid="-1" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="
162" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="3" /><identity signer-type="1" signing-id="com.apple.cfprefsd" signing-id-trunca
ted="no" team-id="" team-id-truncated="no" cdhash="0x4f246c6b26707431546ad165cf49b7b0267224a9" /></record>
```


e. Once the Bluetooth is enabled, the TOE will display its’ computer name under “Now discoverable as”. This means that the TOE is ready to connect or pair with other Bluetooth enabled devices.

f. To disable Bluetooth, click on “Turn Bluetooth Off”.



g. Audit Log:

<record version="11" event="open(2) - read" modifier="0" time="Wed Sep 16 12:53:45 2020" msec=" + 140 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/Library/Preferences/com.apple.Bluetooth.plist</path><path>/Library/Preferences/com.apple.Bluetooth.plist</path><attribute mode="100644" uid="root" gid="wheel" fsid="16777220" nodeid="2210166" device="0" /><subject audit-uid="-1" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="162" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="3" /><identity signer-type="1" signing-id="com.apple.cfprefsd" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0x4f246c6b26707431546ad165cf49b7b0267224a9" /></record>

11. Network Time Server

a. Execute the commands below:

i. /usr/sbin/systemsetup -setnetworktimeserver “time.euro.apple.com”

ii. /usr/sbin/systemsetup -setusingnetworktime on

```
sh-3.2# /usr/sbin/systemsetup -setnetworktimeserver "time.euro.apple.com"
setNetworkTimeServer: time.euro.apple.com
sh-3.2# /usr/sbin/systemsetup -setusingnetworktime on
Network Time is already on.
sh-3.2#
```

b. Audit Log:

sh-3.2# auditreduce 20200827093436.20200827093652 | praudit -lx | grep ntp
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:36:00 2020" msec=" + 813 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/etc/ntp.conf</path><path>/private/etc/ntp.conf</path><attribute mode="100644" uid="root" gid="wheel" fsid="16777220" nodeid="356936" device="0" /><subject audit-uid="-1" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="1064" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="3" /><identity signer-type="1" signing-id="com.apple.systemadministration.writeconfig" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0x4ce3e6d4e65e153214bd6419a2d1ec32b7923a7d" /></record>
<record version="11" event="open(2) - read" modifier="0" time="Thu Aug 27 05:36:00 2020" msec=" + 943 msec" ><argument arg-num="2" value="0x0" desc="flags" /><path>/etc/ntp.conf</path><path>/private/etc/ntp.conf</path><attribute mode="100644" uid="root" gid="wheel" fsid="16777220" nodeid="356936" device="0" /><subject audit-uid="-1" uid="_timed" gid="_timed" ruid="_timed" rgid="_timed" pid="132" sid="100000" tid="0 0.0.0.0" /><return errval="success" retval="7" /><identity signer-type="1" signing-id="com.apple.timed" signing-id-truncated="no" team-id="" team-id-truncated="no" cdhash="0xa280266e5473cd117ef6a929f66421f98a2fa53d" /></record>

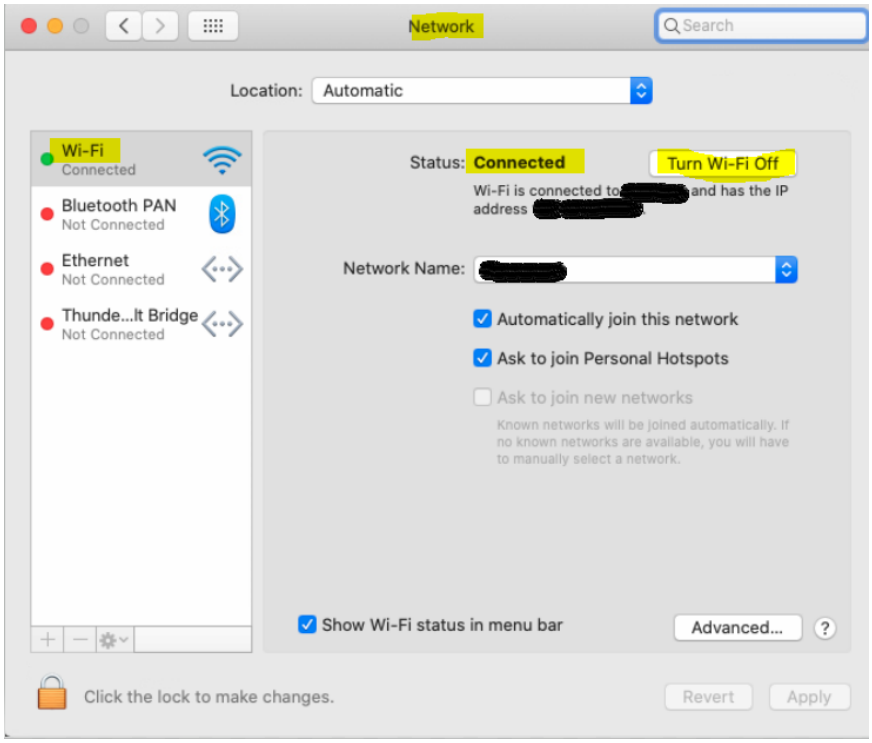
12. Configure Wi-Fi Interface

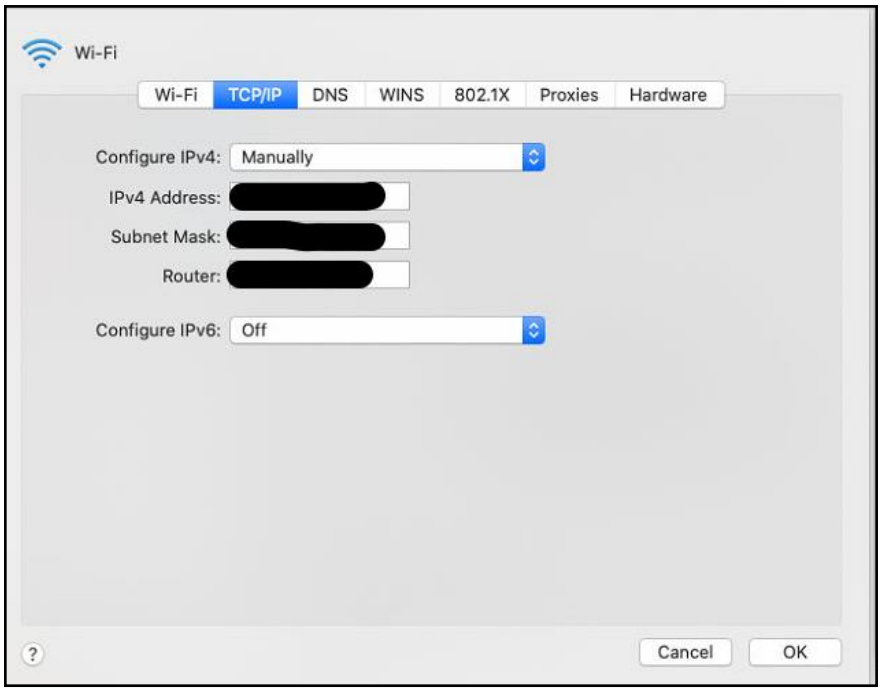
a. Open Apple Menu,

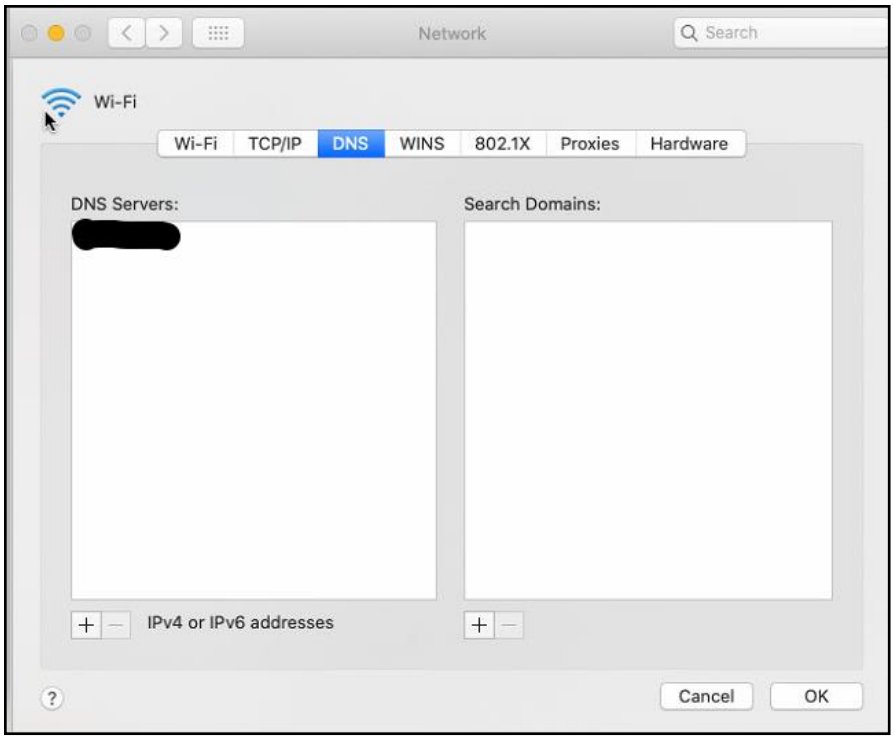
b. Click on System Preferences -> Network -> Wi-Fi

c. At the bottom left, click on “Click the lock to make changes”. The TOE will prompt the user to enter their administrator user account password.

d. Click on Advanced symbol to manually configure different Wi-Fi settings such as TCP/IP, DNS etc. Note some information is intentionally blurred in the screenshots below to protect confidential information.







e. Audit Log:

```
<record version="11" event="execve(2)" modifier="0" time="Wed Sep 16 11:49:31 2020" msec=" + 523 msec" ><exec_args><arg>auditreduce</arg><arg>-e</arg
><arg>Testuser</arg><arg>20200916154634.not_terminated</arg></exec_args><path>/usr/sbin/auditreduce</path><path>/usr/sbin/auditreduce</path><attribut
e mode="100755" uid="root" gid="wheel" fsid="16777220" nodeid="521084" device="0" /><arbitrary print="hex" type="1" count="20" > 6a cf b5 f9 80 9a a6
 f2 c8 34 c0 f 1c fe 8a c1 53 5c 3e e8</arbitrary><subject audit-uid="admin" uid="root" gid="wheel" ruid="root" rgid="wheel" pid="48893" sid="48607"
tid="7995 192.168.254.93" /><return errval="success" retval="0" /><identity signer-type="1" signing-id="com.apple.bash" signing-id-truncated="no" tea
m-id="" team-id-truncated="no" cdhash="0x508595e78370793873b546fdc6ed6b32422627eb" /></record>
```

4. Installation of the Apple macOS Catalina 10.15

The TOE comes pre-installed on the supporting hardware platforms as mentioned above. The TOE has a built-in update feature that the user can leverage. Should the need arise, the user can manually download and re-install and/or update the TOE on the supporting hardware.

It is recommended to take a back-up of the TOE before updating the TOE.

Updating the TOE using built-in feature:

- Start System & Preferences.
- Click on Software Update.
- If there is a software update to the TOE, then it will be displayed here.
- Click on Update Now to update the TOE.

The TOE implements anti-rollback feature that prevents the user from downgrading the TOE software version to an earlier version. This feature helps avoid rollback attacks.

4.1. Clean Installation steps for the TOE

Before installing/re-installing the TOE, the user should backup the TOE to a media (e.g. hard drive) and keep the media in a safe and secure location such as a locker/safe. The TOE can be installed in different ways as described below.

4.1.1. Install the TOE from Apple website

- Download macOS Catalina from https://support.apple.com/downloads/macos
- After downloading, double click on macOSUpdXXX.dmg file, where XXX = actual macOS Catalina version.
- Before updating, the TOE will verify the downloaded file via digital signature verification.
- If the digital signature verification is successful, the TOE will proceed to install the update. Occasional TOE reboot is normal during the installation.
- If the digital signature verification is unsuccessful, the TOE will not proceed with the update/installation process.
- The TOE will reboot after the update is successfully installed.

4.1.2. Reinstall the TOE

- On the TOE, choose Apple menu > Restart.
- Immediately after the TOE reboots, do one of the following:
    - Install the latest version of the TOE from the Internet: Press and hold Option-Command-R until a spinning globe appears, then release the keys. This option installs the latest version of the TOE that is compatible with the user's computer.
    - Reinstall your computer's original version of macOS from the Internet: Press and hold Shift-Option-Command-R until a spinning globe appears, then release the keys. This option installs the most recent version of the TOE that came up with the user's computer, including any available updates to that version.
    - Reinstall the TOE from the built-in recovery disk on your computer: Press and hold Command-R until the Utilities window appears. This option reinstalls the version of the TOE stored on the user's computer built-in recovery disk, including any updates that the user installed.
- Select Reinstall macOS, then click continue.
- Follow the onscreen instructions. In the pane where you select a disk, select your current macOS disk (in most cases, it's the only one available).
- For additional information and support about reinstalling the TOE refer macOS User Guide.

5. Check Software Updates

The TOE operating system (OS) updates, and software application updates can be downloaded manually from Apple website. The installation of authentic OS updates and software application updates is covered in Section 6.

Note: By default, the TOE uses TLS v1.3 to establish a secure channel with Apple website and hence the TOE had to be configured to use a Proxy that limited the TLS version to TLS v1.2. Therefore, the user must configure a Proxy instance on the TOE that would restrict the TLS version to TLS v1.2. An example of Proxy configuration is provided in Appendix A.

6. Installing Updates

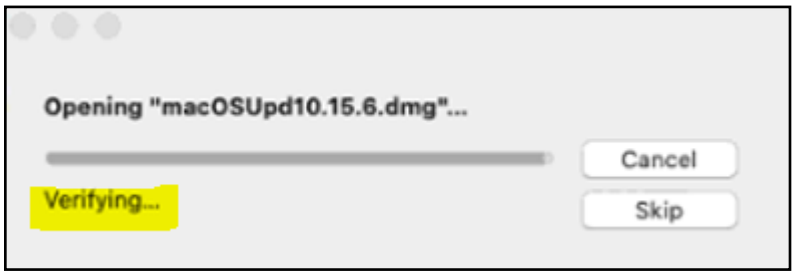
Authentic OS and software application updates can be downloaded from https://support.apple.com/downloads. Once an update(s) is downloaded, the user can initiate the installation of that update in the following manner:

- Download the appropriate update from https://support.apple.com/downloads according to the user requirement(s).
- Double click on the downloaded update.
- The TOE verifies the integrity of the software update by performing an RSA 2048-bit digital signature verification.
- After the digital signature verification is successful the TOE will install the update.
- If the digital signature verification fails, the TOE will warn the user that the digital signature verification failed and will not install the update. The TOE then terminates the update process.

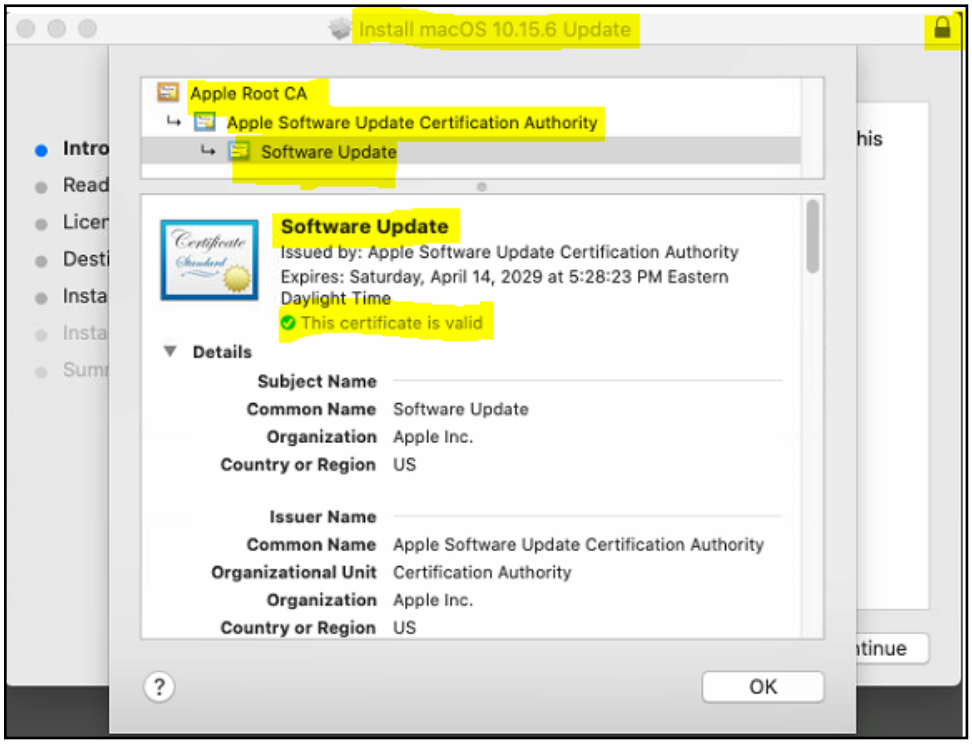
Note: For OS updates, the TOE may occasionally reboot itself during the update process. This behavior is not uncommon. Software Application updates may or may not require the TOE to reboot.

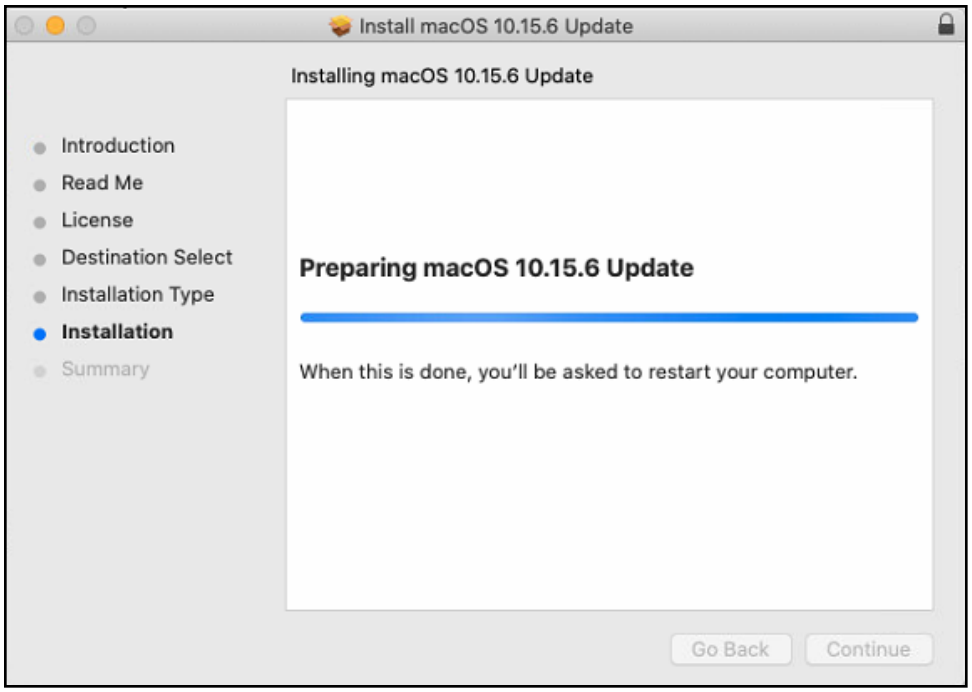
6.1. Installing OS updates

- Download the appropriate OS update from https://support.apple.com/downloads according to the user requirements.
- In this case, "macOSUpd10.15.6.dmg" was downloaded and installed. Before installing the update, the TOE performs a digital signature verification.

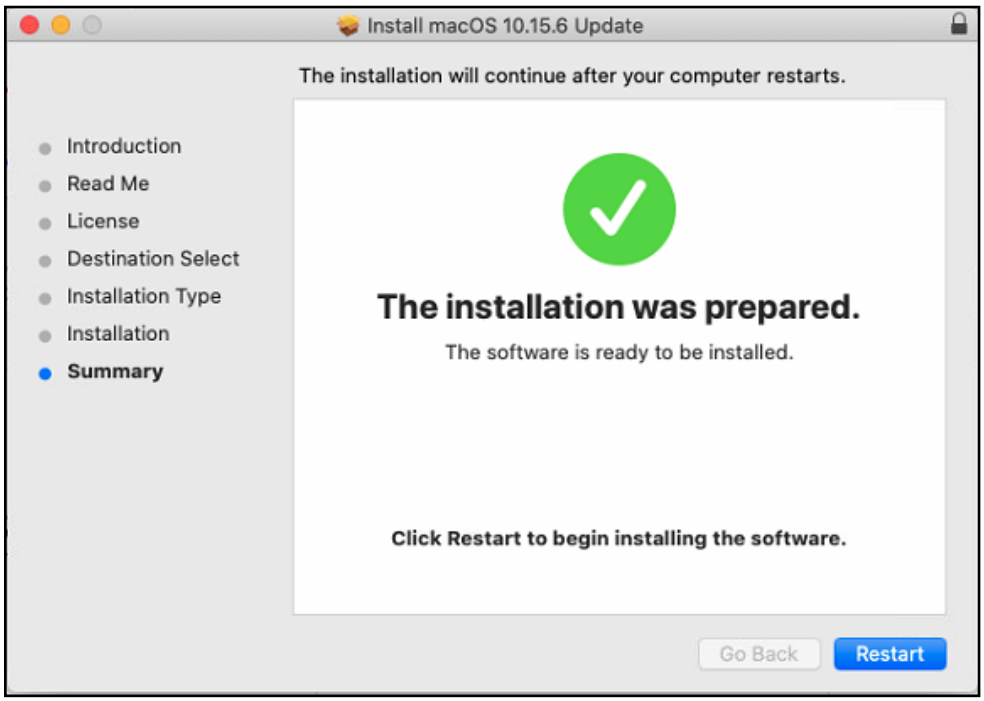


- After successful digital signature verification, the TOE will proceed with the installation. Follow onscreen instructions.
- Additionally, the user can click on the "lock icon" in the upper right-hand corner of the installer window to verify the digital signature as shown below. This approach can be used to verify digital signature on any installer.

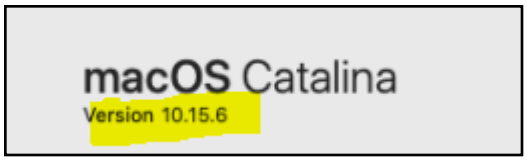




- After the TOE prepares the OS update, the TOE will prompt the user to install the OS update. If the user does not want to install the update, then the user can simply close the window and the TOE will not be updated.
- Click on Restart to start the installation process. The TOE will reboot once the update is successfully installed.



The user can verify that the updated version of the TOE by navigating to "Apple Symbol" -> "About This Mac" -> "Overview"

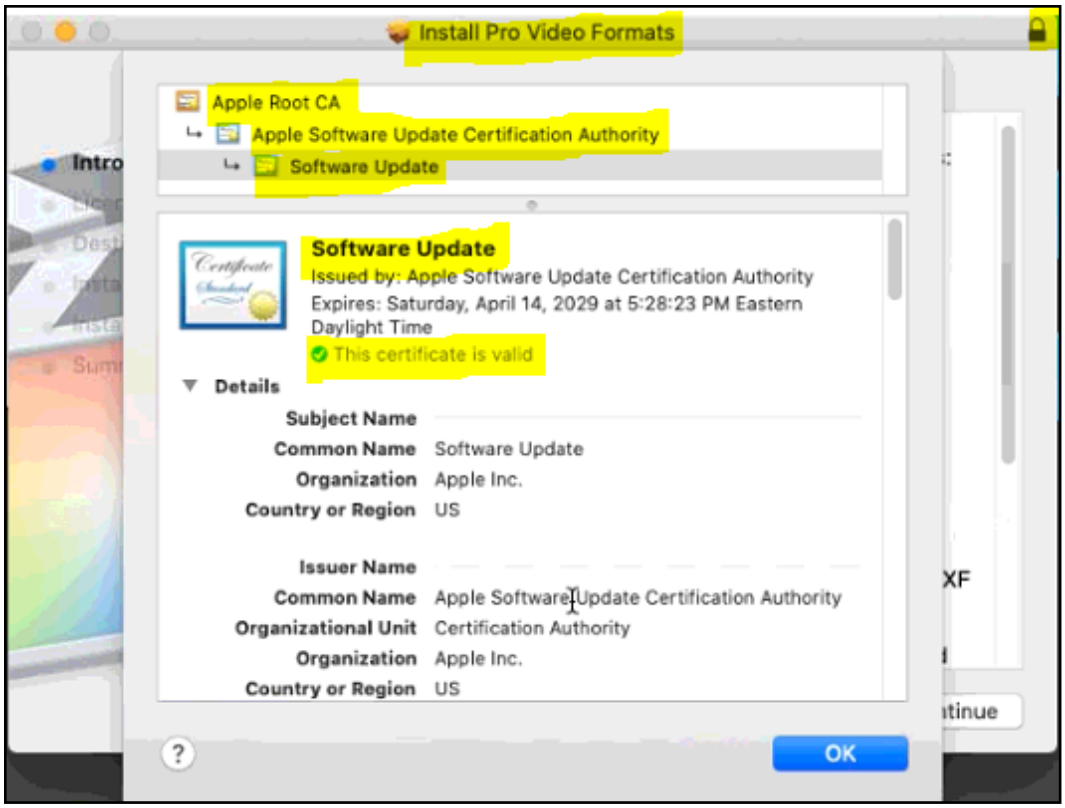


6.2. Installing Software Application Updates

The software application updates can be installed on the TOE in a similar way as that of OS updates.

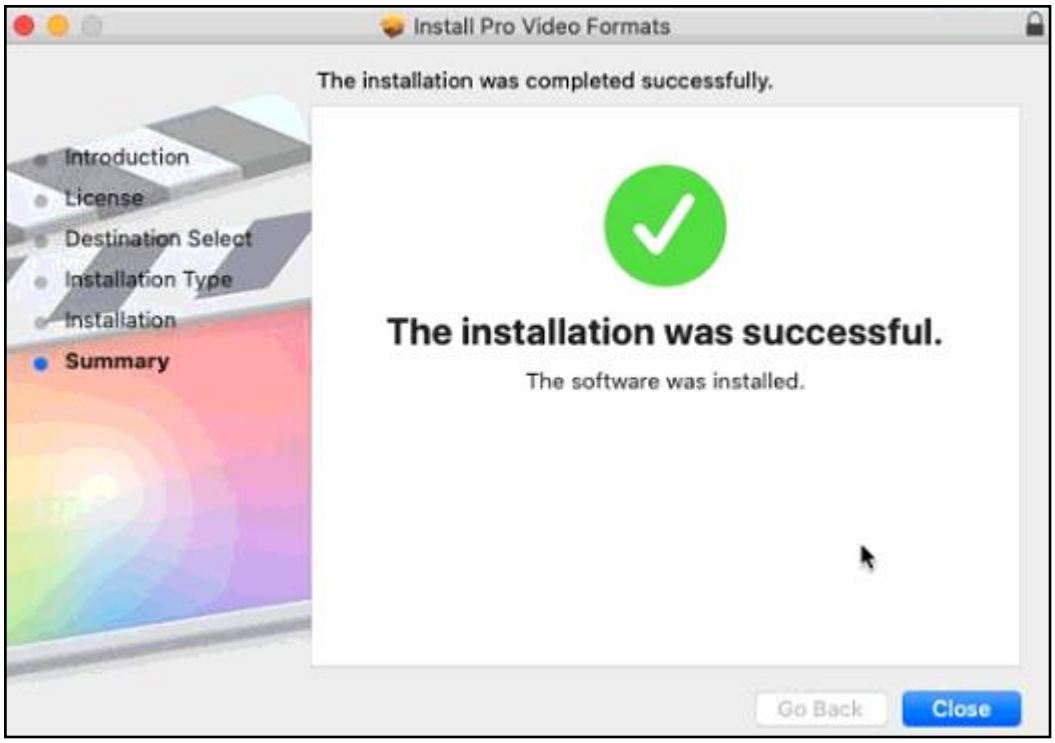
- Download the appropriate Software Application update from https://support.apple.com/downloads according to the user requirements.
- In this case, "ProVideoFormats.dmg" was downloaded and installed. Before installing the update, the TOE performs a digital signature verification.

- After successful digital signature verification, the TOE will proceed with the installation. Follow onscreen instructions.
- Additionally, the user can click on the "lock icon" in the upper right-hand corner of the installer window to verify the digital signature as shown below. This approach can be used to verify digital signature on any installer.



- The TOE will request the user to enter admin credentials to proceed with the update.

- Click on Install Software to begin with the installation process. Follow onscreen instructions.



- Click on Close to close the window.

7. TOE Startup Security Utility

Startup Security Utility is a replacement to the previous Firmware Password Utility. On Mac computers with an Apple T2 Security Chip, it handles a larger set of security policy settings. The utility is accessible by booting into recovery OS and selecting Startup Security Utility from the Utilities menu. The advantage of putting critical system security policy controls (such as secure boot or SIP) in the recovery OS is that the entire OS is integrity checked. This ensures that any attacker code that has broken into the Mac can't trivially impersonate the user for purposes of further disabling security policies.

Critical policy changes now require authentication, even in recovery mode. This feature is available only on Mac computers containing the T2 chip. When Startup Security Utility is first opened, it prompts the user to enter an administrator password from the primary macOS installation associated with the currently booted macOS Recovery. If no administrator exists, one must be created before the policy can be changed. The T2 chip requires that the Mac computer is currently booted into macOS Recovery and that an authentication with a Secure Enclave backed credential has occurred before such a policy change can be made. Security policy changes have two implicit requirements. macOS Recovery must:

- Be booted from a storage device directly connected to the T2 chip, because partitions on other devices don't have Secure Enclave backed credentials bound to the internal storage device.
- Reside on an APFS-based volume, because there is support only for storing the Authentication in Recovery credentials sent to the Secure Enclave on the "Pre-boot" APFS volume of a drive. HFS plus-formatted volumes can't use secure boot.

This policy is only shown in Startup Security Utility on Mac computers with an Apple T2 Security Chip. Although most use cases shouldn't require changes to the secure boot policy, administrators are ultimately in control of their device's settings, and may choose, depending on their needs, to disable or downgrade the secure boot functionality on their Mac.

Secure boot policy changes made from within this app apply only to the evaluation of the chain of trust being verified on the Intel processor. The option "Secure boot the T2 chip" is always in effect.

The secure boot policy can be configured to one of three settings: Full Security, Medium Security, and No Security. No Security completely disables secure boot evaluation on the Intel processor and allows the user to boot from any supported and allowed boot media.

7.1. Mac startup key combinations:

To use any of these key combinations, press and hold the keys immediately after pressing the power button to turn on your Mac, or after your Mac begins to restart. Keep holding until the described behavior occurs.

- **Command (⌘)-R:** Start up from the built-in macOS Recovery system. Or use Option-Command-R or Shift-Option-Command-R to start up from macOS Recovery over the Internet. macOS Recovery installs different versions of macOS, depending on the key combination you use while starting up. If your Mac is using a firmware password, you're prompted to enter the password.
- **Option (⌥) or Alt:** Start up to Startup Manager, which allows you to choose other available startup disks or volumes. If your Mac is using a firmware password, you're prompted to enter the password.
- **Option-Command-P-R:** Reset NVRAM or PRAM. If your Mac is using a firmware password, it ignores this key combination or starts up from macOS Recovery.
- **Shift (⇧):** Start up in safe mode. Disabled when using a firmware password.
- **D:** Start up to the Apple Diagnostics utility. Or use Option-D to start up to this utility over the Internet. Disabled when using a firmware password.
- **N:** Start up from a NetBoot server, if your Mac supports network startup volumes. To use the default boot image on the server, hold down Option-N instead. Disabled when using a firmware password.
- **Command-S:** Start up in single-user mode. Disabled in macOS Mojave or later, or when using a firmware password.
- **T:** Start up in target disk mode. Disabled when using a firmware password.
- **Command-V:** Start up in verbose mode. Disabled when using a firmware password.
- **Eject** (⏏) or **F12** or **mouse button** or **trackpad button:** Eject removable media, such as an optical disc. Disabled when using a firmware password.

7.1.1. If a key combination doesn't work

If a key combination doesn't work at startup, one of these solutions might help:

- Be sure to press and hold all keys in the combination together, not one at a time.
- Shut down your Mac. Then press the power button to turn on your Mac. Then press and hold the keys as your Mac starts up.
- Wait a few seconds before pressing the keys, to give your Mac more time to recognize the keyboard as it starts up. Some keyboards have a light that flashes briefly at startup, indicating that the keyboard is recognized and ready for use.
- If you're using a wireless keyboard, plug it into your Mac, if possible. Or use your built-in keyboard or a wired keyboard. If you're using a keyboard made for a PC, such as a keyboard with a Windows logo, try a keyboard made for Mac.

- If you're using Boot Camp to start up from Microsoft Windows, set Startup Disk preferences to start up from macOS instead. Then shut down or restart and try again.

Remember that some key combinations are disabled when your Mac is using a firmware password.

8. Configuring TLS

The TOE implements TLS v1.2 (RFC 5246). The following cipher suites are supported by the OS for TLS session establishments. No configuration is needed from the user.

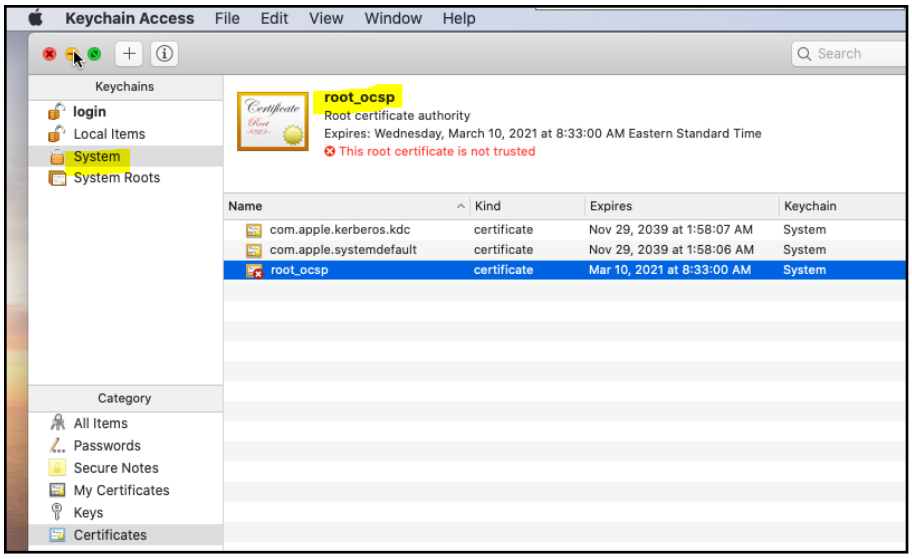
- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246
- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

The TOE supports the following NIST Elliptic Curves in the Client Hello. No configuration is needed from the user.

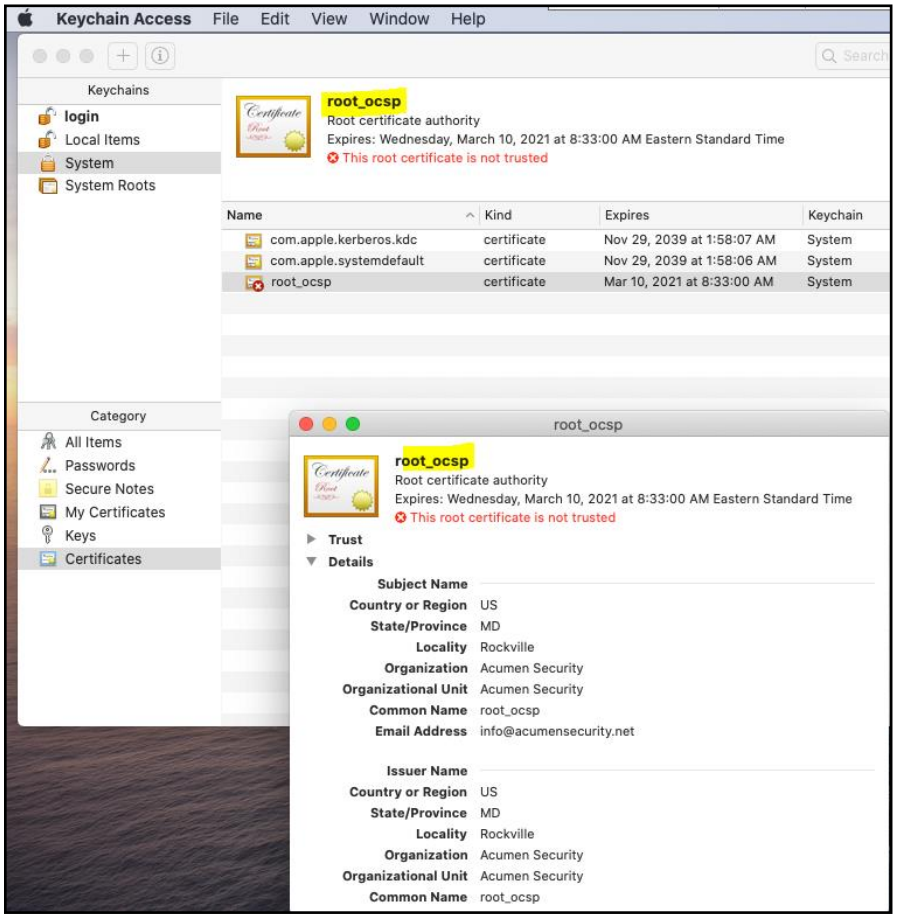
- Secp256r1
- Secp384r1
- Secp521r1

The TOE comes pre-installed with Certificate Authorities (CA) in the Trust Anchor Database to establish a chain of trust. Additionally, the TOE allows the user to manually import, install and trust custom CAs. This can be achieved by installing the CA certificate to the TOE *System* keychain. The screenshots below show how to install and trust a custom CA certificate on the TOE.

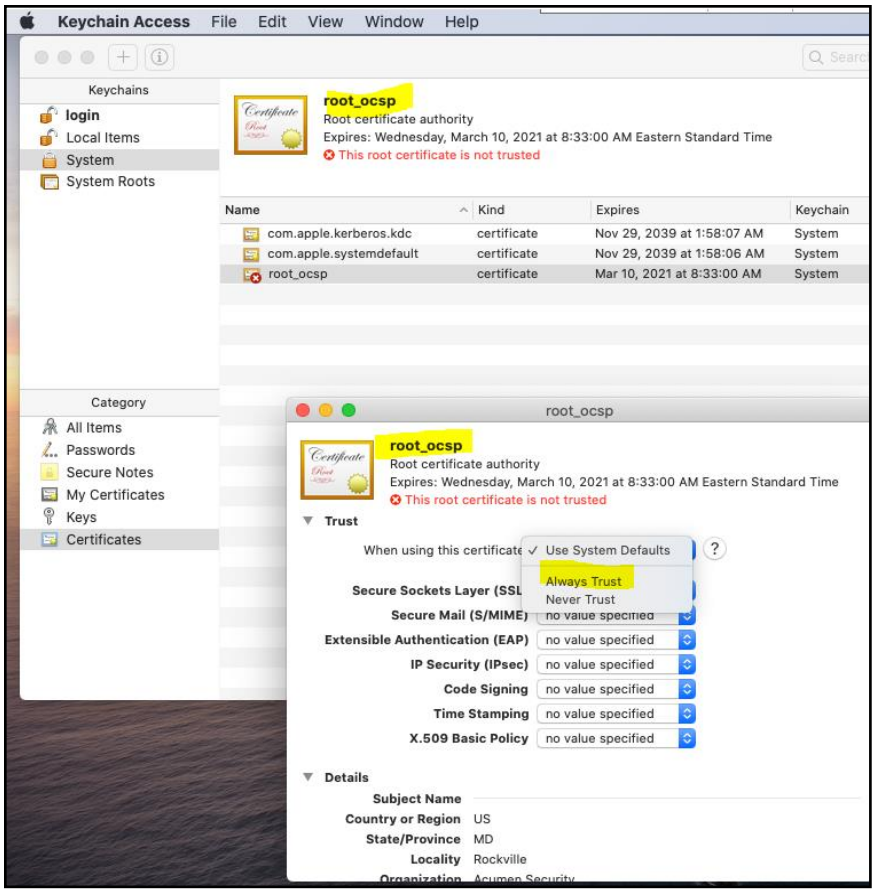
- Double click on the CA certificate that needs to be installed on the TOE. In this case, root_ocsp
- The TOE will automatically open the built-in Keychain Access application as shown below.

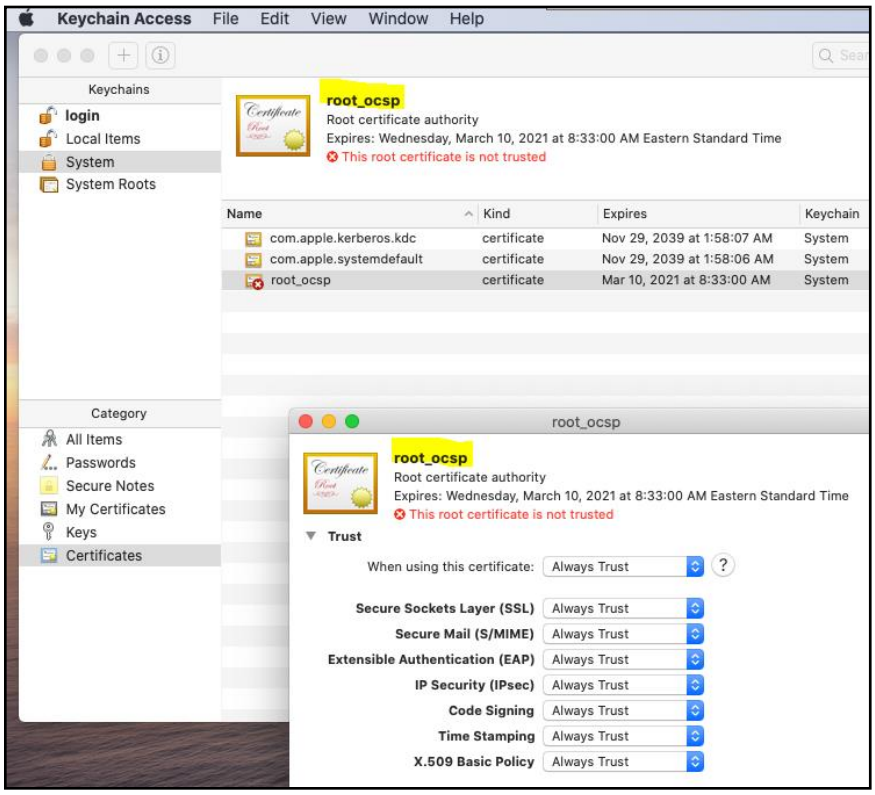


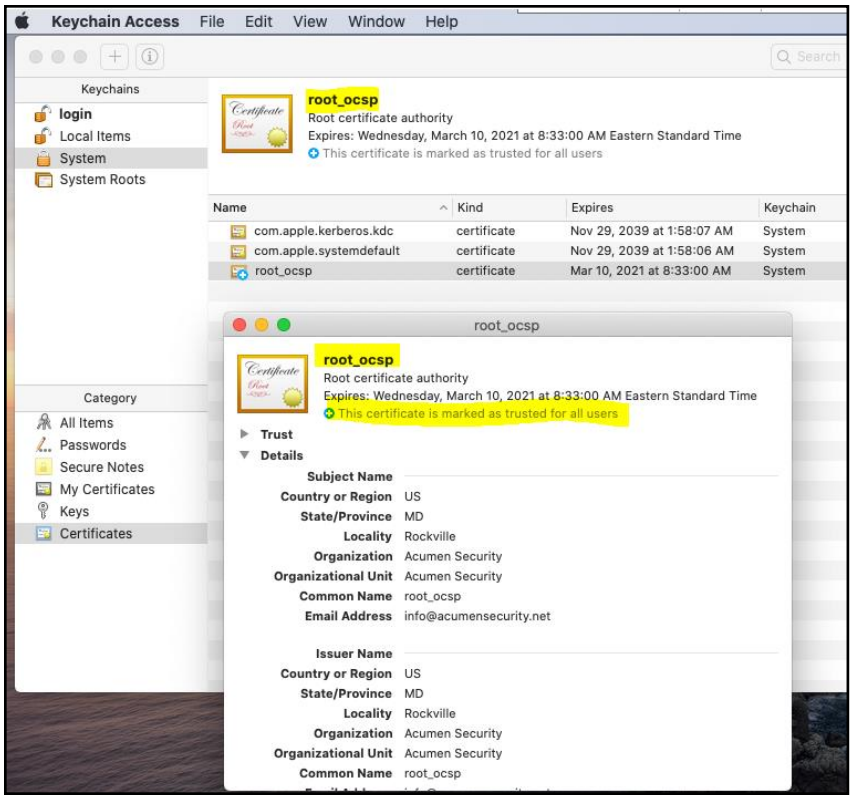
- Double click on root_ocsp certificate to view its contents as shown below:



- Click on arrow to the left of *Trust* and select Always Trust







8.1 Configure TOE for TLS Mutual Authentication

The user can configure the TOE for mutual authentication as described below:

- Obtain a TLS Client certificate.
- Install the TLS Client Certificate on the TOE Keychain.
- Establish a connection with the TLS webserver that requests the TLS Client certificate.
- During the TLS handshake, the TOE will prompt the user to enter their account password. This password proves as the authorization factor to use the TLS Client certificate.
- After entering the correct password in the password prompt, the TOE will use its' TLS Client certificate to authenticate itself to the TLS webserver.

9. TOE Cryptographic Operation – Hashing, Encryption and Decryption

The TOE performs encryption and decryption services for data using AES-CBC (as defined in NIST SP 800-38A), AES-XTS (as specified in IEEE 1619) and AES_GCM (as defined in NIST SP 800-38D) with cryptographic key sizes 128-bit and 256-bit.

The TOE supports Cryptographic hashing services conforming to FIPS PUB 180-4. The hashing

algorithms are used for signature services and HMAC services. The following hashing algorithms are supported: SHA-1, SHA-256, SHA-384 and SHA-512. The message digest sizes supported are: 160 bits, 256 bits, 384 bits and 512 bits.
Note: By default, the TOE supports the hash sizes. The TOE does not allow the user to configure the hash size.

10. Buffer Overflow Protections

The TOE employs Stack-based Buffer Overflow Protections (SBOP) using address space layout randomization and non-executable stack and heap. The host platforms of the TOE support a feature called the NX bit which allows the operating system to mark certain parts of memory as non-executable. If the processor tries to execute code in any memory page marked as non-executable, the program will crash. The TOE leverages this feature by marking the stack and heap as non-executable. This makes buffer overflow attacks difficult because any attack(s) that places executable code on the stack or heap and then tries to execute that code will fail.

The rationale for all the binaries that are not protected by SBOP are provided below:

- Type 1: The compiler can optimize away stack usage (which is certainly something macOS heavily rely on for performance reasons).
- Type 2: Some binaries are just small entry points that rely on system frameworks for all of their functionality. There, the binary itself is going to be really small (less than ~1000 instructions, sometimes as small as 10 instructions), so is much less likely to need stack protection.
- Type 3: There are very short program/functions that does not access the stack (and just forwards to system frameworks to do the real work)
- Type 4: There are tiny binaries with a single trivial function that does not need stack protections or tiny wrappers that does not make use of the stack.
- Type 5: Some binaries do not access the stack in any kind of vulnerable way.

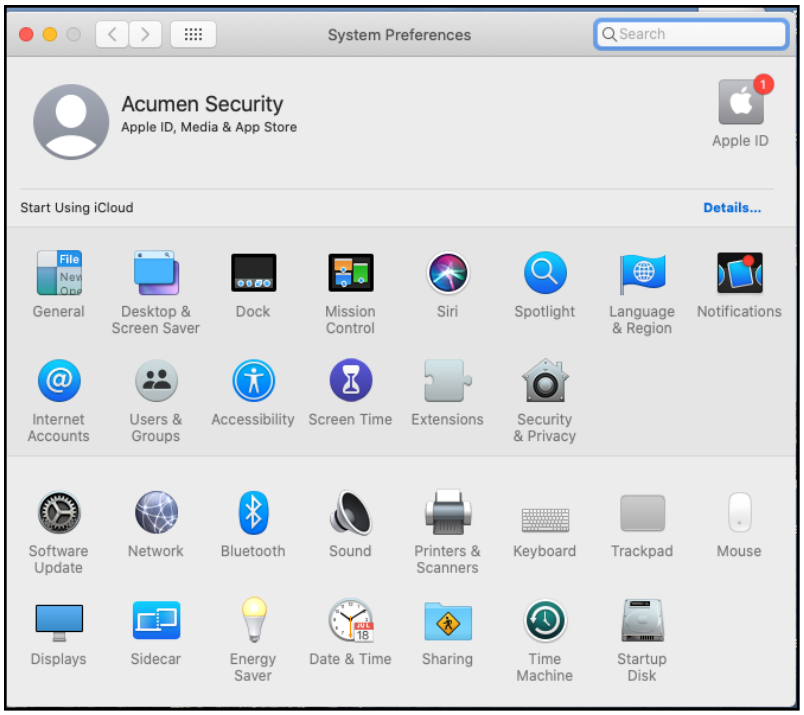
The TOE also randomizes process address memory location with 16 bits of entropy.

Note: By default, the TOE implements the SBOP feature which cannot be disabled by the user.

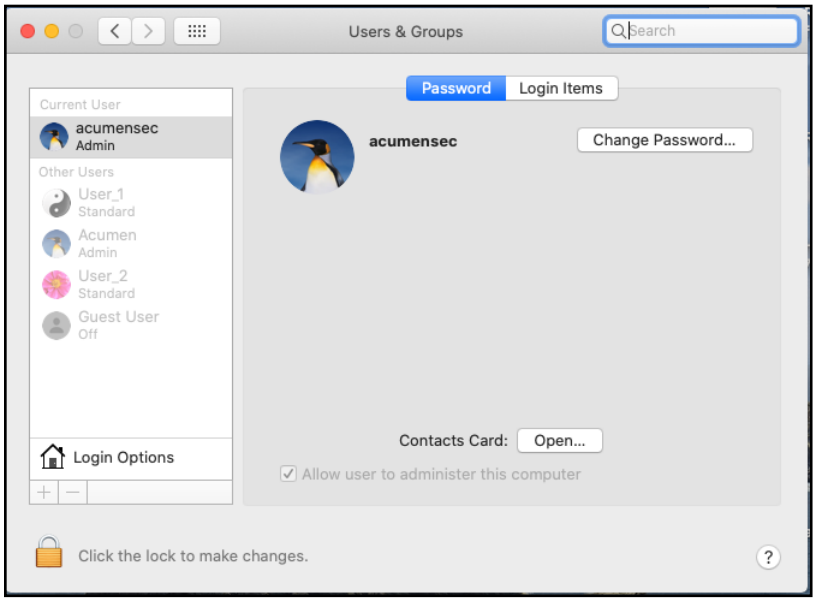
11. Creating User Accounts

The TOE allows the user to add or create a new user(s). The process of creating a new user is shown below:

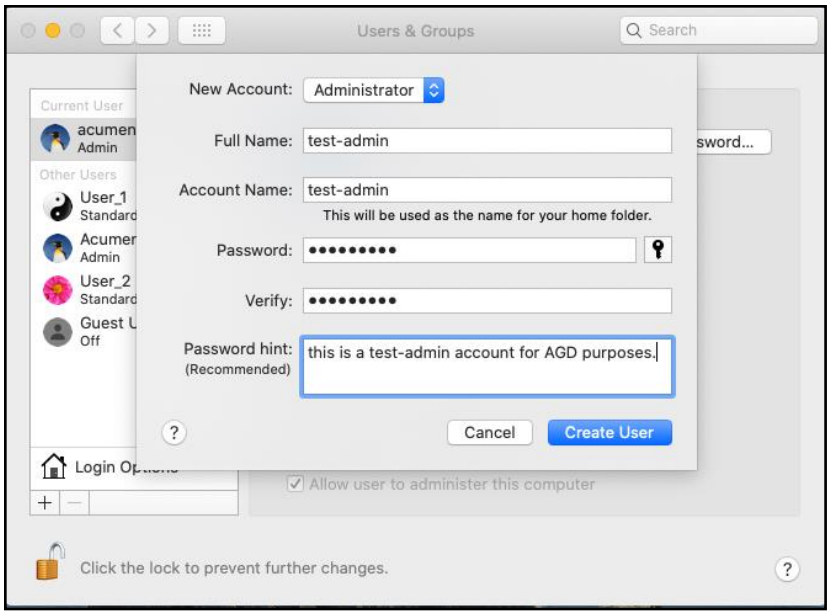
- Start System Preferences application:



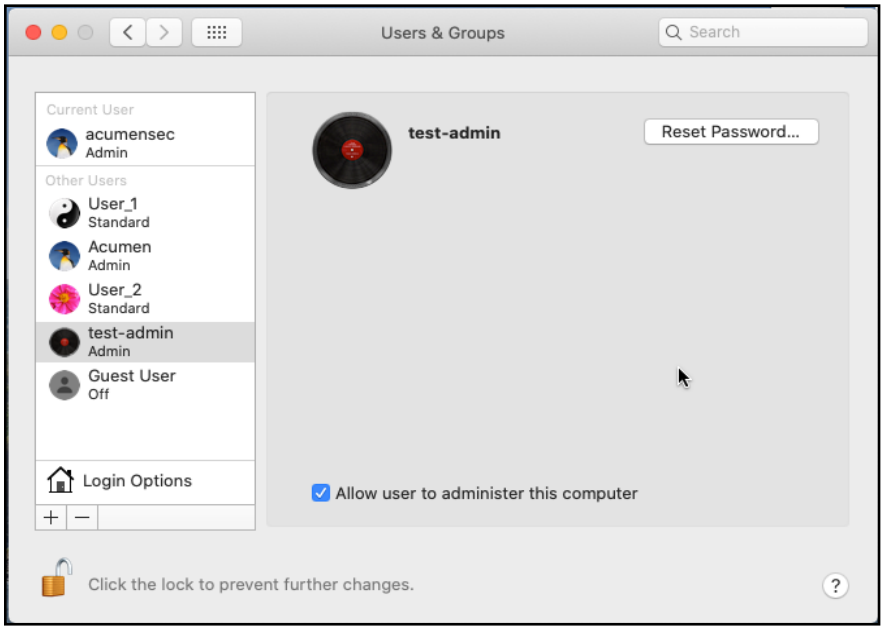
- Select Users & Groups



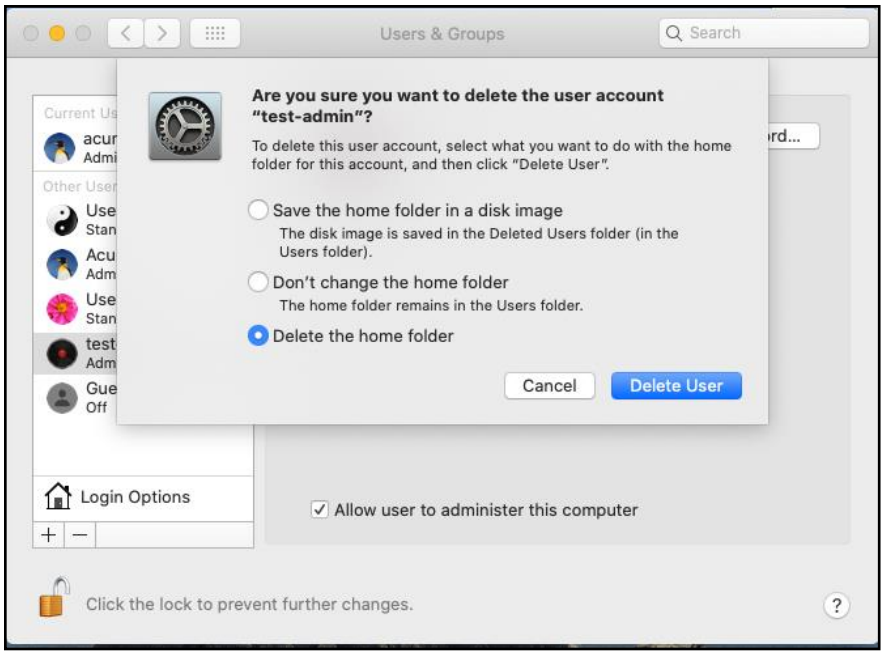
- Click on Click the lock to make changes and enter your current login password.
- Then click on + symbol and add a new user as shown below:



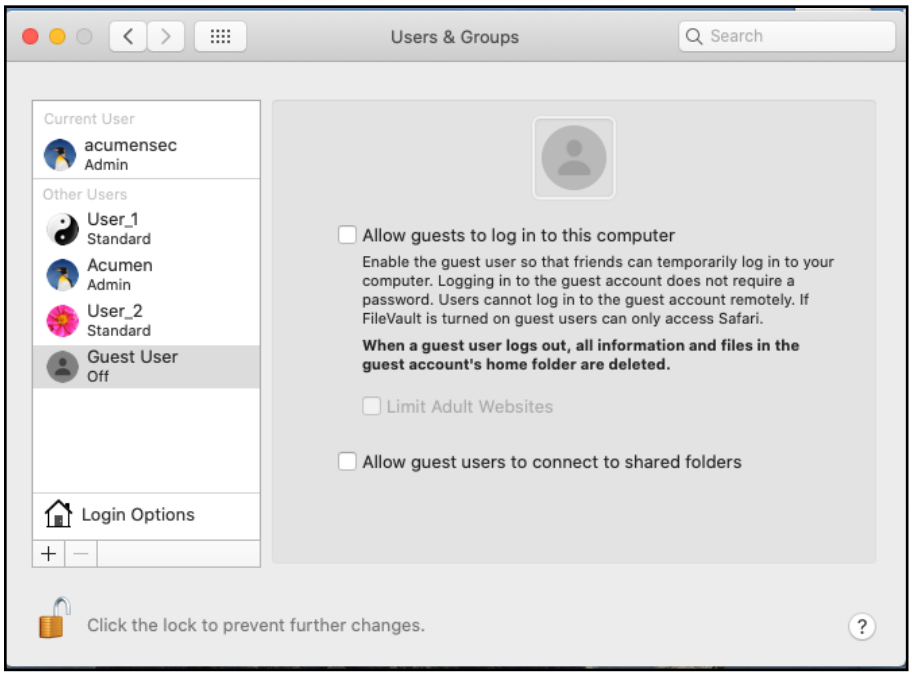
- Click on Create User.
- After clicking on Create, a new user test-admin is added to the TOE with Administrator privileges.



- To delete a user, click on – symbol and click on Delete.



- After deleting the user, the TOE looks like below:



After successfully creating the user account(s), the users can locally administer the TOE. The trusted path communication is secured using TLS v1.2 protocol.

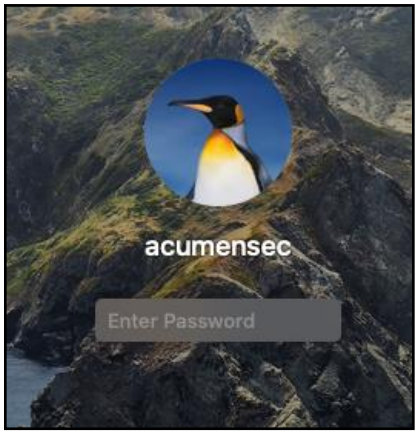
11.1. Locking an Account

The TOE allows the user to lock a user account as shown below:

- Click on Apple symbol

- Click on Lock Screen.
- The TOE will now lock the screen for the currently signed in User.

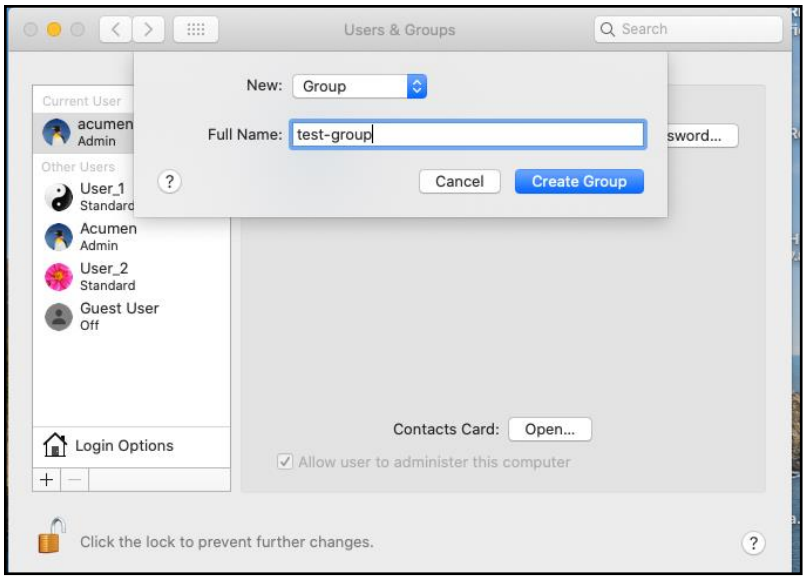
- After locking, the TOE screen looks like below:



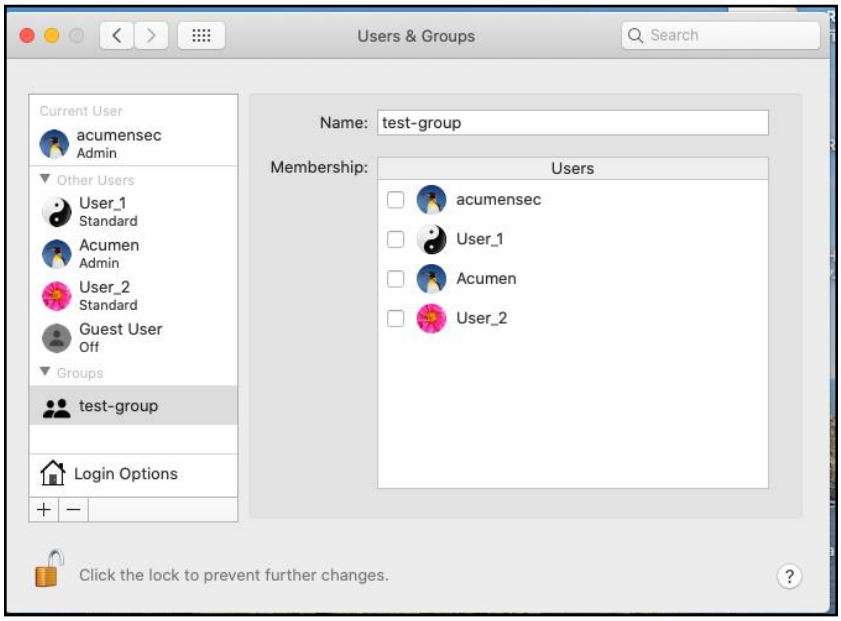
11.2. Creating Groups

The TOE allows the user to create Group(s) as shown below:

- Open System Preferences and click on Users & Accounts
- Click on + symbol and select Group



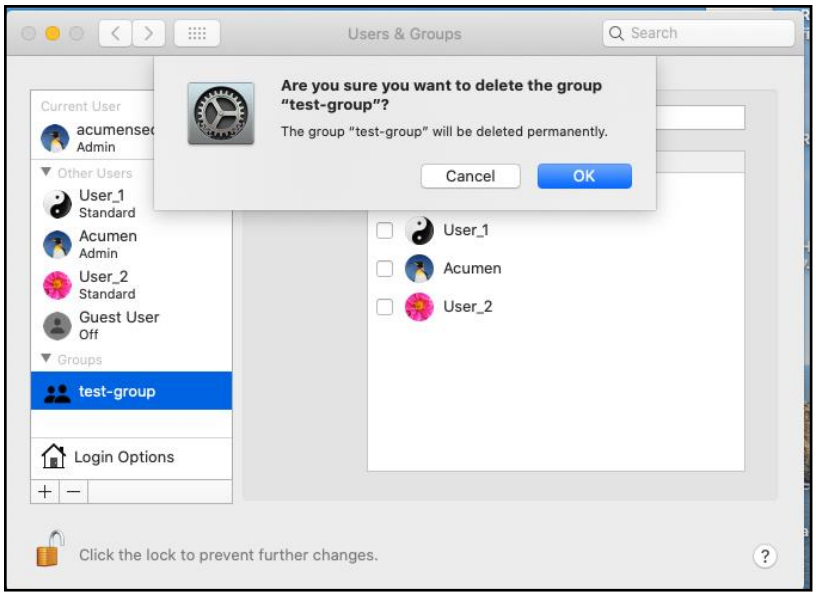
- Click on Create Group
- New group will be created as shown below:



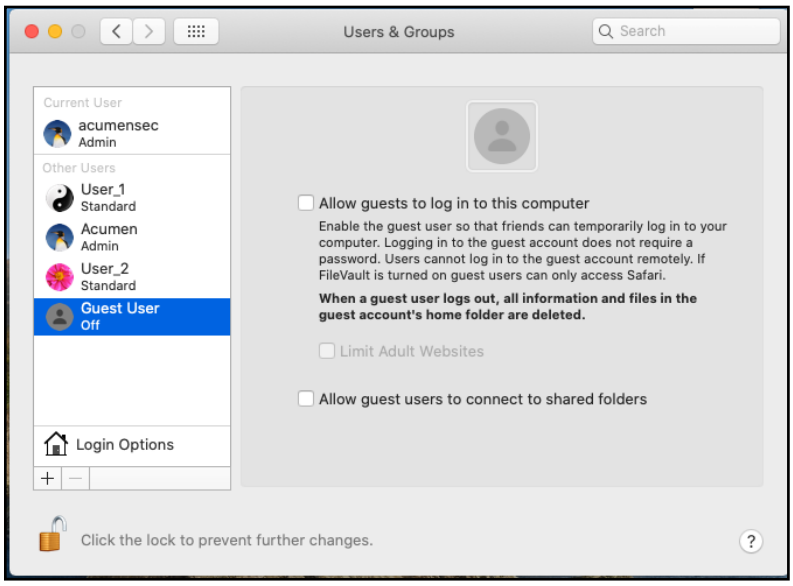
11.3. Modifying or Deleting Groups

The TOE allows the user to delete group(s) as shown below:

- Open System Preferences and click on Users & Accounts
- Click on - symbol to delete group



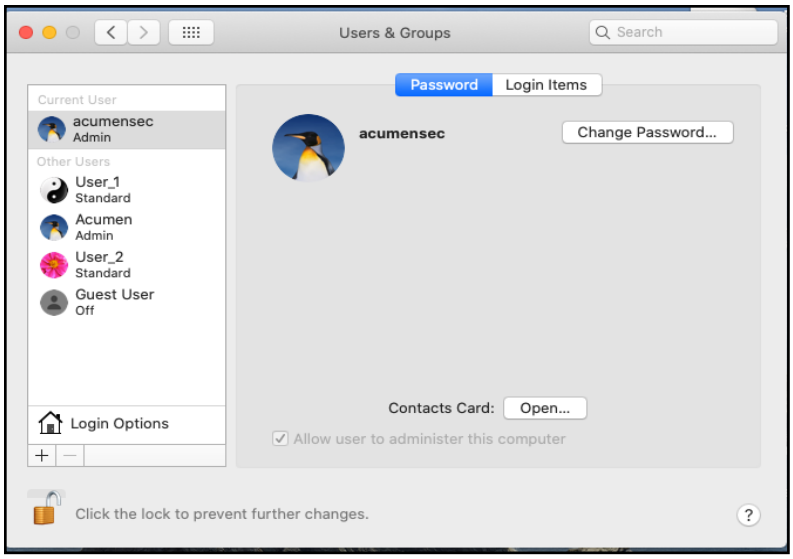
- Click on OK to delete the group.



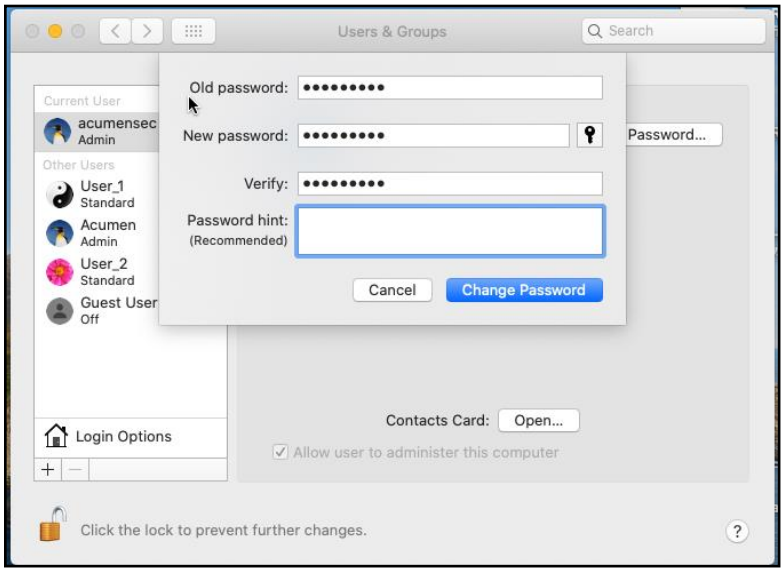
11.4. Changing User Passwords

The TOE allows the User to change the existing password.

- Open System Preferences and click on Users & Groups



- Click on Change Password



- Click on Change Password and the new password will be enforced by the TOE.

12. Password Policy

The TOE supports a password length of minimum 8 characters including Letters (Uppercase and Lowercase), Numbers and Special Symbols. To apply a password policy the pwpolicy command uses the -setaccountpolicies subcommand. This subcommand sets (replaces) the account polices for the specified user. If no user is specified, it sets the global account policies. This subcommand takes one argument: the path of the XML file containing the policies. To import a global policy, it must be saved in a file (e.g. policy_file) and imported by typing the following command in Terminal:

sudo pwpolicy -setaccountpolicies [path to policy_file]

The following is an example of the text for a password policy that will configure the minimum password length to 8, minimum special characters to 2, minimum numeric characters to 2, minimum upper and lower case characters to 2 (each), and sets the timeout between failed password attempts to 60 seconds and the maximum limit of authentication attempts to 3 until lockout:

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version ="1.0">
<dict>
   <key>policyCategoryPasswordContent</key>
   <array>
         <dict>
         <key>policyContent</key>
               <string>policyAttributePassword matches
'.{8,}+'</string>
         <key>policyIdentifier</key>
               <string>Has at least 8 characters</string>
         <key>policyParameters</key>
               <dict>
               <key>minimumLength</key>
                     <integer>8</integer>
               </dict>
         </dict>

         <dict>
         <key>policyContent</key>
               <string>policyAttributePassword matches '(.*[^a-zA-Z0-
9].*){2,}+'</string>
         <key>policyIdentifier</key>
               <string>Has at least 2 special
character</string>
         <key>policyParameters</key>
               <dict>
               <key>minimumSymbols</key>
                     <integer>2</integer>
               </dict>
         </dict>

         <dict>
         <key>policyContent</key>
               <string>policyAttributePassword matches
'(.*[0-9].*){2,}+'</string>
         <key>policyIdentifier</key>
               <string>Has at least 2 numbers</string>
         <key>policyParameters</key>
               <dict>
               <key>minimumNumericCharacters</key>
                     <integer>2</integer>
               </dict>
         </dict>
```

```
            <dict>
            <key>policyContent</key>
                  <string>policyAttributePassword matches
   '(.*[A-Z].*){2,}+'</string>
            <key>policyIdentifier</key>
                  <string>Has at least 2 upper case
   letters</string>
            <key>policyParameters</key>
                  <dict>
                  <key>minimumAlphaCharacters</key>
                        <integer>2</integer>
                  </dict>
            </dict>
            <dict>
            <key>policyContent</key>
                  <string>policyAttributePassword matches
   '(.*[a-z].*){2,}+'</string>
            <key>policyIdentifier</key>
                  <string>Has at least 2 lower case
   letters</string>
            <key>policyParameters</key>
                  <dict>
                  <key>minimumAlphaCharactersLowerCase</key>
                        <integer>2</integer>
                  </dict>
            </dict>

            <dict>
            <key>policyContent</key>
                  <string>(policyAttributeFailedAuthentications
   &lt; policyAttributeMaximumFailedAuthentications) OR
   (policyAttributeCurrentTime &gt
   (policyAttributeLastFailedAuthenticationTime +
   autoEnableInSeconds))</string>
            <key>policyIdentifier</key>
                  <string>Authentication Lockout</string>
            <key>policyParameters</key>
                  <dict>
                  <key>autoEnableInSeconds</key>
                        <integer>60</integer>

      <key>policyAttributeMaximumFailedAuthentications</key>
                        <integer>3</integer>
                  </dict>
            </dict>

   </array>
   </dict>
</plist>
```

Any of the integer values in the above text can be modified to update the policy to the desired values. The number of failed authentication attempts is configurable within the range of 1-50 attempts. The following steps are then performed to import and apply the policies:

1. The XML text is saved in a text file. For this example, the file is titled "policy_file" and is saved to the Desktop of the Mac.
2. Open Terminal
3. Enter the following command in Terminal: sudo pwpolicy -setaccountpolicies ~/Desktop/policy_file
4. Enter the administrator password when prompted
5. When the password is entered correctly you will see "Setting global account policies". This indicates that the policy has been accepted.

13. Setting System Date and Time

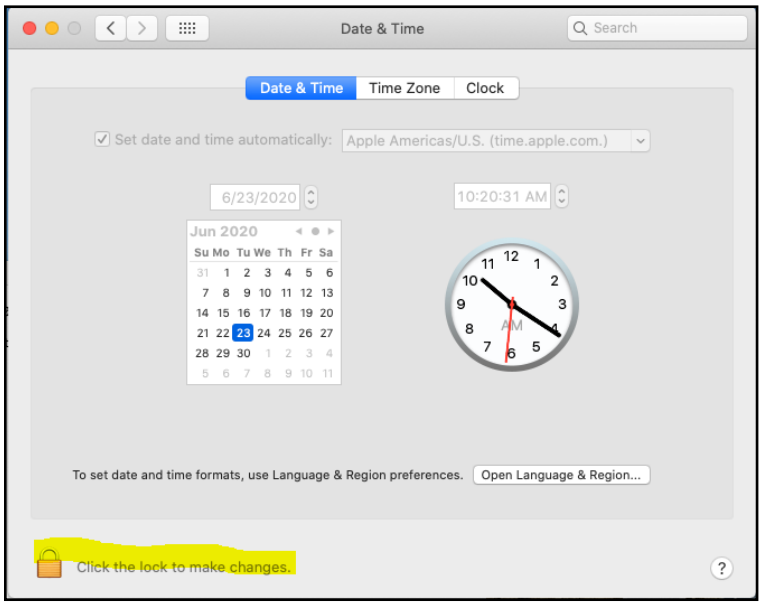
The TOE allows the user to set the time either automatically or manually. The TOE requires the user to authenticate prior to making any change to Date and Time settings.

13.1. Change System Date and Time:

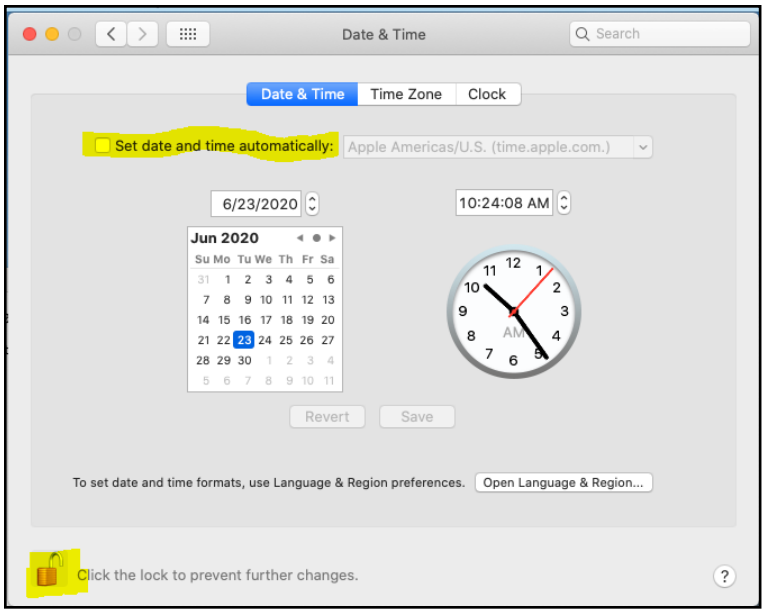
1. Navigate to Apple symbol and click on System Preferences:

- Click on Date & Time

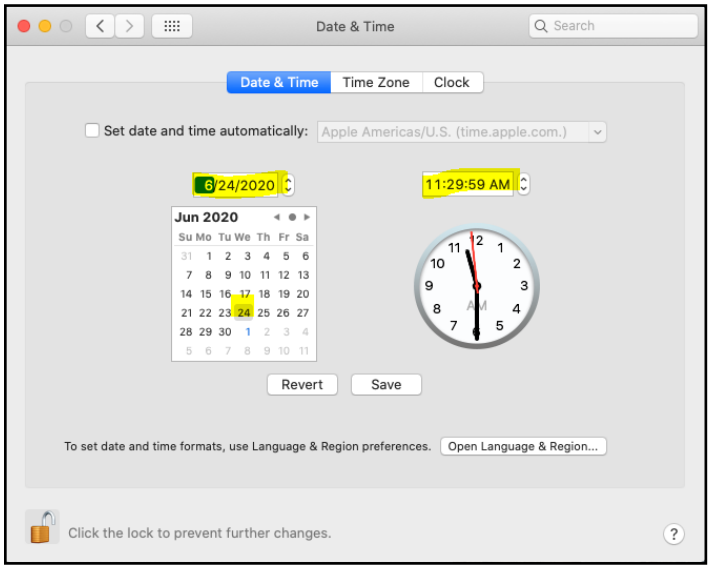
2. Click the lock to make changes. This will prompt the user to enter their password.
    - Note: The option "Set Date and time automatically" is set by default. However, the user can make changes to the default setting and enter date and time manually.



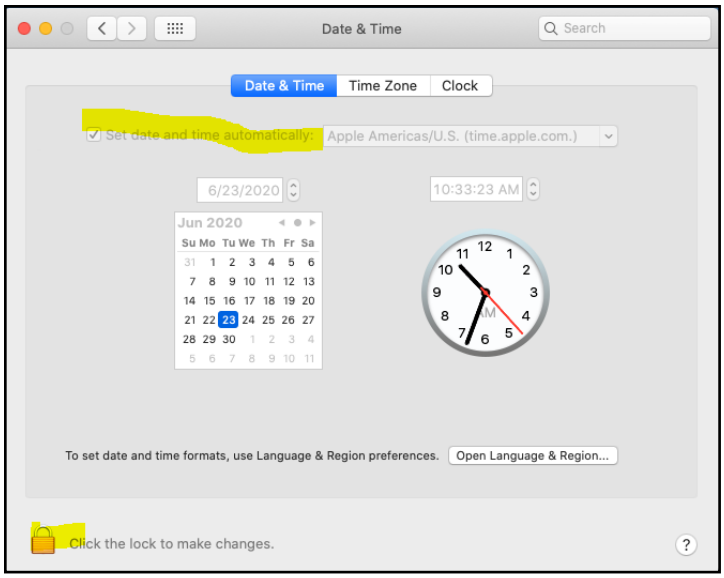
3. "Set date and time automatically" option is disabled.



4. The date and time are manually changed to June 24,2020 11:29:59 AM



5. The date and time can again be set to “automatic” as shown below.



14. Auditing

The audit subsystem is controlled by the audit utility (/usr/sbin/audit). This utility transitions the system in and out of audit operation. The default configuration of the audit mechanism is controlled by a set of configuration files in /etc/security. If auditing is enabled, the /etc/rc startup script will start the audit daemon at system startup. All the features of the daemon are controlled by the audit utility and audit_control file. The audit subsystem generates warnings when relevant events such as storage space exhaustion and errors in operation are recognized during audit startup or log rotation. These warnings are communicated to the audit_warn script, which can then communicate these events to the authorized administrator.
The TOE logging system captures messages across all levels of the system, and it stores the log data in memory and data store on disk. Audit events are only accessible by administrators. To access the audit logs, the TOE provides built-in utilities “audit”, “praudit”, and “auditreduce”. All audit records are BSM compliant, and any BSM Audit Tool could be used for viewing audit logs. Audit events are generated for the following audit functions:

- Start-up and shut down of the audit functions,
- Authentication events (Success/Failure),
- Use of privileged/special rights events (Successful and unsuccessful security, audit, and configuration changes) and
- Privilege or role escalation events (Success/Failure).

Each audit record contains the following information:

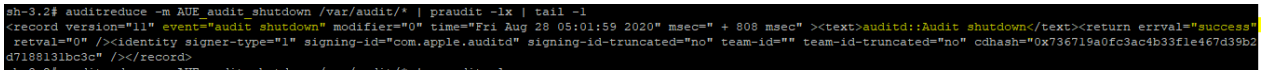
- Date and time of the event, type of event, subject identity (if applicable), and outcome (success or failure) of the event.
    - **Date and Time of Event:** This information is represented by the field “time”. This field represents the exact date and time of the audit event. Example, time=Aug 28 13:54:41 2020.

- **Type of event:** This information is represented by the field “event”. This field represents the audit event type. Example, event= “AUE_audit_startup”.
- **Subject Identity:** This information is represented by the field “identity signer-type”. This field represents the Subject Identity corresponding to the audit event. Example, identity signer-type=1. For audit events such as authentication or privileged access events, an additional field “uid” is included in the audit log. The “uid” field represents the actual user that executed the TOE management function.
- **Outcome:** This information is represented by the field “return errval”. If the audit event is successful, then the value of “return errval” will be shown as “success”. Similarly, if the audit event is unsuccessful then the value of “return errval” will be shown as “failure”. Example, “return errval=success” or “return errval=failure”.

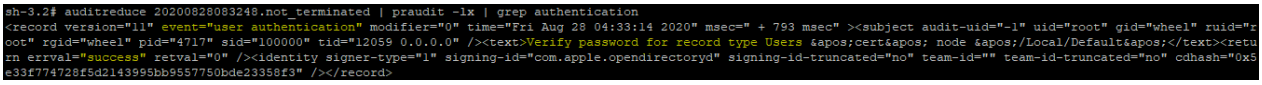
The above TOE audit log details can be observed in the screenshots below:

- TOE Start-up audit log

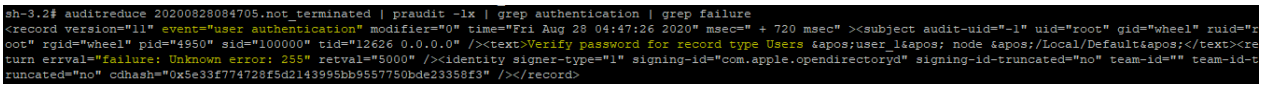
- TOE Shut down audit log



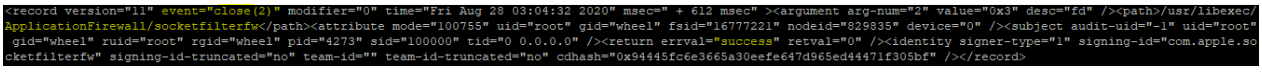
- TOE authentication events
  - TOE authentication event: Success



  - TOE authentication event: Failure

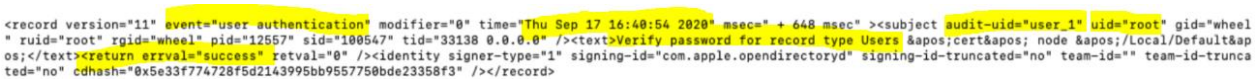


- Use of privileged/special rights events events (Successful and unsuccessful security, audit, and configuration changes)
  - Use of privileged/special rights events: Success

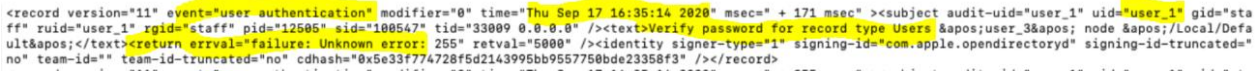


  - Use of privileged/special rights events: Failure

- Privilege or role escalation events (Success/Failure)
  - Privilege escalation: Success



- Privilege escalation: Failure



By default, the TOE logs all audit events as highlighted above. The TOE does not allow the user to stop the audit log. The TOE stores the Audit logs at the locations below:

- System Log Folder: /var/log
- System Log: /var/log/system.log
- Mac Analytics Data: /var/log/DiagnosticMessages
- System Application Logs: /Library/Logs
- System Reports: /Library/Logs/DiagnosticReports
- User Application Logs: ~/Library/Logs (in other words, /Users/NAME/Library/Logs)
- User Reports: ~/Library/Logs/DiagnosticReports

14.1. Command Line Programs

14.1.1. audit

Auditing is managed by the audit utility. The audit utility follows this syntax: audit [-einst]

The audit utility controls the state of the auditing sub-system. The optional file operand specifies the location of the audit_control input file. The default file is default /etc/security/audit_control
You can use the following options with audit:

Sr. No	Option	Description
1	-e	Forces the audit system to immediately remove audit log files that meet the expiration criteria specified in the audit control file without doing a log rotation
2	-i	Initializes and starts auditing
3	-n	Forces the audit system to close the existing audit log file and rotate to a new log file in a location specified in the audit control file
4	-s	Specifies that the audit system should [re]start and re-read its configuration from the audit control file. A new log file will be created

5	-t	Specifies that the audit system should terminate. Log files are closed and renamed to indicate the time of the shutdown

For more information on using audit, see the man pages in Terminal, by typing:

```
man audit
```

14.1.2. auditreduce

The auditreduce utility allows you to select events that have been logged in the audit records. Matching audit records are printed to the standard output in their raw binary form. If no filename is specified, the standard input is used by default. Use the praudit utility to print the selected audit records in human-readable form.

The auditreduce tool follows this syntax:

```
auditreduce [-A] [-a YYYYMMDD[HH[MM[SS]]]] [-b YYYYMMDD[HH[MM[SS]]]] [-c flags]
[-d YYYYMMDD] [-e euid] [-f egid] [-g rgid] [-j id] [-m event] [-o object=value] [-r ruid] [-u auid]
[-v][file …]
```

You can use the following options with auditreduce:

Sr. No	Option	Formatting & Description
1	-A	Select all records
2	-a	YYYYMMDD[HH[MM[SS]]] Select records that occurred after or on the given datetime.
3	-b	YYYYMMDD[HH[MM[SS]]] Select records that occurred before the given datetime
4	-c	Flags. Select records matching the given audit classes specified as a comma separated list of audit flags
5	-d	YYYMMDD, Select records that occurred on a given date. This option cannot be used with -a or -b.
6	-e	euid Select records with the given effective user ID or name.
7	-f	egid Select records with the given effective group ID or name.
8	-g	rgid Select records with the given real group ID or name
9	-j	id Select records having a subject token with matching ID.

10	-m	Event. Option can be used more tha once to select records of multiple event types. See audit_ event(5) for a description of audit event names and numbers.
11	-o	Object=value. File file Select records containing path tokens, where the pathname matches one of the comma delimited extended regular expression contained in given specification. Regular expressions which are prefixed with a tilde (‘~’) are excluded from the search results. These extended regular expressions are processed from left to right, and a path will either be selected or deselected based on the first match msgqid - Select records containing the given message queue ID pid - Select records containing the given process ID semid - Select records containing the given semaphore ID shmid - Select records containing the given shared memory ID.
12	-r	ruid - ruid Select records with the given real user ID or name.
13	-U	Use unbuffered output. This option is automatically set when the input file is a character device (e.g. /dev/auditpipe), allowing records to be fully processed as they are generated.
14	-u	auid - Select records with the given audit ID
15	-v	Invert sense of matching, to select records that do not match.

Examples:

To select all records associated with effective user ID root from the audit log

/var/audit/20031016184719.20031017122634:

auditreduce -e root /var/audit/20031016184719.20031017122634

To select all setlogin(2) events from that log:

auditreduce -m AUE_SETLOGIN var/audit/20031016184719.20031017122634

For more information on using auditreduce, see the man pages in Terminal, by typing:

man auditreduce

14.1.3. praudit

The praudit utility prints the contents of the audit records. The audit records are displayed in standard output (stdout). If no filename is specified, standard input (stdin) is used by default.

The praudit tool uses this syntax:

```
praudit [-lnpx] [-r | -s] [-d del] [file ...]
```

You can use praudit with the following options:

Sr. No	Option	Description
1	-d	del Specifies the delimiter. The default delimiter is the comma
2	-l	Prints the entire record on the same line. If this option is not specified, every token is displayed on a different line.
3	-n	Do not convert user and group IDs to their names but leave in their numeric forms.
4	-p	Specify this option if input to praudit is piped from the tail(1) utility. This causes praudit to sync to the start of the next record.
5	-r	Prints the records in their raw form. Show records and event types in a numeric form (also known as raw form). This option is exclusive from -s
6	-s	Prints the records in their short form. Show records and events in a short textual representation. This option is exclusive from -r.
7	-x	Print audit records in the XML output format.

If the raw or short forms are not specified, the default is to print the tokens in their raw form. Events are displayed as per their descriptions given in /etc/security/audit_event; UIDs and GIDs are expanded to their names; dates and times are displayed in human-readable format.

For more information on using praudit, see the man pages in Terminal, by typing:

```
man praudit
```

14.1.4. Managing audit log files

If auditing is enabled, the auditing subsystem adds records of auditable events to an audit log file. The name of an audit log file consists of the date and time it was created, followed by a period, and the date and time it was terminated, for example:

```
20040322183133.20040322184443
```

The audit subsystem appends records to only one audit log file at any given time. The currently active file has a suffix ".not_terminated" instead of a date and time.

Audit log files are stored in the directories specified in the audit_control file. The audit subsystem creates an audit log file in the first directory specified.

When less than the 'minfree 'amount of disk space is available on the volume containing the audit log file, the audit subsystem will:

- Issue an audit_warn soft warning
- Terminate the current audit log file
- Create a new audit log file in the next specified directory

Once all directories specified have exceeded this 'minfree 'limit, auditing will resume in the first directory again. However, if that directory is full, an auditing subsystem failure may occur.

Administrators may also choose to terminate the current audit log file and create a new one manually using the audit utility. This action is commonly referred to as "rotating the audit logs".

Use audit -n to rotate the current log file. Use audit -s to force the audit subsystem to reload its settings from the audit_control file (this will also rotate the current log file.)

14.1.5. Deleting audit records

An administrator can clear the audit trail by deleting audit files. Administrators can delete audit files from the command line.

Example:

```
sudo rm /var/audit/20031016184719.20031017122634
```

Warning: The administrator should not delete the currently active audit log.

14.2. Audit Control Files

There are several text files the audit system uses to control auditing and write audit records. The default location for these files is the /etc/security directory.

14.2.1. audit_class

The audit_class file contains descriptions of the auditable event classes on the system. Each auditable event is a member of an event class. Each line maps an audit event mask (bitmap) to a class and a description.

Example entries in this file are:

```
0x00000000:no:invalid class
0x00000001:fr:file read
0x00000002:fw:file write
0x00000004:fa:file attribute access
0x00000080:pc:process
```

0xffffffff:all:all flags set

File Location

/etc/security/audit_class

For more information on using audit_class, see the man pages in Terminal, by typing:

man audit_class

14.3. audit_control

The audit_control file contains several audit system parameters. Each line of this file is of the form parameter:value.

You can use audit_control with the following parameters:

Sr. No	Option	Description
1	dir	The directory where audit log files are stored. There may be more than one of these entries. Changes to this entry can only be enacted by restarting the audit system. See audit (8) for a description of how to restart the audit system.
2	flags	Specifies which audit event classes are audited for all users. audit_user (5) describes how to audit events for individual users. See the information below for the format of the audit flags.
3	host	Specify the hostname or IP address to be used when setting the local system's audit host information. This hostname will be converted into an IP or IPv6 address and will be included in the header of each audit record. Due to the possibility of transient errors coupled with the security issues in the DNS protocol itself, the use of DNS should be avoided. Instead, it is strongly recommended that the hostname be specified in the /etc/hosts file. For more information see hosts (5).
4	naflags	Contains the audit flags that define what classes of events are audited when an action cannot be attributed to a specific user.
5	minfree	The minimum free space required on the file system audit logs are being written to. When the free space falls below this limit a warning will be issued. If no value for the minimum free space is set, the default of 20 percent is applied by the kernel
6	policy	A list of global audit policy flags specifying various behaviours, such as fail stop, auditing of paths and arguments, etc.

7	filesz	Maximum trail size in bytes; if set to a non-0 value, the audit daemon will rotate the audit trail file at around this size. Sizes less than the minimum trail size (default of 512K) will be rejected as invalid. If 0, trail files will not be automatically rotated based on file size. For convenience, the trail size may be expressed with suffix letters: B (Bytes), K (Kilobytes), M (Megabytes), or G (Gigabytes). For example, 2M is the same as 2097152.
8	expire-after	Specifies when audit log files will expire and be removed. This may be after a time period has passed since the file was last written to or when the aggregate of all the trail files have reached a specified size or a combination of both. If no expire- after parameter is given, then audit log files will not expire and be removed by the audit control system. See the information below for the format of the expiration specification.

14.3.1. Audit Flags

Audit flags are a comma delimited list of audit classes as defined in the audit_class file. Event classes may be preceded by a prefix that changes their interpretation.

The following prefixes can be used for each class:

Sr. No	Prefix	Description
1	(none)	Record both successful and failed events.
2	+	Record successful events.
3	-	Record failed events.
4	^	Record neither successful nor failed events.
5	^+	Do not record successful events.
6	^-	Do not record failed events.

14.3.2. Default

The following settings appear in the default audit_control file:

```
dir:/var/audit flags:lo,aa minfree:5 naflags:lo,aa policy:cnt,argv filesz:2M expire-after:10M
```

The flags parameter above specifies the system-wide mask corresponding to login/logout as well as authentication and authorization events. The policy parameter specifies that the system should neither fail stop nor suspend processes when the audit store fills and that command line arguments should be audited for AUE_EXECVE events. The trail file will be automatically rotated by the audit daemon when the file size reaches approximately 2MB. Trail files will expire when their aggregate size exceeds 10MB.

File Location

```
etc/security/audit_control
```

For more information on using audit_control, see the man pages in Terminal, by typing:

```
man audit_control
```

14.4. audit_event

The audit_event file contains descriptions of the auditable events on the system. Each line maps an audit event number to a name, a description, and a class. Entries are of the form:

```
eventnum:eventname:description:eventclass
```

Each eventclass should have a corresponding entry in the audit_class file.

Example entries in this file are:

```
0:AUE_NULL:indir system call:no
1:AUE_EXIT:exit(2):pc
2:AUE_FORK:fork(2):pc
3:AUE_OPEN:open(2):fa
```

File Location:

```
/etc/security/audit_event
```

The values for the events in this file must match the values in the bsm_kevents.h file. For more information on using audit_event, see the man pages in Terminal, by typing:

```
man audit_event
```

14.5. audit_user

The audit_user file specifies which audit event classes are to be audited for the given users. If specified, these flags are combined with the system-wide audit flags in the audit_control file to determine which classes of events to audit for that user. These settings take effect when the user logs in.

Each line maps a user name to a list of classes that should be audited and a list of classes that should not be audited. Entries are of the form:

```
username:alwaysaudit:neveraudit
```

In this example, alwaysaudit is a set of event classes that are always audited, and neveraudit is a set of event classes that should not be audited. These sets can indicate the inclusion or exclusion of multiple classes, and whether to audit successful or failed events.

Example entries in this file are:

```
root:lo,ad:no jdoe:-fc,ad:+fw
```

These settings would cause login and administrative events that succeed on behalf of user root to be audited. No failure events are audited. For the user jdoe, failed file creation events are audited, administrative events are audited, and successful file write events are never audited.

File Location:

```
/etc/security/audit_user
```

For more information on using audit_user, see the man pages in Terminal, by typing:

```
man audit_user
```

14.6. audit_warn

audit_warn runs when auditd (8) generates warning messages.

The default audit_warn is a script whose first parameter is the type of warning; the script appends its arguments to /etc/security/audit_messages. Administrators may replace this script with a more comprehensive one would take different actions based on the type of warning. For example, a low-space warning could result in an email message being sent to the administrator.

An example script is below:

```
#!/bin/sh
echo "Audit warning: $@ " >> /etc/security/audit_messages
```

This example script simply appends all of its arguments to the audit_messages file

14.7. Audit Log Files

The dir entry in the audit_control file specifies where audit logs are to be stored.

Multiple dir entries are permitted. The log files are created with names of the form date.date for log files that are closed (terminated), and date.not_terminated for the current log file that has not been closed. The date portion of the log file name is of the form yyyymmddHHMMSS with the hours in 24-hour format.

14.8. Configure the address of remote audit server to which to send audit records

Prerequisites:

- A user account with administrator privileges.
- A Syslog server running and configured to receive the audit events.

To redirect logs in macOS to a syslog server, it is necessary to edit the file syslog.conf located in:

/etc/syslog.conf

Then add the following line (replace it with the IP of your syslog server):

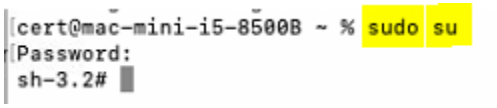
. @syslog_server_ip

To customize the port number (e.g. 514) use @syslog_server_ip:514 instead.

This change should be instant, if not the syslog service has to be restarted typing the following command in Terminal app:

launchctl start com.apple.syslogd

To verify that the configuration was successful, perform a privilege elevation, by typing “sudo su” in the Terminal app:



The audit record should be sent and received by the remote audit server:

15. Reference Identifiers

The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125. The reference identifiers supported are DNS and IP addresses. The TOE does not support certificate pinning. Wild cards are supported.

16. Access Control Policy

The TOE provides access control policy through the System Integrity Protection (SIP). The System Integrity program restricts the root user account (an administrator superuser account) and limits the actions that the root user can perform on protected parts of the Mac operating system. The System Integrity Protection is enabled by default on the TOE; however, it can be disabled. Note that the user should not disable this configuration setting as this setting is designed to help prevent potentially malicious software from modifying protected files and folders on the user’s Mac.

System Integrity Protection includes protection for these parts of the system:

/System

/usr

/bin

/sbin

/var

Apps that are pre-installed with TOE:

Kernel drivers and modules:

-/System/Library/Extensions/

Security audit logs:

-/var/audit/*

Shared libraries:

-/Library/Frameworks/

-/Library/PrivateFrameworks/

-/System/Library/Frameworks/

-/System/Library/PrivateFrameworks/

System executables:

-/Applications

System configuration files:

-System-wide “preferences”

-/Library/Preferences/

-User-specific “preferences”

-/Users/<username>/Library/Preferences

Security Audit Logs:

-/etc/security/audit-control

System-wide local directory services credentials:

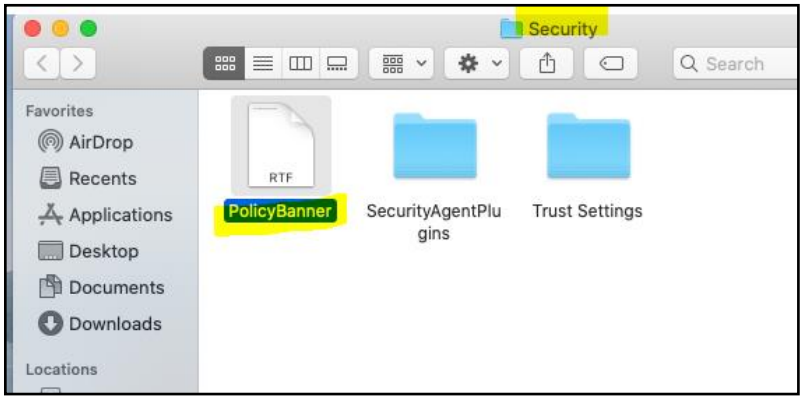
-/private/var/db/dslocal/nodes/Default/

System Integrity Protection is designed to allow modification of these protected parts only by processes that are signed by Apple and have special entitlements to write to system files, such as Apple software updates and Apple installers. Apps that you download from the Mac App Store already work with System Integrity Protection.

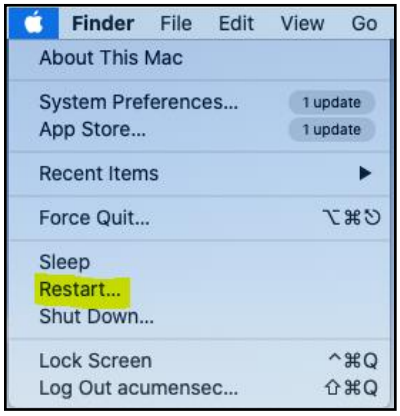
17. Warning Banner

The TOE allows the user to set up a warning banner or message regarding unauthorized use of the TOE prior establishing a user session. Steps to configure a warning banner are outlined below:

- Create a plain text (.txt) or rich text (.rtf) document that contains the message "TEST TEST Warning Message TEST TEST".
- Save the file and enter PolicyBanner for the document name.
- Copy the PolicyBanner file to the /Library/Security/ folder on the TOE.



- Restart the TOE so that the policy banner will take effect.



- After restarting the TOE but prior to starting a user session, the TOE displays the warning banner.

18. Authorization Factors

The TOE supports password and external smart card authentication factors. Passwords of up to 256 characters are supported and can be comprised of any combination of upper-case characters, lower case characters, numbers, and any other 8-bit special character.

For password-based authentication, the user's password, the TOE's UID and a salt value are used to perform a password-based derivation function (PBKDF2) and derive the Top-level Key. The Top-level Key is defined as the Boarder Encryption Value (BEV) and is used to unwrap the User's Keybag with the AES Key Wrap (AES-KW) algorithm (RFC3394). The password is successfully validated if the User's Keybag can be cryptographically unwrapped with AES-KW using the derived Top-level Key and the function will return a "success" result.

For Smart card authentication, the user's smart card must first be registered with the TOE. The registration process involves the derivation of the Top-level Key. Once the Top-level Key has been derived, the SEP randomly generates the USK to encrypt the Top-level Key. The SEP then uses the smart card's public key to encrypt the Unlock Secret Key (USK), which is later used as a challenge for authentication. The user is required to enter the smart card's defined PIN to decrypt the USK and present it to the TOE for decryption of the Top-level Key. The Top-level Key is once again used to unwrap the Class Key with the AES Key Wrap (KW) algorithm. The smart card is validated if the AES KW function does not return a "Fail" result. The TOE supports RSA key sizes of 2048 bits and 3072 bits.

In order to resume from a compliant power state, the user must re-authenticate to the TOE. The user can either authenticate using username and password or by using smart card authentication. If the smart card has already been registered, then the user can use the smart card and the PIN.

18.1 Smart Card Registration

The Smart Card's certificate is sent to the SEP. The SEP randomly generates the USK and encrypts the Top-level key using AES in GCM mode. The SEP then performs a SHA-256 hash on the Smart Card's Public Key (SCpk) and concatenates the value with the encrypted Top-level key to form the Escrow Record value. The SEP stores the Escrow Record in non-volatile memory.

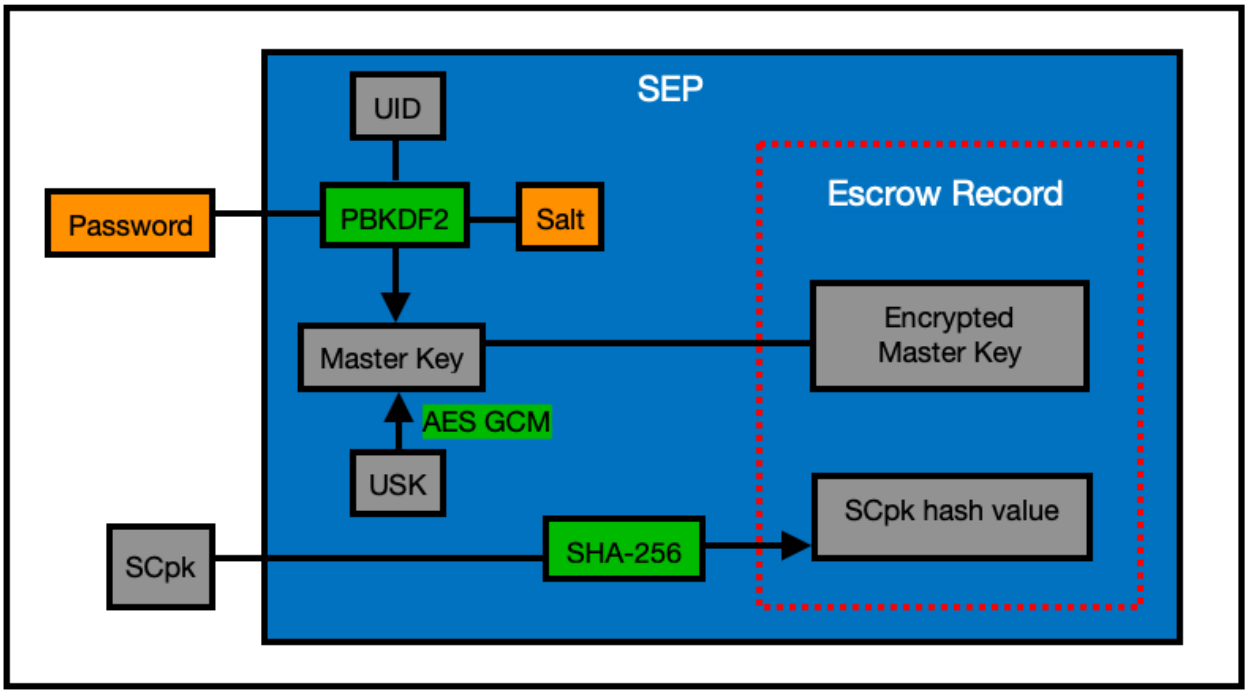


Fig 1: Creating the Escrow Record with SCpk

The SEP then encrypts the USK with the Smart Card's public key (SCpk) or a key associated with the SCpk to form the Encrypted Record. As the SCpk can be an RSA or ECC key, one of the following is performed:

- If the SCpk is an RSA key, the SEP encrypts the USK using the SCpk.
- If the SCpk is an EC key:
    - The SEP generates a key pair (P-256): SEP_ECpk and SEP_ECsk
    - A shared key is established using ECDH with the generated SEP_ECsk and the SCpk
    - The SEP performs a SHA-256 hash over the shared key, SEP_ECpk, and the SCpk. The hash value is the EC_Encryption Key.
    - The SEP encrypts the USK using the EC_Encryption Key with AES in CBC mode

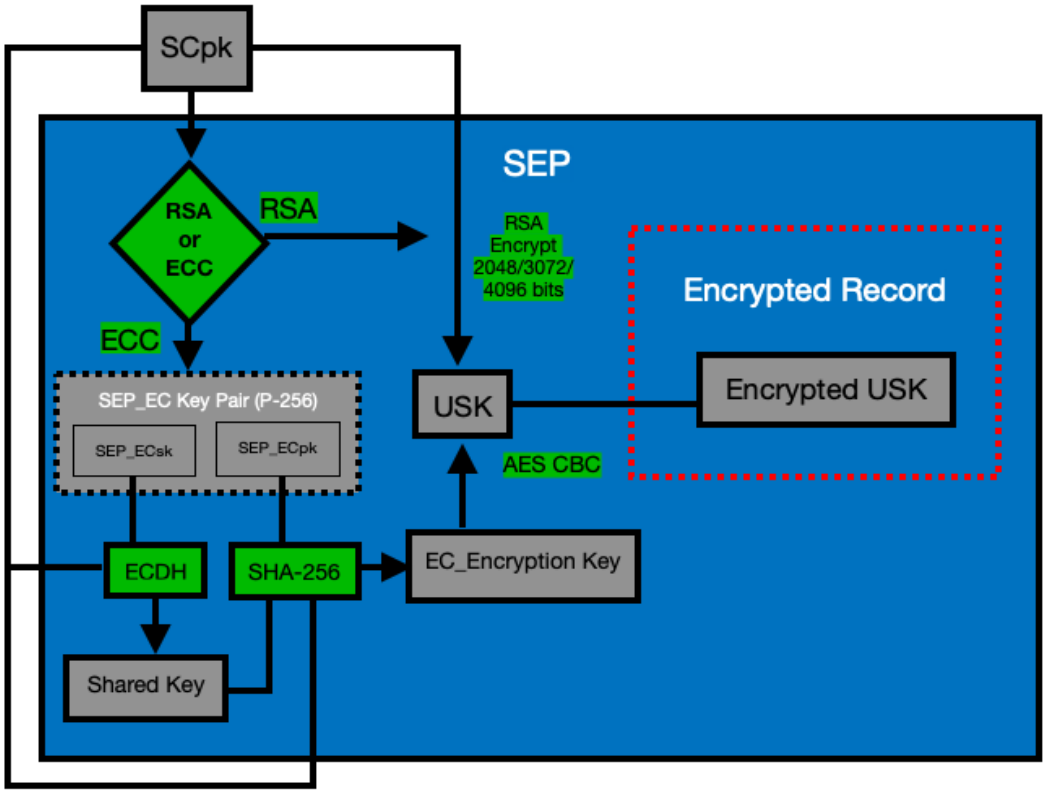


Fig 2: Creating the Encrypted Record

The SEP then performs the following to complete the registration:

- Randomly generates the PK.
- Encrypts the Encrypted Record using the PK with AES-GCM in encryption and authentication mode.
- Creates the Authentication Record by concatenating: the SCpk, the algorithm of the SCpk, and when the SCpk is an EC key, the SEP_ECpk.
- Secures the Authentication Record with the PK using AES-GCM in authentication mode.
- Encrypts the PK using a Class Key using AES in GCM mode. The Smart Card can be configured to be associated with either Class A or Class D keys.
- Sends the Smart Card authentication block (SC_blob) to the application processor (Intel chip) for storage. The SC_blob consists of the concatenation of the following: the encrypted PK, encrypted Encrypted Record, and the secured Authentication Record.
- Cryptographic data is removed from volatile memory.

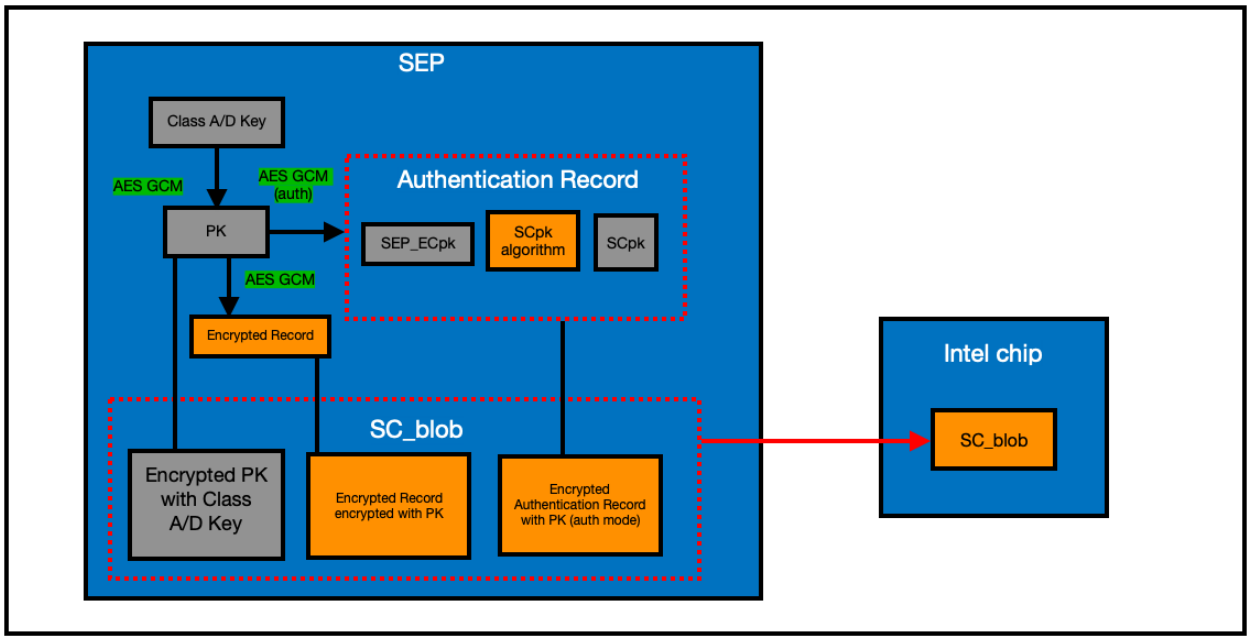


Fig 3: Creating the SC_blob

18.2 Smart Card Authentication

Once the above process is complete, the Smart Card is registered and can be used to authenticate to the TOE. When the Smart Card is connected, the SCpk is provide and the application processor (Intel chip) sends SC_blob(i) to the SEP; where (i) indicates a counter that is incremented after a successful authentication. The SEP then performs the following with the SC_blob:

- Obtains the PK by extracting the encrypted PK and decrypting it using the Class A/D Key
- Obtains the encrypted USK (Encrypted Record) by decrypting and verifying it using the PK
- Verifies the Authentication Record using the PK
- Performs an integrity check on the SCpk by comparing SHA-256 hash of the stored value with the value provided.
- Prepares and sends the Smart Card challenge:
    - If the SCpk is an RSA key, the challenge is only the encrypted USK
    - If the SCpk is an EC key, the challenge is the encrypted USK concatenated with the SEP_ECpk
- A 5 minute timer is started for return of the USK or the authentication process is terminated.

The Smart Card must use its private key to decrypt the USK. The user unlocks the private key by entering the PIN associated with the card. The number of failed attempts to unlock the card is defined by the card specification/configuration and is not known or defined by the TOE. When the designated number of failed attempts is reached, the card will be locked. After successfully entering the correct PIN, the card decrypts the USK and sends it back to the SEP. The SEP retrieves the USK provided and decrypts the Escrow Record to obtain the Top-level key. The Top-level key is then used to decrypt the volume data. After the successful authentication, the SEP generates a new USK and sends SC_blob (i+1) to the application processor.

Additional information on Smart Card configuration can be found at:

- Use a smart card with Mac,
- Prepare for smart card changes in macOS Catalina and
- Configure macOS for smart card-only authentication.

19. Key Destruction

All keys are erased when the host device is powered off, during reboot, when a user locks or logs off the host device, the TOE detects the configured inactivity time has passed and the host device logs out, or when the host device is put to sleep. Once the keys are no longer required, the key that was used to perform the specific operation is erased from volatile memory by performing a single overwrite of zeroes. The erase operation is performed by the SEP and is not configurable by a user. There are no circumstances that do not conform to the key destruction requirement (e.g. sudden unexpected power loss).

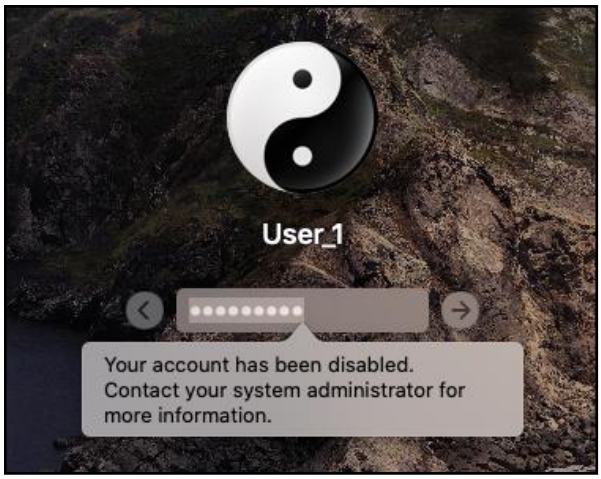
20. Validation of Cryptographic Elements

The TOE requires the validation of the BEV prior to allowing access to TSF data after exiting a Compliant power saving state and it will block validation after 10 consecutive failed validation attempts.

The TOE can be configured for validation as below:

```
cert@mac-mini-i5-8500B ~ % sudo pwpolicy -u User_1 -setpolicy "MaxFailedLoginAttempts=10"
Password:
Setting policy for User_1
cert@mac-mini-i5-8500B ~ %
```

After 10 consecutive failed authentication attempts, the TOE blocks the validation attempts by disabling the user account.



21. Enable Full Disk Encryption

On Mac computers with the Apple T2 Security Chip, encrypted internal storage devices directly connected to the T2 chip leverage the hardware security capabilities of the chip.

Without valid login credentials or a cryptographic recovery key, the internal APFS volume (in macOS 10.15, this includes the System and Data volumes) remains encrypted and is protected from unauthorized access even if the physical storage device is removed and connected to another computer. Internal volume encryption on a Mac with the T2 chip is implemented by constructing and managing a hierarchy of keys, and builds on the hardware encryption technologies built into the chip.

This hierarchy of keys is designed to simultaneously achieve four goals:

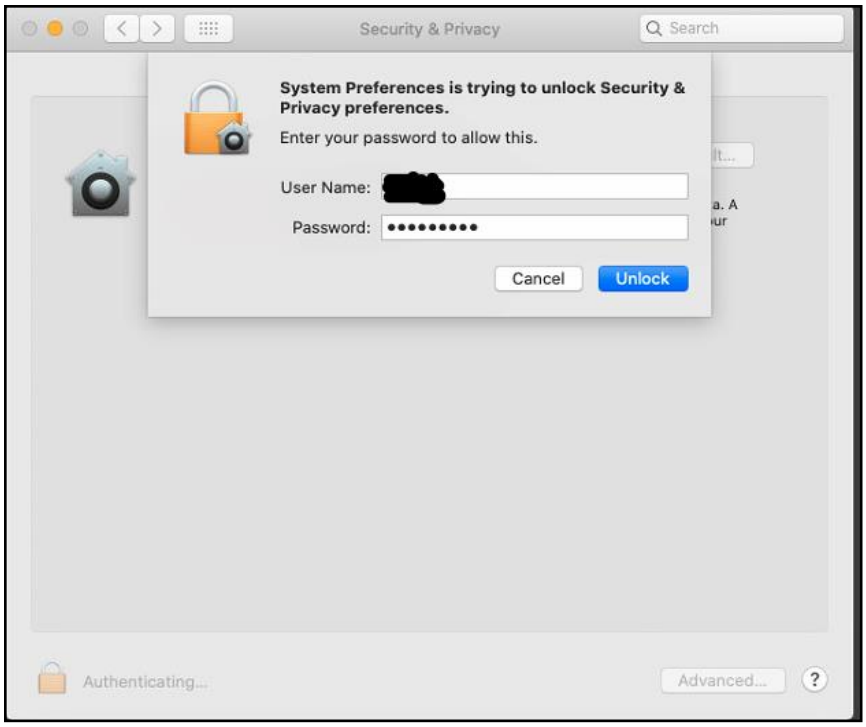
- Require the user's password for decryption.
- Protect the system from a brute-force attack directly against storage media removed from Mac.
- Provide a swift and secure method for wiping content via deletion of necessary cryptographic material.
- Enable users to change their password (and in turn the cryptographic keys used to protect their files) without requiring re-encryption of the entire volume.

Volume and metadata contents are encrypted with this volume key, which is wrapped with the class key. This protection is the default on Mac computers with the T2 chip.

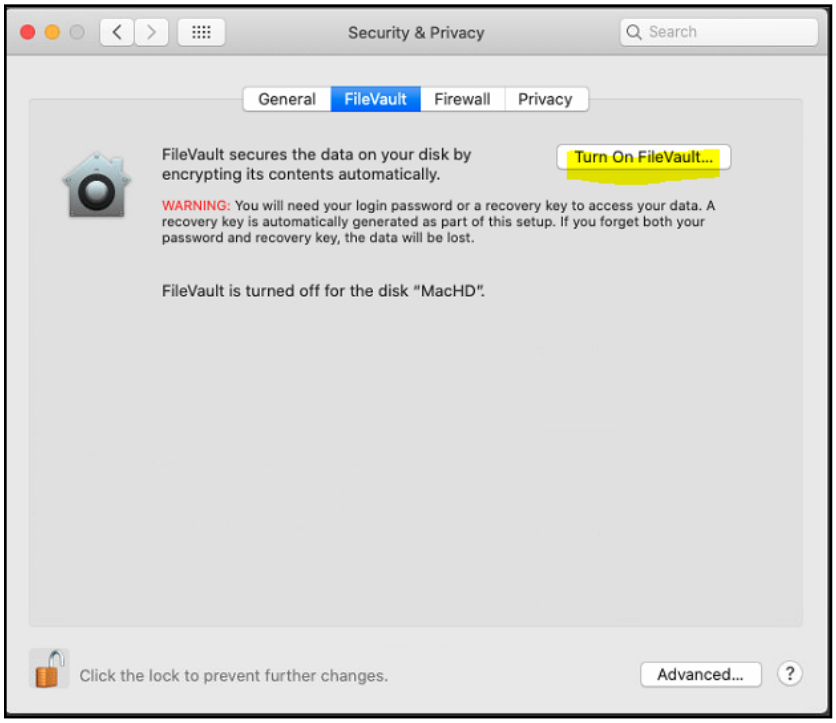
Note: Encryption of removable storage devices doesn't utilize the security capabilities of the Apple T2 Security Chip, and its encryption is performed in the same manner as Mac computers without the T2 chip.

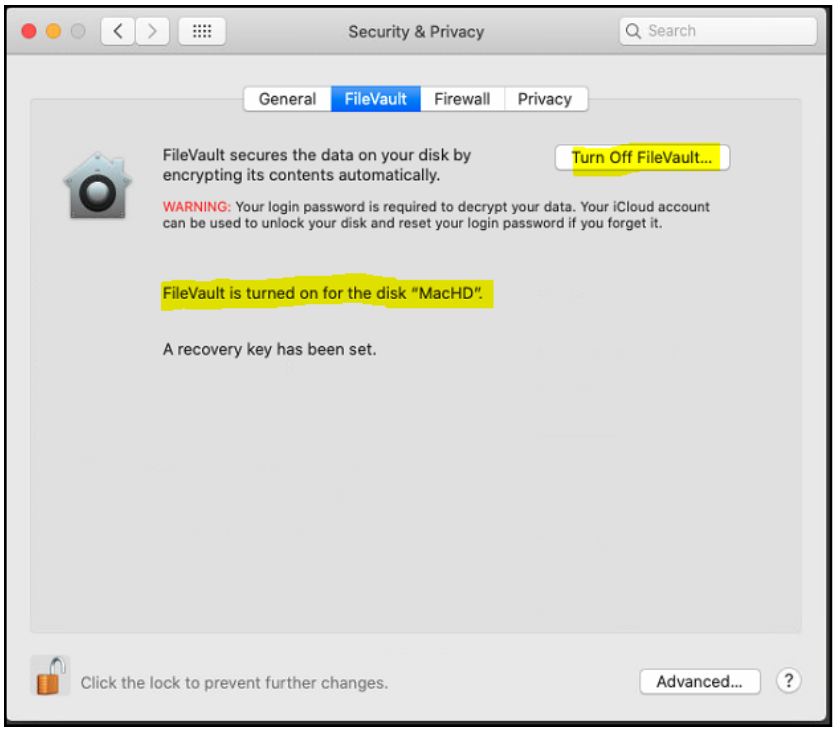
- Click on FileVault

- Click the lock to make changes and enter Administrator credentials.



- Click on Turn On FileVault



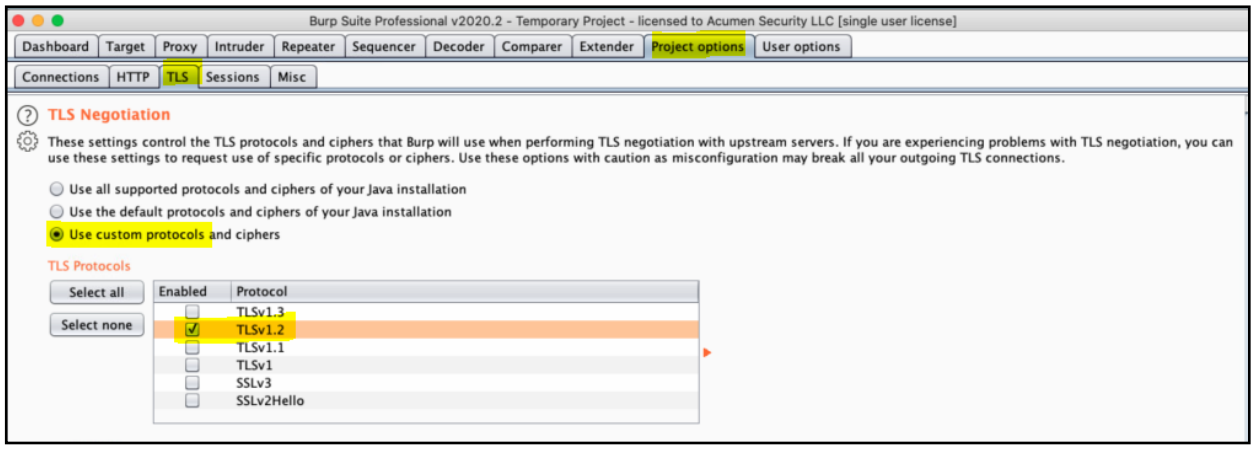


22. Appendix A

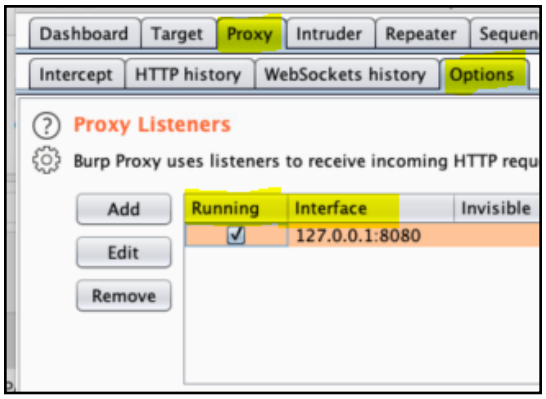
The user can check and download authentic OS updates and software application updates by visiting https://support.apple.com/downloads. The TOE defaults to using TLS v1.3 while checking for all updates. To comply with NIAP PP_OS_V4.2.1 requirements, the evaluator had to restrict the TOE TLS version to TLS v1.2.

To check for software updates over TLS v1.2, the user must configure a Proxy instance on their TOE as described below:

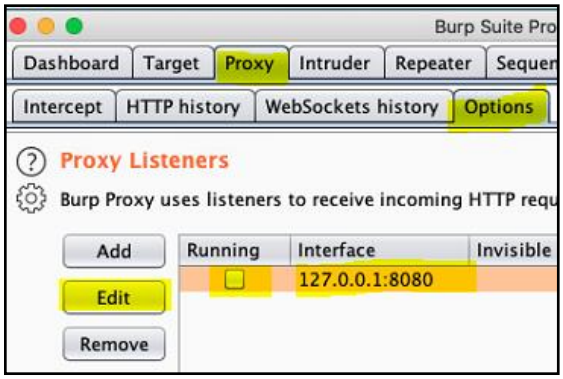
- Download and install BurpSuite Pro v2020.2 (referred to as Proxy in this document) on the TOE platform from https://portswigger.net/burp/pro.
- **Note: BurpSuite Pro is an example of a Web Proxy tool, however, it is not mandatory to use BurpSuite Pro.**
- After installing the Proxy on the TOE, start the Proxy and click on "Project Options" and click on "TLS" and then select "Use custom protocols and ciphers" and select "TLSv1.2" as shown below:



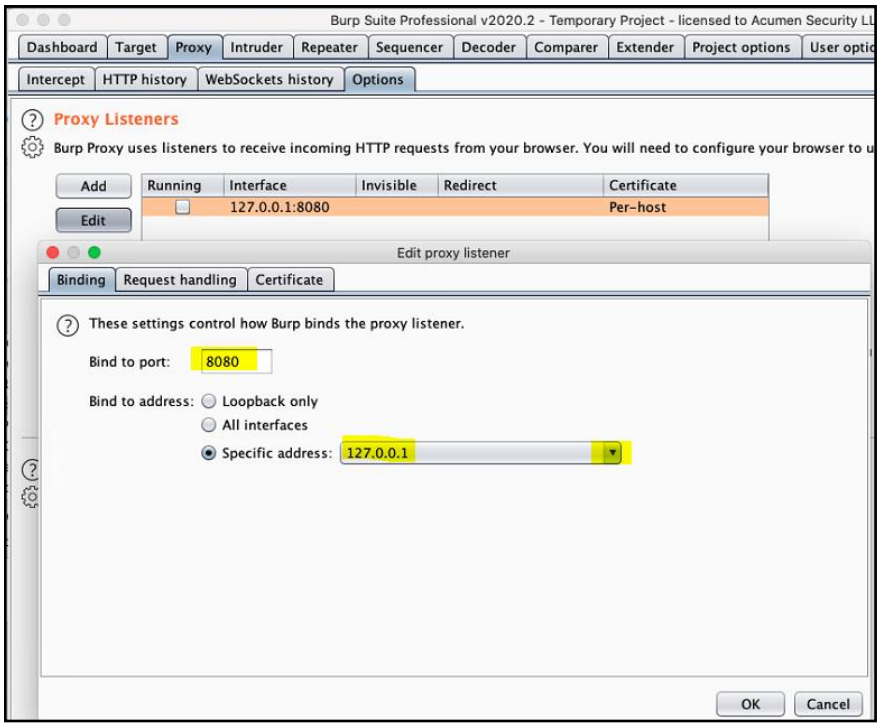
- Then click on "Proxy" and click on "Options" and ensure that the proxy is running. Since the proxy is installed on the same platform as the TOE, the Interface is selected to be "127.0.0.1:8080" or localhost. The IP address and/or the Port number can be changed according to the user requirements by clicking on "Edit", however, deselect "Running" prior to making changes to the IP and/or Port number.



- After deselecting "Running"

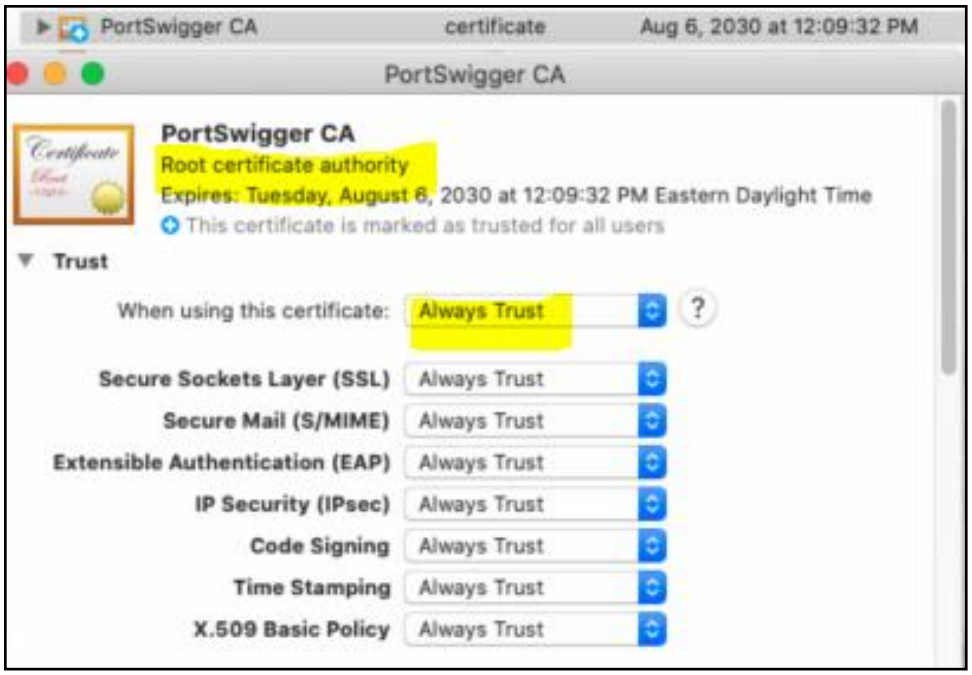


- The IP address and the Port number can be edited as per users' requirements:

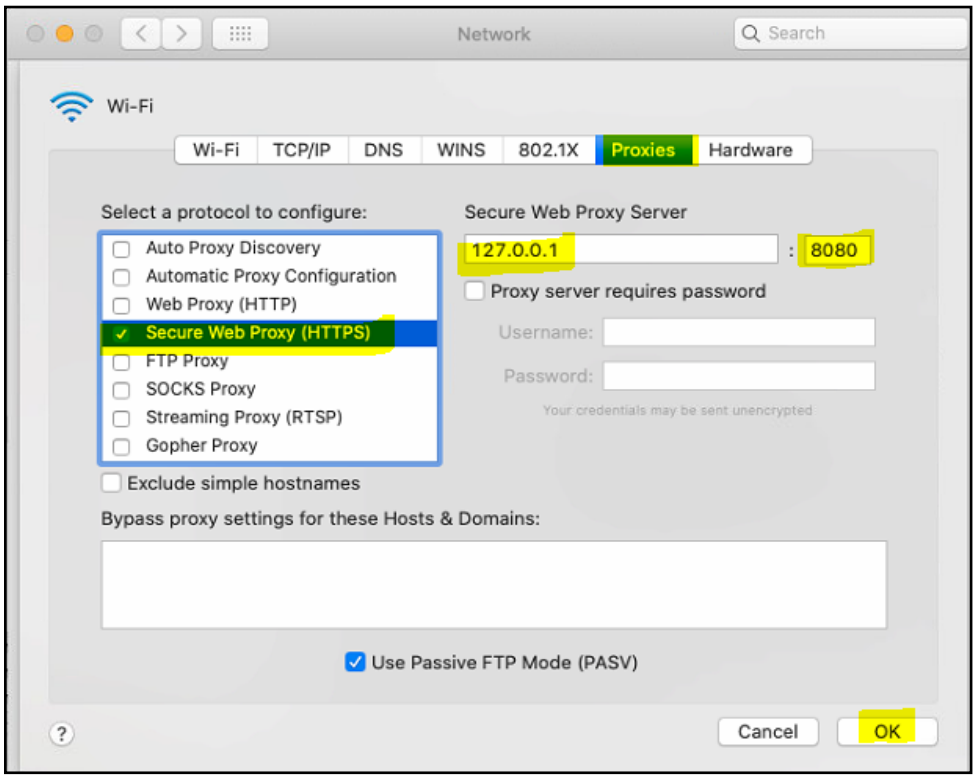


- Click on Import/ export CA certificate and save the certificate on the TOE.

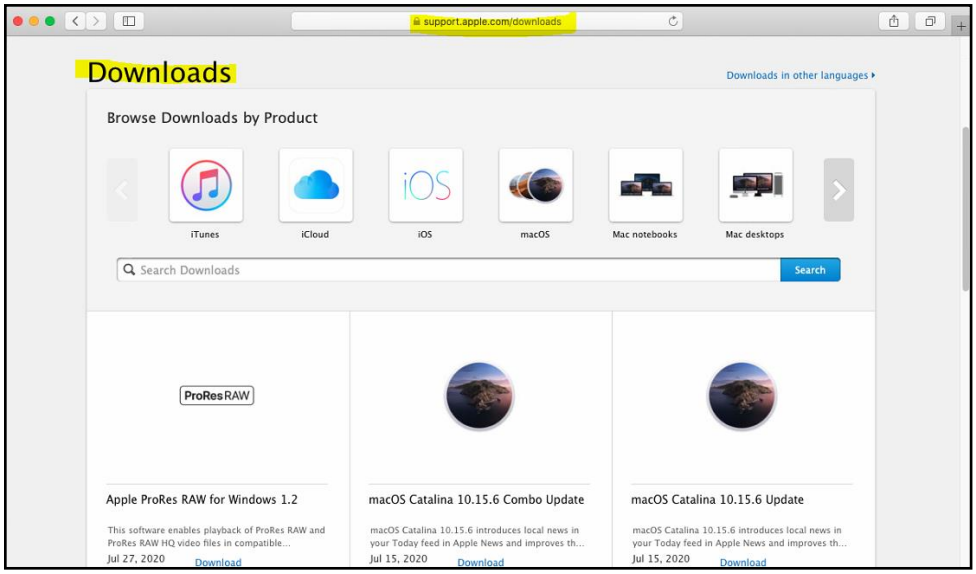
- Once exported, this proxy CA certificate must be installed on the TOE keychain before checking for any updates. Double click on the certificate and provide admin credentials to install the certificate on the TOE. Then right click on the certificate and mark as "Always Trust".
    - Note: The administrator can send the proxy CA certificate via email or host it on an internal web server and request the user to download the CA certificate from the web server and install the CA certificate on their system.



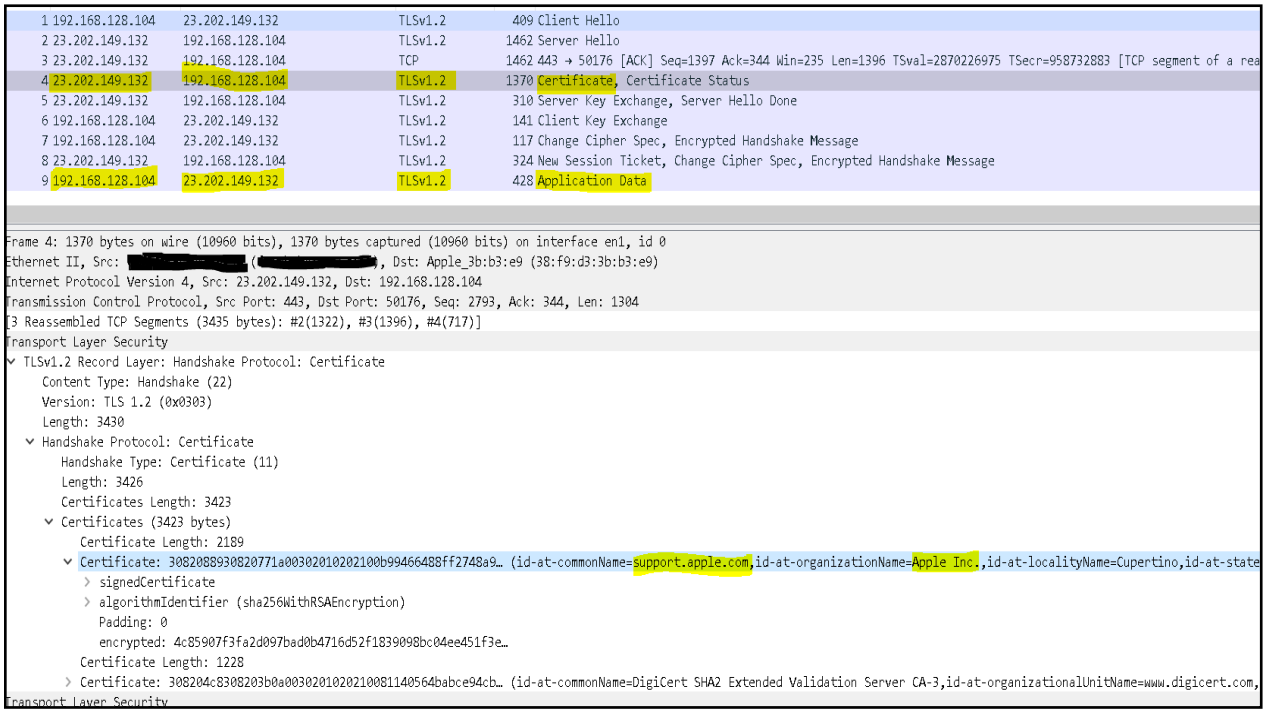
- Navigate to “System Preferences” -> “Network” -> “Advanced” -> “Proxies”.
    - Since the Proxy is running on localhost interface, enter 127.0.0.1 (localhost) and Port 8080 and click on “OK”. Then click on “Apply” so that the TOE will apply these settings system wide.



- Start Safari application and navigate to https://support.apple.com/downloads. After visiting the web URL all available OS and Software Application updates will be shown to the user. The user can download update(s) according to their requirements.



- The TOE successfully validates the Server certificate as shown in below screenshot:



END OF DOCUMENT