macOS Security Compliance

macOS 10.15

Security Configuration - CISV8

Version , Revision 6 (2022-03-16)

Table of Contents

Chapter 1. Foreword

The macOS Security Compliance Project is an open source effort to provide a programmatic approach to generating security guidance. The configuration settings in this document were derived from National Institute of Standards and Technology (NIST) Special Publication (SP) 800-53, *Security and Privacy Controls for Information Systems and Organizations*, Revision 5.

This project can be used as a resource to easily create customized security baselines of technical security controls by leveraging a library of atomic actions which are mapped to the compliance requirements defined in NIST SP 800-53 (Rev. 5). It can also be used to develop customized guidance to meet the particular cybersecurity needs of any organization.

The objective of this effort was to simplify and radically accelerate the process of producing up-to-date macOS security guidance that is also accessible to any organization and tailorable to meet each organization's specific security needs.

Any and all risk based decisions to tailor the content produced by this project in order to meet the needs of a specific organization shall be approved by the responsible Information System Owner (ISO) and Authorizing Official (AO) and formally documented in their System Security Plan (SSP). While the project attempts to provide settings to meet compliance requirements, it is recommended that each rule be reviewed by your organization's Information System Security Officer (ISSO) prior to implementation.

Chapter 2. Scope

This guide describes the actions to take when securing a macOS 10.15 system against the CISV8 baseline.

Chapter 3. Authors

Name	Organization

Chapter 4. Acronyms and Definitions

Table 1. Acronyms and Abbreviations

AES	Advanced Encryption Standard
ABM	Apple Business Manager
AFP	Apple Filing Protocol
ALF	Application Layer Firewall
AO	Authorizing Official
API	Application Programming Interface
ARD	Apple Remote Desktop
CA	Certificate Authority
CIS	Center for Internet Security
CRL	Certificate Revocation List
DISA	Defense Information Systems Agency
DMA	Direct Memory Access
FISMA	Federal Information Security Modernization Act
FPKI	Federal Public Key Infrastructure
IR	Infrared
ISO	Information System Owner
ISSO	Information System Security Officer
MDM	Mobile Device Management
NASA	National Aeronautics and Space Administration
NFS	Network File System
NIST	National Institute of Standards and Technology
NSA	National Security Agency
OCSP	Online Certificate Status Protocol
OS	Operating System
PF	Packet Filter
PIV	Personal Identity Verification
PIV-M	Personal Identity Verification Mandatory
PKI	Public Key Infrastructure
SIP	System Integrity Protection
SMB	Server Message Block
SSH	Secure Shell
SSP	System Security Plan

STIG	Security Technical Implementation Guide
UAMDM	User Approved MDM
UUCP	Unix-to-Unix Copy Protocol

Chapter 5. Applicable Documents

5.1. Government Documents

Table 2. National Institute of Standards and Technology (NIST)

Document Number or Descriptor	Document Title
NIST Special Publication 800-53 Rev 5	*NIST Special Publication 800-53 Rev 5*
NIST Special Publication 800-63	*NIST Special Publication 800-63*
NIST Special Publication 800-171	*NIST Special Publication 800-171 Rev 2*

Table 3. Defense Information Systems Agency (DISA)

Document Number or Descriptor	Document Title
STIG Ver 1, Rel 7	*Apple macOS 10.15 STIG*

Table 4. Committee on National Security Systems (CNSS)

Document Number or Descriptor	Document Title
CNSSI No. 1253	*Security Categorization and Control Selection for National Security Systems*

5.2. Non-Government Documents

Table 5. Apple

Document Number or Descriptor	Document Title
Apple Platform Security Guide	*Apple Platform Security*
Deployment Reference for Mac	*Deployment Reference*
Mobile Device Management Settings	*Mobile Device Management Settings*
Profile-Specific Payload Keys	*Profile-Specific Payload Keys*
Security Certifications and Compliance Center	*Security Certifications and Compliance Center*

Table 6. Center for Internet Security

Document Number or Descriptor	Document Title
Apple macOS 10.15	*CIS Apple macOS 10.15 Benchmark*

Chapter 6. Auditing

This section contains the configuration and enforcement of the OpenBSM settings.

The check/fix commands outlined in this section *MUST* be run with elevated privileges.

6.1. Enable Security Auditing

The information system *MUST* be configured to generate audit records.

Audit records establish what types of events have occurred, when they occurred, and which users were involved. These records aid an organization in their efforts to establish, correlate, and investigate the events leading up to an outage or attack.

The content required to be captured in an audit record varies based on the impact level of an organization's system. Content that may be necessary to satisfy this requirement includes, for example, time stamps, source addresses, destination addresses, user identifiers, event descriptions, success/fail indications, filenames involved, and access or flow control rules invoked.

The information system initiates session audits at system start-up.

Security auditing is enabled by default on macOS.

To check the state of the system, run the following command(s):

```
/bin/launchctl list | /usr/bin/grep -c com.apple.auditd
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl load -w /System/Library/LaunchDaemons/com.apple.auditd.plist
```

ID	audit_auditd_enabled

References	**800-53r5**	• AU-12, AU-12(1), AU-12(3) • AU-14(1) • AU-3, AU-3(1) • AU-8 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 8.2, 8.5
	CCE	• CCE-84706-1

6.2. Configure System to Audit All Authorization and Authentication Events

The auditing system *MUST* be configured to flag authorization and authentication (aa) events.

Authentication events contain information about the identity of a user, server, or client. Authorization events contain information about permissions, rights, and rules. If audit records do not include aa events, it is difficult to identify incidents and to correlate incidents to subsequent events.

Audit records can be generated from various components within the information system (e.g., via a module or policy filter).

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec 'aa'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*[^-]aa" /etc/security/audit_control || /usr/bin/sed -
i.bak '/^flags/ s/$/,aa/' /etc/security/audit_control; /usr/sbin/audit -s
```

ID	audit_flags_aa_configure

References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84711-1

6.3. Configure System to Audit All Administrative Action Events

The auditing system *MUST* be configured to flag administrative action (ad) events.

Administrative action events include changes made to the system (e.g. modifying authentication policies). If audit records do not include ad events, it is difficult to identify incidents and to correlate incidents to subsequent events.

Audit records can be generated from various components within the information system (e.g., via a module or policy filter).

The information system audits the execution of privileged functions.

We recommend changing the line "43127:AUE_MAC_SYSCALL:mac_syscall(2):ad" to "43127:AUE_MAC_SYSCALL:mac_syscall(2):zz" in the file /etc/security/audit_event. This will prevent sandbox violations from being audited by the ad flag.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec 'ad'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*[^-]ad" /etc/security/audit_control || /usr/bin/sed -
i.bak '/^flags/ s/$/,ad/' /etc/security/audit_control; /usr/sbin/audit -s
```

ID	audit_flags_ad_configure	
References	**800-53r5**	• AC-2(12), AC-2(4) • AC-6(9) • AU-12 • AU-2 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84712-9

6.4. Configure System to Audit All Failed Program Execution on the System

The audit system *MUST* be configured to record enforcement actions of access restrictions, including failed program execute (-ex) attempts.

Enforcement actions are the methods or mechanisms used to prevent unauthorized access and/or changes to configuration settings. One common and effective enforcement action method is using program execution restrictions (e.g., denying users access to execute certain processes).

This configuration ensures that audit lists include events in which program execution has failed. Without auditing the enforcement of program execution, it is difficult to identify attempted attacks, as there is no audit trail available for forensic investigation.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '\-ex'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*-ex" /etc/security/audit_control || /usr/bin/sed -i.bak
'/^flags/ s/$/,-ex/' /etc/security/audit_control; /usr/sbin/audit -s
```

ID	audit_flags_ex_configure	
References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • CM-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84913-3

6.5. Configure System to Audit All Deletions of Object Attributes

The audit system *MUST* be configured to record enforcement actions of attempts to delete file attributes (fd).

***Enforcement actions are the methods or mechanisms used to prevent unauthorized changes to configuration settings. One common and effective enforcement action method is using access restrictions (i.e., denying modifications to a file by applying file permissions).

This configuration ensures that audit lists include events in which enforcement actions prevent attempts to delete a file.

Without auditing the enforcement of access restrictions, it is difficult to identify attempted attacks, as there is no audit trail available for forensic investigation.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '\-fd'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*-fd" /etc/security/audit_control || /usr/bin/sed -i.bak
'/^flags/ s/$/,-fd/' /etc/security/audit_control;/usr/sbin/audit -s
```

ID	audit_flags_fd_configure	
References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • AU-9 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84922-4

6.6. Configure System to Audit All Failed Change of Object Attributes

The audit system *MUST* be configured to record enforcement actions of failed attempts to modify file attributes (fm).

Enforcement actions are the methods or mechanisms used to prevent unauthorized changes to configuration settings. One common and effective enforcement action method is using access restrictions (i.e., denying modifications to a file by applying file permissions).

This configuration ensures that audit lists include events in which enforcement actions prevent attempts to modify a file.

Without auditing the enforcement of access restrictions, it is difficult to identify attempted attacks, as there is no audit trail available for forensic investigation.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '\-fm'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*-fm" /etc/security/audit_control || /usr/bin/sed -i.bak
```

```
'/^flags/ s/$/,-fm/' /etc/security/audit_control;/usr/sbin/audit -s
```

ID	audit_flags_fm_configure	
References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • AU-9 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84715-2

6.7. Configure System to Audit All Failed Read Actions on the System

The audit system *MUST* be configured to record enforcement actions of access restrictions, including failed file read (-fr) attempts.

Enforcement actions are the methods or mechanisms used to prevent unauthorized access and/or changes to configuration settings. One common and effective enforcement action method is using access restrictions (e.g., denying access to a file by applying file permissions).

This configuration ensures that audit lists include events in which enforcement actions prevent attempts to read a file.

Without auditing the enforcement of access restrictions, it is difficult to identify attempted attacks, as there is no audit trail available for forensic investigation.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '\-fr'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*-fr" /etc/security/audit_control || /usr/bin/sed -i.bak
'/^flags/ s/$/,-fr/' /etc/security/audit_control;/usr/sbin/audit -s
```

ID	audit_flags_fr_configure	
References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • AU-9 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84713-7

6.8. Configure System to Audit All Failed Write Actions on the System

The audit system *MUST* be configured to record enforcement actions of access restrictions, including failed file write (-fw) attempts.

Enforcement actions are the methods or mechanisms used to prevent unauthorized access and/or changes to configuration settings. One common and effective enforcement action method is using access restrictions (e.g., denying users access to edit a file by applying file permissions).

This configuration ensures that audit lists include events in which enforcement actions prevent attempts to change a file.

Without auditing the enforcement of access restrictions, it is difficult to identify attempted attacks, as there is no audit trail available for forensic investigation.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '\-fw'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*-fw" /etc/security/audit_control || /usr/bin/sed -i.bak
'/^flags/ s/$/,-fw/' /etc/security/audit_control;/usr/sbin/audit -s
```

ID	audit_flags_fw_configure	
References	**800-53r5**	• AC-2(12) • AU-12 • AU-2 • AU-9 • CM-5(1) • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84714-5

6.9. Configure System to Audit All Log In and Log Out Events

The audit system *MUST* be configured to record all attempts to log in and out of the system (lo).

Frequently, an attacker that successfully gains access to a system has only gained access to an account with limited privileges, such as a guest account or a service account. The attacker must attempt to change to another user account with normal or elevated privileges in order to proceed. Auditing both successful and unsuccessful attempts to switch to another user account (by way of monitoring login and logout events) mitigates this risk.

The information system monitors login and logout events.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F':' '/^flags/ { print $NF }' /etc/security/audit_control | /usr/bin/tr
',' '\n' | /usr/bin/grep -Ec '^lo'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/grep -qE "^flags.*[^-]lo" /etc/security/audit_control || /usr/bin/sed -
i.bak '/^flags/ s/$/,lo/' /etc/security/audit_control; /usr/sbin/audit -s
```

ID	audit_flags_lo_configure	
References	**800-53r5**	• AC-17(1) • AC-2(12) • AU-12 • AU-2 • MA-4(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.14, 8.2, 8.5
	CCE	• CCE-84716-0

6.10. Configure Audit Retention to 7d

The audit service *MUST* be configured to require records be kept for a organizational defined value before deletion, unless the system uses a central audit record storage facility.

When "expire-after" is set to "7d", the audit service will not delete audit logs until the log data is at least seven days old.

To check the state of the system, run the following command(s):

```
/usr/bin/awk -F: '/expire-after/{print $2}' /etc/security/audit_control
```

If the result is not **7d**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/sed -i.bak 's/^expire-after.*/expire-after:7d/'
/etc/security/audit_control; /usr/sbin/audit -s
```

ID	audit_retention_configure

References	**800-53r5**	• AU-11 • AU-4
	CIS Benchmark	• N/A
	CIS Controls V8	• 8.3, 8.1
	CCE	• CCE-84719-4

Chapter 7. iCloud

This section contains the configuration and enforcement of iCloud and the Apple ID service settings.

The check/fix commands outlined in this section *MUST* be run by a user with with elevated privileges.

7.1. Disable iCloud Address Book

The macOS built-in Contacts.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data, and, therefore, automated contact synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudAddressBook').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudAddressBook</key>
<false/>
```

ID	icloud_addressbook_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84730-1

7.2. Disable the System Preference Pane for Apple ID

The system preference pane for Apple ID *MUST* be disabled.

Disabling the system preference pane prevents login to Apple ID and iCloud.

To check the state of the system, run the following command(s):

```
/usr/bin/profiles show -output stdout-xml | /usr/bin/xmllint --xpath
'string(//*[contains(text(), "DisabledPreferencePanes")]/following-sibling::*[1])' - |
/usr/bin/grep -c com.apple.preferences.AppleIDPrefPane
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.systempreferences) payload type:

```
<key>DisabledPreferencePanes</key>
<array>
    <string>com.apple.preferences.AppleIDPrefPane</string>
</array>
```

ID	icloud_appleid_prefpane_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84731-9

7.3. Disable iCloud Bookmarks

The macOS built-in Safari.app bookmark synchronization via the iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated bookmark synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudBookmarks').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudBookmarks</key>
<false/>
```

ID	icloud_bookmarks_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84732-7

7.4. Disable the iCloud Calendar Services

The macOS built-in Calendar.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated calendar synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/profiles -P -o stdout | /usr/bin/grep -c 'allowCloudCalendar = 0' |
/usr/bin/awk '{ if ($1 >= 1) {print "1"} else {print "0"}}'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudCalendar</key>
<false/>
```

ID	icloud_calendar_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84733-5

7.5. Disable iCloud Document Sync

The macOS built-in iCloud document synchronization service *MUST* be disabled to prevent organizational data from being synchronized to personal or non-approved storage.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated document synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudDocumentSync').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudDocumentSync</key>
<false/>
```

ID	icloud_drive_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84734-3

7.6. Disable iCloud Keychain Sync

The macOS system's ability to automatically synchronize a user's passwords to their iCloud account *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, password management and synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudKeychainSync').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudKeychainSync</key>
<false/>
```

ID	icloud_keychain_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84735-0

7.7. Disable iCloud Mail

The macOS built-in Mail.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated mail synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudMail').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudMail</key>
<false/>
```

ID	icloud_mail_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84736-8

7.8. Disable iCloud Notes

The macOS built-in Notes.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated Notes synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudNotes').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudNotes</key>
<false/>
```

ID	icloud_notes_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84737-6

7.9. Disable iCloud Photo Library

The macOS built-in Photos.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated photo synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudPhotoLibrary').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudPhotoLibrary</key>
<false/>
```

ID	icloud_photos_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84738-4

7.10. Disable iCloud Reminders

The macOS built-in Reminders.app connection to Apple's iCloud service *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated reminders synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudReminders').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudReminders</key>
<false/>
```

ID	icloud_reminders_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84739-2

7.11. Disable iCloud Desktop and Document Folder Sync

The macOS system's ability to automatically synchronize a user's desktop and documents folder to their iCloud Drive *MUST* be disabled.

Apple's iCloud service does not provide an organization with enough control over the storage and access of data and, therefore, automated file synchronization *MUST* be controlled by an organization approved service.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowCloudDesktopAndDocuments').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowCloudDesktopAndDocuments</key>
<false/>
```

ID	icloud_sync_disable

References	**800-53r5**	• AC-20, AC-20(1) • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84740-0

Chapter 8. macOS

This section contains the configuration and enforcement of operating system settings.

The check/fix commands outlined in this section *MUST* be run by a user with elevated privileges.

8.1. Disable AirDrop

AirDrop *MUST* be disabled to prevent file transfers to or from unauthorized devices. AirDrop allows users to share and receive files from other nearby Apple devices.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowAirDrop').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowAirDrop</key>
<false/>
```

ID	os_airdrop_disable	
References	**800-53r5**	• AC-20 • AC-3 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 6.7
	CCE	• CCE-84747-5

8.2. Enforce Apple Mobile File Integrity

Apple Mobile File Integrity (AMFI) is a macOS kernel module that enforces the code-signing validation within Gatekeeper and library validation. AMFI checks the signatures of every app that is run.

AMFI is enabled by default on macOS systems.

To check the state of the system, run the following command(s):

```
/usr/sbin/nvram -p | /usr/bin/grep -c "amfi_get_out_of_my_way=1"
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/sbin/nvram boot-args=""
```

ID	os_apple_mobile_file_integrity_enforce	
References	**800-53r5**	• SI-3 • SI-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.1, 10.4, 10.5
	CCE	• CCE-84926-5

8.3. Disable Apple ID Setup during Setup Assistant

The prompt for Apple ID setup during Setup Assistant *MUST* be disabled.

macOS will automatically prompt new users to set up an Apple ID while they are going through Setup Assistant if this is not disabled, misleading new users to think they need to create Apple ID accounts upon their first login.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SetupAssistant.managed')\
.objectForKey('SkipCloudSetup').js
```

```
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SetupAssistant.managed) payload type:

```
<key>SkipCloudSetup</key>
<true/>
```

ID	os_appleid_prompt_disable	
References	**800-53r5**	• AC-20
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84748-3

8.4. Disable Bonjour Multicast

Bonjour multicast advertising *MUST* be disabled to prevent the system from broadcasting its presence and available services over network interfaces.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.mDNSResponder')\
.objectForKey('NoMulticastAdvertisements').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mDNSResponder) payload type:

```
<key>NoMulticastAdvertisements</key>
<true/>
```

ID	os_bonjour_disable	
References	**800-53r5**	• CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84749-1

8.5. Disable Calendar.app

The macOS built-in Calendar.app *MUST* be disabled as this application can establish a connection to non-approved services. This rule is in place to prevent inadvertent data transfers.

Some organizations allow the use of the built-in Calendar.app for organizational communication. Information System Security Officers (ISSOs) may make the risk-based decision not to disable the macOS built-in Mail.app to avoid losing this functionality, but they are advised to first fully weigh the potential risks posed to their organization.

To check the state of the system, run the following command(s):

```
/usr/bin/profiles -P -o stdout | /usr/bin/grep -A 20 familyControlsEnabled |
/usr/bin/grep -c "/Applications/Calendar.app"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess.new) payload type:

```
<key>familyControlsEnabled</key>
<true/>
<key>pathBlackList</key>
<array>
    <string>/Applications/Calendar.app</string>
</array>
```

ID	os_calendar_app_disable	
References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84750-9

8.6. Enforce Installation of XProtect, MRT, and Gatekeeper Updates Automatically

Software Update *MUST* be configured to update XProtect, MRT, and Gatekeepr automatically.

This setting enforces definition updates for XProtect, MRT, and Gatekeeper; with this setting in place, new malware and adware that Apple has added to the list of malware or untrusted software will not execute. These updates do not require the computer to be restarted.

https://support.apple.com/en-us/HT207005

Software update will automatically update XProtect, MRT, and Gatekeeper by default in the macOS.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SoftwareUpdate')\
.objectForKey('ConfigDataInstall').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SoftwareUpdate) payload type:

```
<key>ConfigDataInstall</key>
<true/>
```

ID	os_config_data_install_enforce	
References	**800-53r5**	• SI-2(5) • SI-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.1, 10.2, 10.4
	CCE	• CCE-84929-9

8.7. Integrate System into a Directory Services Infrastructure

The macOS system *MUST* be integrated into a directory services infrastructure.

A directory service infrastructure enables centralized user and rights management, as well as centralized control over computer and user configurations. Integrating the macOS systems used throughout an organization into a directory services infrastructure ensures more administrator oversight and security than allowing distinct user account databases to exist on each separate system.

To check the state of the system, run the following command(s):

```
/usr/bin/dscl localhost -list . | /usr/bin/grep -vE '(Contact|Search|Local|^$)';
/bin/echo $?
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Integrate the system into an existing directory services infrastructure.

ID	os_directory_services_configured	
References	**800-53r5**	• N/A
	CIS Benchmark	• N/A
	CIS Controls V8	• 6.7
	CCE	• CCE-84951-3

8.8. Must Use ESS

The approved ESS solution *MUST* be installed and configured to run.

The macOS system must employ automated mechanisms to determine the state of system components. The DoD requires the installation and use of an approved ESS solution to be implemented on the operating system. For additional information, reference all applicable ESS OPORDs and FRAGOs on SIPRNET.

To check the state of the system, run the following command(s):

```
Ask the System Administrator (SA) or Information System Security Officer (ISSO) if the
approved ESS solution is loaded on the system.
If the installed components of the ESS solution are not at the DoD approved minimal
versions, this is a finding.
```

If the result is not **N/A**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Install the approved ESS solution onto the system.

ID	os_ess_installed	
References	**800-53r5**	• N/A
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.1, 10.2, 10.6, 10.7
	CCE	• CCE-84931-5

8.9. Disable FaceTime.app

The macOS built-in FaceTime.app *MUST* be disabled.

The FaceTime.app establishes a connection to Apple's iCloud service, even when security controls have been put in place to disable iCloud access.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
function run() {
  let pref1 = ObjC.unwrap(
```

```
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('familyControlsEnabled'))
  let pathlist =
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('pathBlackList').js
  for ( let app in pathlist ) {
      if ( ObjC.unwrap(pathlist[app]) == "/Applications/FaceTime.app" && pref1 == true
){
          return("true")
      }
  }
  return("false")
  }
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess.new) payload type:

```
<key>familyControlsEnabled</key>
<true/>
<key>pathBlackList</key>
<array>
    <string>/Applications/FaceTime.app</string>
</array>
```

ID	os_facetime_app_disable	
References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-85308-5

8.10. Enable Firewall Logging

Firewall logging *MUST* be enabled.

Firewall logging ensures that malicious network activity will be logged to the system.

The firewall data is logged to Apple's Unified Logging with the subsystem com.apple.alf and the data is marked as private.

To check the state of the system, run the following command(s):

```
/usr/libexec/ApplicationFirewall/socketfilterfw --getloggingmode | /usr/bin/grep -c
"Log mode is on"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/libexec/ApplicationFirewall/socketfilterfw --setloggingmode on
```

ID	os_firewall_log_enable	
References	**800-53r5**	• AU-12 • SC-7
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.5, 8.2, 8.5
	CCE	• CCE-84757-4

8.11. Enable Gatekeeper

Gatekeeper *MUST* be enabled.

Gatekeeper is a security feature that ensures that applications are digitally signed by an Apple-issued certificate before they are permitted to run. Digital signatures allow the macOS host to verify that the application has not been modified by a malicious third party.

Administrator users will still have the option to override these settings on a case-by-case basis.

To check the state of the system, run the following command(s):

```
/usr/sbin/spctl --status | /usr/bin/grep -c "assessments enabled"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.systempolicy.control) payload type:

```
<key>EnableAssessment</key>
<true/>
```

ID	os_gatekeeper_enable	
References	**800-53r5**	• CM-14 • CM-5 • SI-3 • SI-7(1), SI-7(15)
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.1, 10.2, 10.5
	CCE	• CCE-84759-0

8.12. Enforce Gatekeeper 30 Day Automatic Rearm

Gatekeeper *MUST* be configured to automatically rearm after 30 days if disabled.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.security')\
.objectForKey('GKAutoRearm').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

The following settings are in the (com.apple.ManagedClient.preferences) payload. This payload requires the additional settings to be sub-payloads within, containing their defined payload types.

Create a configuration profile containing the following keys in the (com.apple.security) payload type:

```
<key>GKAutoRearm</key>
<true/>
```

ID	os_gatekeeper_rearm	
References	**800-53r5**	• CM-5
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.5
	CCE	• CCE-84852-3

8.13. Disable Handoff

Handoff *MUST* be disabled.

Handoff allows you to continue working on a document or project when the user switches from one Apple device to another. Disabling Handoff prevents data transfers to unauthorized devices.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowActivityContinuation').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowActivityContinuation</key>
<false/>
```

ID	os_handoff_disable

References	**800-53r5**	• AC-20 • AC-3 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84763-2

8.14. Disable the Built-in Web Server

The built-in web server is a non-essential service built into macOS and *MUST* be disabled.

The built in web server service is disabled at startup by default macOS.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"org.apache.httpd" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/org.apache.httpd
```

ID	os_httpd_disable	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 6.7
	CCE	• CCE-84765-7

8.15. Disable iCloud Storage Setup during Setup Assistant

The prompt to set up iCloud storage services during Setup Assistant *MUST* be disabled.

The default behavior of macOS is to prompt new users to set up storage in iCloud. Disabling the iCloud storage setup prompt provides organizations more control over the storage of their data.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SetupAssistant.managed')\
.objectForKey('SkipiCloudStorageSetup').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SetupAssistant.managed) payload type:

```
<key>SkipiCloudStorageSetup</key>
<true/>
```

ID	os_icloud_storage_prompt_disable	
References	**800-53r5**	• AC-20
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84766-5

8.16. Disable the Internet Accounts System Preference Pane

The Internet Accounts System Preference pane *MUST* be disabled to prevent the addition of unauthorized internet accounts.

Some organizations may allow the use and configuration of the built-in Mail.app,

> Calendar.app, and Contacts.app for organizational communication. Information System Security Officers (ISSOs) may make the risk-based decision not to disable the Internet Accounts System Preference pane to avoid losing this functionality, but they are advised to first fully weigh the potential risks posed to their organization.

To check the state of the system, run the following command(s):

```
/usr/bin/profiles show -output stdout-xml | /usr/bin/xmllint --xpath
'string(//*[contains(text(), "DisabledPreferencePanes")]/following-sibling::*[1])' - |
/usr/bin/grep -c com.apple.preferences.internetaccounts
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.systempreferences) payload type:

```
<key>DisabledPreferencePanes</key>
<array>
    <string>com.apple.preferences.internetaccounts</string>
</array>
```

ID	os_internet_accounts_prefpane_disable	
References	**800-53r5**	• AC-20 • CM-7(5)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.8, 15.2
	CCE	• CCE-84767-3

8.17. Disable Infrared (IR) support

Infrared (IR) support *MUST* be disabled to prevent users from controlling the system with IR devices.

By default, if IR is enabled, the system will accept IR control from any remote device.

This is applicable only to models of Mac Mini systems earlier than Mac Mini8,1.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.driver.AppleIRController')\
.objectForKey('DeviceEnabled').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

The following settings are in the (com.apple.ManagedClient.preferences) payload. This payload requires the additional settings to be sub-payloads within, containing their defined payload types.

Create a configuration profile containing the following keys in the (com.apple.driver.AppleIRController) payload type:

```
<key>DeviceEnabled</key>
<false/>
```

ID	os_ir_support_disable	
References	**800-53r5**	• AC-18 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 12.6
	CCE	• CCE-84768-1

8.18. Disable Mail App

The macOS built-in Mail.app *MUST* be disabled.

The Mail.app contains functionality that can establish connections to Apple's iCloud, even when security controls to disable iCloud access have been put in place.

Some organizations allow the use of the built-in Mail.app for organizational

> communication. Information System Security Officers (ISSOs) may make the risk-based decision not to disable the macOS built-in Mail.app to avoid losing this functionality, but they are advised to first fully weigh the potential risks posed to their organization.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
function run() {
  let pref1 = ObjC.unwrap(
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('familyControlsEnabled'))
  let pathlist =
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('pathBlackList').js
  for ( let app in pathlist ) {
      if ( ObjC.unwrap(pathlist[app]) == "/Applications/Mail.app" && pref1 == true ){
          return("true")
      }
  }
  return("false")
  }
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess.new) payload type:

```
<key>familyControlsEnabled</key>
<true/>
<key>pathBlackList</key>
<array>
    <string>/Applications/Mail.app</string>
</array>
```

ID	os_mail_app_disable

References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84769-9

8.19. Enforce Enrollment in Mobile Device Management

You *MUST* enroll your Mac in a Mobile Device Management (MDM) software.

User Approved MDM (UAMDM) enrollment or enrollment via Apple Business Manager (ABM)/Apple School Manager (ASM) is required to manage certain security settings. Currently these include: * Whitelisting Approved Kernel Extensions * Privacy Preferences Policy Control Payload * ExtensibleSingleSignOn

To check the state of the system, run the following command(s):

```
/usr/bin/profiles status -type enrollment | /usr/bin/awk -F: '/MDM enrollment/ {print $2}' | /usr/bin/grep -c "Yes (User Approved)"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Ensure that system is enrolled via UAMDM.

ID	os_mdm_require	
References	**800-53r5**	• CM-2 • CM-6
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 5.1
	CCE	• CCE-84803-6

8.20. Disable Messages App

The macOS built-in Messages.app *MUST* be disabled.

The Messages.app establishes a connection to Apple's iCloud service, even when security controls to disable iCloud access have been put in place.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
function run() {
  let pref1 = ObjC.unwrap(
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('familyControlsEnabled'))
  let pathlist =
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
  .objectForKey('pathBlackList').js
  for ( let app in pathlist ) {
      if ( ObjC.unwrap(pathlist[app]) == "/Applications/Messages.app" && pref1 == true
){
          return("true")
      }
  }
  return("false")
  }
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess.new) payload type:

```
<key>familyControlsEnabled</key>
<true/>
<key>pathBlackList</key>
<array>
    <string>/Applications/Messages.app</string>
</array>
```

ID	os_messages_app_disable

References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84770-7

8.21. Disable Network File System Service

Support for Network File Systems (NFS) services is non-essential and, therefore, *MUST* be disabled.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.nfsd" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.nfsd
```

The system may need to be restarted for the update to take effect.

ID	os_nfsd_disable	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 6.7
	CCE	• CCE-84772-3

8.22. Enable Parental Controls

Parental Controls *MUST* be enabled.

Control of program execution is a mechanism used to prevent program execution of unauthorized

programs, which is critical to maintaining a secure system baseline.

Parental Controls on the macOS consist of many different payloads, which are set individually depending on the type of control required. Enabling parental controls allows for further configuration of these restrictions.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess.new')\
.objectForKey('familyControlsEnabled').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess.new) payload type:

```
<key>familyControlsEnabled</key>
<true/>
```

ID	os_parental_controls_enable	
References	**800-53r5**	• CM-7(2)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.8
	CCE	• CCE-84773-1

8.23. Disable Password Autofill

Password Autofill *MUST* be disabled.

macOS allows users to save passwords and use the Password Autofill feature in Safari and compatible apps. To protect against malicious users gaining access to the system, this feature *MUST* be disabled to prevent users from being prompted to save passwords in applications.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowPasswordAutoFill').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowPasswordAutoFill</key>
<false/>
```

ID	os_password_autofill_disable	
References	**800-53r5**	• CM-7, CM-7(1) • IA-11 • IA-5, IA-5(13)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84774-9

8.24. Disable Proximity Based Password Sharing Requests

Proximity based password sharing requests *MUST* be disabled.

The default behavior of macOS is to allow users to request passwords from other known devices (macOS and iOS). This feature *MUST* be disabled to prevent passwords from being shared.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowPasswordProximityRequests').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowPasswordProximityRequests</key>
<false/>
```

ID	os_password_proximity_disable	
References	**800-53r5**	• IA-5
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84775-6

8.25. Disable Password Sharing

Password Sharing *MUST* be disabled.

The default behavior of macOS is to allow users to share a password over Airdrop between other macOS and iOS devices. This feature *MUST* be disabled to prevent passwords from being shared.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowPasswordSharing').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowPasswordSharing</key>
<false/>
```

ID	os_password_sharing_disable	
References	**800-53r5**	• IA-5
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84776-4

8.26. Disable Privacy Setup Services During Setup Assistant

The prompt for Privacy Setup services during Setup Assistant *MUST* be disabled.

Organizations *MUST* apply organization-wide configuration settings. The macOS Privacy Setup services prompt guides new users through enabling their own specific privacy settings; this is not essential and, therefore, *MUST* be disabled to prevent against the risk of individuals electing privacy settings with the potential to override organization-wide settings.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SetupAssistant.managed')\
.objectForKey('SkipPrivacySetup').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SetupAssistant.managed) payload type:

```
<key>SkipPrivacySetup</key>
<true/>
```

ID	os_privacy_setup_prompt_disable	
References	**800-53r5**	• CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84781-4

8.27. Disable Siri Setup during Setup Assistant

The prompt for Siri during Setup Assistant *MUST* be disabled.

Organizations *MUST* apply organization-wide configuration settings. The macOS Siri Assistant Setup prompt guides new users through enabling their own specific Siri settings; this is not essential and, therefore, *MUST* be disabled to prevent against the risk of individuals electing Siri settings with the potential to override organization-wide settings.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SetupAssistant.managed')\
.objectForKey('SkipSiriSetup').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SetupAssistant.managed) payload type:

```
<key>SkipSiriSetup</key>
<true/>
```

ID	os_siri_prompt_disable

References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84791-3

8.28. Disable Trivial File Tansfer Protocol Service

If the system does not require Trivial File Tansfer Protocol (TFTP), support it is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling TFTP helps prevent the unauthorized connection of devices and the unauthorized transfer of information.

TFTP service is disabled at startup by default macOS.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.tftpd" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.tftpd
```

The system may need to be restarted for the update to take effect.

ID	os_tftpd_disable

References	**800-53r5**	• AC-17 • AC-3 • IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 3.1, 5.2
	CCE	• CCE-84853-1

8.29. Enable Time Synchronization Daemon

The macOS time synchronization daemon (timed) *MUST* be enabled for proper time synchronization to an authorized time server.

To check the state of the system, run the following command(s):

```
/bin/launchctl list | /usr/bin/grep -c com.apple.timed
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl load -w /System/Library/LaunchDaemons/com.apple.timed.plist
```

ID	os_time_server_enabled	
References	**800-53r5**	• AU-12(1) • SC-45(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 8.4
	CCE	• CCE-84801-0

8.30. Disable TouchID Prompt during Setup Assistant

The prompt for TouchID during Setup Assistant *MUST* be disabled.

macOS prompts new users through enabling TouchID during Setup Assistant; this is not essential and, therefore, *MUST* be disabled to prevent against the risk of individuals electing to enable TouchID to override organization-wide settings.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SetupAssistant.managed')\
.objectForKey('SkipTouchIDSetup').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SetupAssistant.managed) payload type:

```
<key>SkipTouchIDSetup</key>
<true/>
```

ID	os_touchid_prompt_disable	
References	**800-53r5**	• CM-6
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1
	CCE	• CCE-84802-8

8.31. Disable Unix-to-Unix Copy Protocol Service

The system *MUST* not have the Unix-to-Unix Copy Protocol (UUCP) service active.

UUCP, a set of programs that enable the sending of files between different UNIX systems as well as sending commands to be executed on another system, is not essential and *MUST* be disabled in order to prevent the unauthorized connection of devices, transfer of information, and tunneling.

UUCP service is disabled at startup by default macOS.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.uucp" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.uucp
```

The system may need to be restarted for the update to take effect.

ID	os_uucp_disable	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 4.1, 4.8
	CCE	• CCE-84806-9

Chapter 9. Password Policy

This section contains the configuration and enforcement of settings pertaining to password policies in macOS.

The check/fix commands outlined in this section *MUST* be run by a user with elevated privileges.

The password policy recommendations in the NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

The settings outlined in this section adhere to the recommendations provided in this document for systems that utilize passwords for local accounts. If systems are integrated with a directory service, local password policies should align with domain password policies to the fullest extent feasible.

9.1. Disable Accounts after 35 Days of Inactivity

The macOS *MUST* be configured to disable accounts after 35 days of inactivity.

This rule prevents malicious users from making use of unused accounts to gain access to the system while avoiding detection.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath '//dict/key[text()="policyAttributeInactiveDays"]/following-
sibling::integer[1]/text()' -
```

If the result is not **35**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

This setting may be enforced using local policy or by a directory service.

To set local policy to disable an inactive user after 35 days, edit the current password policy to contain the following <dict> within the "policyCategoryAuthentication":

```
<dict>
<key>policyContent</key>
<string>policyAttributeLastAuthenticationTime &gt; policyAttributeCurrentTime -
(policyAttributeInactiveDays * 24 * 60 * 60)</string>
```

```
<key>policyIdentifier</key>
<string>Inactive Account</string>
<key>policyParameters</key>
<dict>
<key>policyAttributeInactiveDays<key>
<integer>35</integer>
</dict>
</dict>
```

After saving the file and exiting to the command prompt, run the following command to load the new policy file, substituting the path to the file in place of "$pwpolicy_file".

```
/usr/bin/pwpolicy setaccountpolicies $pwpolicy_file
```

See the password policy supplemental on more information on how to implement password policies on macOS.

ID	pwpolicy_account_inactivity_enforce	
References	**800-53r5**	• AC-2(3)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.3
	CCE	• CCE-84808-5

9.2. Limit Consecutive Failed Login Attempts to 3

The macOS *MUST* be configured to limit the number of failed login attempts to a maximum of 3. When the maximum number of failed attempts is reached, the account *MUST* be locked for a period of time after.

This rule protects against malicious users attempting to gain access to the system via brute-force hacking methods.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath
'//dict/key[text()="policyAttributeMaximumFailedAuthentications"]/following-
sibling::integer[1]/text()' - | /usr/bin/awk '{ if ($1 <= 3) {print "yes"} else {print
"no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>maxFailedAttempts</key>
<integer>3</integer>
```

ID	pwpolicy_account_lockout_enforce	
References	**800-53r5**	• AC-7
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1
	CCE	• CCE-84809-3

9.3. Set Account Lockout Time to 15 Minutes

The macOS *MUST* be configured to enforce a lockout time period of at least 15 minutes when the maximum number of failed logon attempts is reached.

This rule protects against malicious users attempting to gain access to the system via brute-force hacking methods.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath '//dict/key[text()="autoEnableInSeconds"]/following-
sibling::integer[1]/text()' - | /usr/bin/awk '{ if ($1/60 >= 15 ) {print "yes"} else
{print "no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>minutesUntilFailedLoginReset</key>
```

```
<integer>15</integer>
```

ID	pwpolicy_account_lockout_timeout_enforce	
References	**800-53r5**	• AC-7
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1
	CCE	• CCE-84810-1

9.4. Require Passwords Contain a Minimum of One Numeric Character

The macOS *MUST* be configured to require at least one numeric character be used when a password is created.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath '//dict/key[text()="policyIdentifier"]/following-
sibling::*[1]/text()' - | /usr/bin/grep "requireAlphanumeric" -c
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>requireAlphanumeric</key>
<true/>
```

ID	pwpolicy_alpha_numeric_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84811-9

9.5. Prohibit Password Reuse for a Minimum of 5 Generations

The macOS *MUST* be configured to enforce a password history of at least 5 previous passwords when a password is created.

This rule ensures that users are not allowed to re-use a password that was used in any of the 5 previous password generations.

Limiting password reuse protects against malicious users attempting to gain access to the system via brute-force hacking methods.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath
'//dict/key[text()="policyAttributePasswordHistoryDepth"]/following-
sibling::*[1]/text()' - | /usr/bin/awk '{ if ($1 >= 5 ) {print "yes"} else {print
"no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>pinHistory</key>
```

```
<integer>5</integer>
```

ID	pwpolicy_history_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84814-3

9.6. Require Passwords Contain a Minimum of One Lowercase Character

The macOS *MUST* be configured to require at least one lower-case character be used when a password is created.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath
'//dict/key[text()="minimumAlphaCharactersLowerCase"]/following-
sibling::integer[1]/text()' - | /usr/bin/awk '{ if ($1 >= 1 ) {print "yes"} else
{print "no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

This setting may be enforced using local policy or by a directory service.

To set local policy to require at least 1 lowercase letter, edit the current password policy to contain the following <dict> within the "policyCategoryPasswordContent":

```
<dict>
<key>policyContent</key>
<string>policyAttributePassword matches &apos;(.*[a-z].*){1,}+&apos;</string>
<key>policyIdentifier</key>
<string>Must have at least 1 lowercase letter</string>
<key>policyParameters</key>
<dict>
<key>minimumAlphaCharactersLowerCase</key>
<integer>1</integer>
</dict>
</dict>
```

After saving the file and exiting to the command prompt, run the following command to load the new policy file, substituting the path to the file in place of "$pwpolicy_file".

```
/usr/bin/pwpolicy setaccountpolicies $pwpolicy_file
```

See the password policy supplemental on more information on how to implement password policies on macOS.

ID	pwpolicy_lower_case_character_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84815-0

9.7. Restrict Maximum Password Lifetime to 60 Days

The macOS *MUST* be configured to enforce a maximum password lifetime limit of at least 60 days.

This rule ensures that users are forced to change their passwords frequently enough to prevent malicious users from gaining and maintaining access to the system.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
```

```
/usr/bin/xmllint --xpath
'//dict/key[text()="policyAttributeExpiresEveryNDays"]/following-sibling::*[1]/text()'
-
```

If the result is not **60**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>maxPINAgeInDays</key>
<integer>60</integer>
```

ID	pwpolicy_max_lifetime_enforce	
References	**800-53r5**	• IA-5
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.7
	CCE	• CCE-84807-7

9.8. Require a Minimum Password Length of 15 Characters

The macOS *MUST* be configured to require a minimum of 15 characters be used when a password is created.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath 'boolean(//*[contains(text(),"policyAttributePassword matches
```

```
'\''.{15,}'\''")])' -
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>minLength</key>
<integer>15</integer>
```

ID	pwpolicy_minimum_length_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84816-8

9.9. Set Minimum Password Lifetime to 24 Hours

The macOS *MUST* be configured to enforce a minimum password lifetime limit of 24 hours.

This rule discourages users from cycling through their previous passwords to get back to a preferred one.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath
'//dict/key[text()="policyAttributeMinimumLifetimeHours"]/following-
sibling::integer[1]/text()' - | /usr/bin/awk '{ if ($1 >= 24 ) {print "yes"} else
{print "no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

This setting may be enforced using local policy or by a directory service.

To set local policy to require a minimum password lifetime, edit the current password policy to contain the following <dict> within the "policyCategoryPasswordContent":

```
<dict>
<key>policyContent</key>
<string>policyAttributeLastPasswordChangeTime &lt; policyAttributeCurrentTime -
(policyAttributeMinimumLifetimeHours * 60 * 60)</string>
<key>policyIdentifier</key>
<string>Minimum Password Lifetime</string>
<key>policyParameters</key>
<dict>
<key>policyAttributeMinimumLifetimeHours</key>
<integer>24</integer>
</dict>
</dict>
```

After saving the file and exiting to the command prompt, run the following command to load the new policy file, substituting the path to the file in place of "$pwpolicy_file".

```
/usr/bin/pwpolicy setaccountpolicies $pwpolicy_file
```

See the password policy supplemental on more information on how to implement password policies on macOS.

ID	pwpolicy_minimum_lifetime_enforce	
References	**800-53r5**	• IA-5
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.7
	CCE	• CCE-84817-6

9.10. Prohibit Repeating, Ascending, and Descending Character Sequences

The macOS *MUST* be configured to prohibit the use of repeating, ascending, and descending

character sequences when a password is created.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath '//dict/key[text()="policyIdentifier"]/following-
sibling::*[1]/text()' - | /usr/bin/grep "allowSimple" -c
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>allowSimple</key>
<false/>
```

ID	pwpolicy_simple_sequence_disable	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84818-4

9.11. Require Passwords Contain a Minimum of One Special Character

The macOS *MUST* be configured to require at least one special character be used when a password is created.

Special characters are those characters that are not alphanumeric. Examples include: ~ ! @ # $ % ^

*.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath 'boolean(//*[contains(text(),"policyAttributePassword matches
'\''(.*[^a-zA-Z0-9].*){1,}'\''")])' -
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.mobiledevice.passwordpolicy) payload type:

```
<key>minComplexChars</key>
<integer>1</integer>
```

ID	pwpolicy_special_character_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84819-2

9.12. Require Passwords Contain a Minimum of One Uppercase Character

The macOS *MUST* be configured to require at least one uppercase character be used when a password is created.

This rule enforces password complexity by requiring users to set passwords that are less vulnerable

to malicious users.

The guidance for password based authentication in NIST 800-53 (Rev 5) and NIST 800-63B state that complexity rules should be organizationally defined. The values defined are based off of common complexity values. But your organization may define its own password complexity rules.

To check the state of the system, run the following command(s):

```
/usr/bin/pwpolicy -getaccountpolicies 2> /dev/null | /usr/bin/tail +2 |
/usr/bin/xmllint --xpath
'//dict/key[text()="minimumAlphaCharactersUpperCase"]/following-
sibling::integer[1]/text()' - | /usr/bin/awk '{ if ($1 >= 1 ) {print "yes"} else
{print "no"}}'
```

If the result is not **yes**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

This setting may be enforced using local policy or by a directory service.

To set local policy to require at least 1 lowercase letter, edit the current password policy to contain the following <dict> within the "policyCategoryPasswordContent":

```
<dict>
<key>policyContent</key>
<string>policyAttributePassword matches &apos;(.*[A-Z].*){1,}+&apos;</string>
<key>policyIdentifier</key>
<string>Must have at least 1 uppercase letter</string>
<key>policyParameters</key>
<dict>
<key>minimumAlphaCharactersUpperCase</key>
<integer>1</integer>
</dict>
</dict>
```

After saving the file and exiting to the command prompt, run the following command to load the new policy file, substituting the path to the file in place of "$pwpolicy_file".

```
/usr/bin/pwpolicy setaccountpolicies $pwpolicy_file
```

See the password policy supplemental on more information on how to implement password policies on macOS.

ID	pwpolicy_upper_case_character_enforce	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2
	CCE	• CCE-84821-8

Chapter 10. System Preferences

This section contains the configuration and enforcement of the settings within the macOS System Preferences application.

The check/fix commands outlined in this section *MUST* be run by a user with elevated privileges.

10.1. Disable Ad Tracking

Ad tracking and targeted ads *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling ad tracking ensures that applications and advertisers are unable to track users' interests and deliver targeted advertisements.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.AdLib')\
.objectForKey('forceLimitAdTracking').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.AdLib) payload type:

```
<key>forceLimitAdTracking</key>
<true/>
```

ID	sysprefs_ad_tracking_disable

References	**800-53r5**	• AC-20 • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84822-6

10.2. Disable Apple Filing Protocol Sharing

If the system does not require Apple Filing Protocol (AFP) Sharing, support it is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling AFP helps prevent the unauthorized connection of devices and the unauthorized transfer of information.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.AppleFileServer"
=> true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.AppleFileServer
```

The system may need to be restarted for the update to take effect.

ID	sysprefs_afp_disable

References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84823-4

10.3. Disable Bluetooth When no Approved Device is Connected

The macOS system *MUST* be configured to disable Bluetooth unless there is an approved device connected.

Information System Security Officers (ISSOs) may make the risk-based decision not to disable Bluetooth, so as to maintain necessary functionality, but they are advised to first fully weigh the potential risks posed to their organization.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.MCXBluetooth')\
.objectForKey('DisableBluetooth').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

The following settings are in the (com.apple.ManagedClient.preferences) payload. This payload requires the additional settings to be sub-payloads within, containing their defined payload types.

Create a configuration profile containing the following keys in the (com.apple.MCXBluetooth) payload type:

```
<key>DisableBluetooth</key>
<true/>
```

ID	sysprefs_bluetooth_disable	
References	**800-53r5**	• AC-18, AC-18(3) • SC-8
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.8, 12.6, 13.9
	CCE	• CCE-84826-7

10.4. Disable Bluetooth Sharing

Bluetooth Sharing *MUST* be disabled.

Bluetooth Sharing allows users to wirelessly transmit files between the macOS and Bluetooth-enabled devices, including personally owned cellphones and tablets. A malicious user might introduce viruses or malware onto the system or extract sensitive files via Bluetooth Sharing. When Bluetooth Sharing is disabled, this risk is mitigated.

The check and fix are for the currently logged in user. To get the currently logged in user, run the following.

```
CURRENT_USER=$( /usr/sbin/scutil <<< "show State:/Users/ConsoleUser" \|
/usr/bin/awk '/Name :/ && ! /loginwindow/ { print $3 }' )
```

To check the state of the system, run the following command(s):

```
/usr/bin/sudo -u "$CURRENT_USER" /usr/bin/defaults -currentHost read
com.apple.Bluetooth PrefKeyServicesEnabled
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/sudo -u "$CURRENT_USER" /usr/bin/defaults -currentHost write
com.apple.Bluetooth PrefKeyServicesEnabled -bool false
```

ID	sysprefs_bluetooth_sharing_disable

References	**800-53r5**	• AC-18(4) • AC-3 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 4.1, 4.8
	CCE	• CCE-84827-5

10.5. Disable Content Caching Service

Content caching *MUST* be disabled.

Content caching is a macOS service that helps reduce Internet data usage and speed up software installation on Mac computers. It is not recommended for devices furnished to employees to act as a caching server.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowContentCaching').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowContentCaching</key>
<false/>
```

ID	sysprefs_content_caching_disable

References	**800-53r5**	• CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84828-3

10.6. Enforce Critical Security Updates to be Installed

Ensure that security updates are installed as soon as they are available from Apple.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.SoftwareUpdate')\
.objectForKey('CriticalUpdateInstall').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SoftwareUpdate) payload type:

```
<key>CriticalUpdateInstall</key>
<true/>
```

ID	sysprefs_critical_update_install_enforce	
References	**800-53r5**	• SI-2
	CIS Benchmark	• N/A
	CIS Controls V8	• 7.3, 7.4, 7.7
	CCE	• CCE-84936-4

10.7. Disable Sending Diagnostic and Usage Data to Apple

The ability to submit diagnostic data to Apple *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling the submission of diagnostic and usage information will mitigate the risk of unwanted data being sent to Apple.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
function run() {
let pref1 = $.NSUserDefaults.alloc.initWithSuiteName('com.apple.SubmitDiagInfo')\
.objectForKey('AutoSubmit').js
let pref2 = $.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowDiagnosticSubmission').js
if ( pref1 == false && pref2 == false ){
    return("true")
} else {
    return("false")
}
}
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.SubmitDiagInfo) payload type:

```
<key>AutoSubmit</key>
<false/>
```

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowDiagnosticSubmission</key>
<false/>
```

ID	sysprefs_diagnostics_reports_disable

References	**800-53r5**	• AC-20 • SC-7(10) • SI-11
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84829-1

10.8. Enforce FileVault

FileVault *MUST* be enforced.

The information system implements cryptographic mechanisms to protect the confidentiality and integrity of information stored on digital media during transport outside of controlled areas.

To check the state of the system, run the following command(s):

```
/usr/bin/fdesetup status | /usr/bin/grep -c "FileVault is On."
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

See the FileVault supplemental to implement this rule.

ID	sysprefs_filevault_enforce	
References	**800-53r5**	• SC-28, SC-28(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.6, 3.11
	CCE	• CCE-84830-9

10.9. Disable Find My Service

The Find My service *MUST* be disabled.

A Mobile Device Management (MDM) solution *MUST* be used to carry out remote locking and

wiping instead of Apple's Find My service.

Apple's Find My service uses a personal AppleID for authentication. Organizations should rely on MDM solutions, which have much more secure authentication requirements, to perform remote lock and remote wipe.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
function run() {
  let pref1 = ObjC.unwrap(
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowFindMyDevice'))
  let pref2 = ObjC.unwrap(
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.applicationaccess')\
.objectForKey('allowFindMyFriends'))
  let pref3 = ObjC.unwrap(
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.icloud.managed')\
.objectForKey('DisableFMMiCloudSetting'))
  if ( pref1 == false && pref2 == false && pref3 == true ) {
    return("true")
  } else {
    return("false")
  }
}
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.applicationaccess) payload type:

```
<key>allowFindMyDevice</key>
<false/>
<key>allowFindMyFriends</key>
<false/>
```

Create a configuration profile containing the following keys in the (com.apple.icloud.managed) payload type:

```
<key>DisableFMMiCloudSetting</key>
<true/>
```

ID	sysprefs_find_my_disable	
References	**800-53r5**	• AC-20 • CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8, 15.3
	CCE	• CCE-84831-7

10.10. Enable macOS Application Firewall

The macOS Application Firewall is the built-in firewall that comes with macOS, and it *MUST* be enabled.

When the macOS Application Firewall is enabled, the flow of information within the information system and between interconnected systems will be controlled by approved authorizations.

To check the state of the system, run the following command(s):

```
/usr/libexec/ApplicationFirewall/socketfilterfw --getglobalstate | /usr/bin/grep -c
"Firewall is enabled"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/libexec/ApplicationFirewall/socketfilterfw --setglobalstate on
```

ID	sysprefs_firewall_enable	
References	**800-53r5**	• AC-4 • CM-7, CM-7(1) • SC-7, SC-7(12)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.5, 13.1
	CCE	• CCE-84832-5

10.11. Enable Firewall Stealth Mode

Firewall Stealth Mode *MUST* be enabled.

When stealth mode is enabled, the Mac will not respond to any probing requests, and only requests from authorized applications will still be authorized.

Enabling firewall stealth mode may prevent certain remote mechanisms used for maintenance and compliance scanning from properly functioning. Information System Security Officers (ISSOs) are advised to first fully weigh the potential risks posed to their organization before opting not to enable stealth mode.

To check the state of the system, run the following command(s):

```
/usr/libexec/ApplicationFirewall/socketfilterfw --getstealthmode | /usr/bin/grep -c
"Stealth mode enabled"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/libexec/ApplicationFirewall/socketfilterfw --setstealthmode on
```

ID	sysprefs_firewall_stealth_mode_enable	
References	**800-53r5**	• CM-7, CM-7(1) • SC-7, SC-7(16)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.5, 4.8
	CCE	• CCE-84833-3

10.12. Disable Guest Access to Shared Apple File Protocol Folders

Guest access to shared Apple File Protocol (AFP) folders *MUST* be disabled.

Turning off guest access prevents anonymous users from accessing files shared via AFP.

To check the state of the system, run the following command(s):

```
/usr/bin/profiles -P -o stdout | /usr/bin/grep -c 'guestAccess = 0' | /usr/bin/awk '{
if ($1 >= 1) {print "1"} else {print "0"}}'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.AppleFileServer) payload type:

```
<key>guestAccess</key>
<false/>
```

ID	sysprefs_guest_access_afp_disable	
References	**800-53r5**	• AC-2, AC-2(9)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2, 6.2, 6.8
	CCE	• CCE-84760-8

10.13. Disable Guest Access to Shared SMB Folders

Guest access to shared Server Message Block (SMB) folders *MUST* be disabled.

Turning off guest access prevents anonymous users from accessing files shared via SMB.

To check the state of the system, run the following command(s):

```
/usr/bin/defaults read /Library/Preferences/SystemConfiguration/com.apple.smb.server
AllowGuestAccess
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.smb.server) payload type:

```
<key>AllowGuestAccess</key>
<false/>
```

ID	sysprefs_guest_access_smb_disable	
References	**800-53r5**	• AC-2, AC-2(9)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2, 6.2, 6.8
	CCE	• CCE-84761-6

10.14. Disable the Guest Account

Guest access *MUST* be disabled.

Turning off guest access prevents anonymous users from accessing files.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.MCX')\
.objectForKey('DisableGuestAccount').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.MCX) payload type:

```
<key>DisableGuestAccount</key>
<true/>
```

ID	sysprefs_guest_account_disable

References	**800-53r5**	• AC-2, AC-2(9)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2, 6.2, 6.8
	CCE	• CCE-84939-8

10.15. Disable Sending Siri and Dictation Information to Apple

The ability for Apple to store and review audio of your Siri and Dictation interactions *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling the submission of Siri and Dictation information will mitigate the risk of unwanted data being sent to Apple.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.assistant.support')\
.objectForKey('Siri Data Sharing Opt-In Status').js
EOS
```

If the result is not **2**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.assistant.support) payload type:

```
<key>Siri Data Sharing Opt-In Status</key>
<integer>2</integer>
```

ID	sysprefs_improve_siri_dictation_disable

References	**800-53r5**	• AC-20 • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84912-5

10.16. Disable Internet Sharing

If the system does not require Internet sharing, support for it is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling Internet sharing helps prevent the unauthorized connection of devices, unauthorized transfer of information, and unauthorized tunneling.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.MCX')\
.objectForKey('forceInternetSharingOff').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.MCX) payload type:

```
<key>forceInternetSharingOff</key>
<true/>
```

ID	sysprefs_internet_sharing_disable

References	**800-53r5**	• AC-20 • AC-4
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84837-4

10.17. Disable Location Services

Location Services *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling Location Services helps prevent the unauthorized connection of devices, unauthorized transfer of information, and unauthorized tunneling.

To check the state of the system, run the following command(s):

```
/usr/bin/defaults read
/var/db/locationd/Library/Preferences/ByHost/com.apple.locationd.plist
LocationServicesEnabled
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/defaults write
/var/db/locationd/Library/Preferences/ByHost/com.apple.locationd
LocationServicesEnabled -bool false; /bin/launchctl kickstart -k
system/com.apple.locationd
```

ID	sysprefs_location_services_disable

References	**800-53r5**	• CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84838-2

10.18. Disable Media Sharing

Media sharing *MUST* be disabled.

When Media Sharing is enabled, the computer starts a network listening service that shares the contents of the user's music collection with other users in the same subnet.

The information system *MUST* be configured to provide only essential capabilities. Disabling Media Sharing helps prevent the unauthorized connection of devices and the unauthorized transfer of information. Disabling Media Sharing mitigates this risk.

The check and fix are for the currently logged in user. To get the currently logged in user, run the following.

```
CURRENT_USER=$( scutil <<< "show State:/Users/ConsoleUser" \| awk
'/Name :/ && ! /loginwindow/ { print $3 }' )
```

To check the state of the system, run the following command(s):

```
/usr/bin/sudo -u "$CURRENT_USER" /usr/bin/defaults read com.apple.amp.mediasharingd |
/usr/bin/grep -Ec '("public-sharing-enabled" = 0;|"home-sharing-enabled" = 0;)'
```

If the result is not **2**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/sudo -u "$CURRENT_USER" /usr/bin/defaults write
com.apple.amp.mediasharingd public-sharing-enabled -int 0
/usr/bin/sudo -u "$CURRENT_USER" /usr/bin/defaults write
com.apple.amp.mediasharingd home-sharing-enabled -int 0
/usr/bin/pkill -9 AMPLibraryAgent
/usr/bin/pkill -9 mediasharingd
```

ID	sysprefs_media_sharing_disabled	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84771-5

10.19. Disable Power Nap

Power Nap *MUST* be disabled.

Power Nap allows your Mac to perform actions while a Mac is asleep. This can interfere with USB power and may cause devices to stop functioning until a reboot and must therefore be disabled on all applicable systems.

The following Macs support Power Nap:

- MacBook (Early 2015 and later)
- MacBook Air (Late 2010 and later)
- MacBook Pro (all models with Retina display)
- Mac mini (Late 2012 and later)
- iMac (Late 2012 and later)
- Mac Pro (Late 2013 and later)

To check the state of the system, run the following command(s):

```
/usr/bin/pmset -g custom | /usr/bin/awk '/powernap/ { sum+=$2 } END {print sum}'
```

If the result is not **0**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/bin/pmset -a powernap 0
```

ID	sysprefs_power_nap_disable

References	**800-53r5**	• CM-7, CM-7(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84780-6

10.20. Disable Remote Apple Events

If the system does not require Remote Apple Events, support for Apple Remote Events is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling Remote Apple Events helps prevent the unauthorized connection of devices, the unauthorized transfer of information, and unauthorized tunneling.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.AEServer" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/usr/sbin/systemsetup -setremoteappleevents off
/bin/launchctl disable system/com.apple.AEServer
```

Systemsetup with -setremoteappleevents flag will fail unless you grant Full Disk Access to systemsetup or it's parent process. Requires UAMDM.

ID	sysprefs_rae_disable	
References	**800-53r5**	• AC-17
		• AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84841-6

10.21. Disable Screen Sharing and Apple Remote Desktop

Support for both Screen Sharing and Apple Remote Desktop (ARD) is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities. Disabling screen sharing and ARD helps prevent the unauthorized connection of devices, the unauthorized transfer of information, and unauthorized tunneling.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.screensharing" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.screensharing
```

NOTE - This will apply to the whole system

ID	sysprefs_screen_sharing_disable	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84842-4

10.22. Enforce Screen Saver Timeout

The screen saver timeout *MUST* be set to 1200 minutes.

This rule ensures that a full session lock is triggered after 1200 minutes of inactivity.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.screensaver')\
.objectForKey('idleTime').js
EOS
```

If the result is not **1200**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.screensaver) payload type:

```
<key>idleTime</key>
<integer>1200</integer>
```

ID	sysprefs_screensaver_timeout_enforce	
References	**800-53r5**	• AC-11 • IA-11
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.3
	CCE	• CCE-84788-9

10.23. Disable Siri

Support for Siri is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.ironwood.support')\
.objectForKey('Ironwood Allowed').js
EOS
```

If the result is not **false**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.ironwood.support) payload type:

```
<key>Ironwood Allowed</key>
<false/>
```

ID	sysprefs_siri_disable	
References	**800-53r5**	• AC-20 • CM-7, CM-7(1) • SC-7(10)
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84843-2

10.24. Disable Server Message Block Sharing

Support for Server Message Block (SMB) file sharing is non-essential and *MUST* be disabled.

The information system *MUST* be configured to provide only essential capabilities.

To check the state of the system, run the following command(s):

```
/bin/launchctl print-disabled system | /usr/bin/grep -c '"com.apple.smbd" => true'
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

```
/bin/launchctl disable system/com.apple.smbd
```

The system may need to be restarted for the update to take effect.

ID	sysprefs_smbd_disable	
References	**800-53r5**	• AC-17 • AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1, 4.8
	CCE	• CCE-84844-0

10.25. Configure macOS to Use an Authorized Time Server

Approved time servers *MUST* be the only servers configured for use.

This rule ensures the uniformity of time stamps for information systems with multiple system clocks and systems connected over a network.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.MCX')\
.objectForKey('timeServer').js
EOS
```

If the result is not **time-a.nist.gov,time-b.nist.gov**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

Create a configuration profile containing the following keys in the (com.apple.MCX) payload type:

```
<key>timeServer</key>
<string>time-a.nist.gov,time-b.nist.gov</string>
```

ID	sysprefs_time_server_configure

References	**800-53r5**	• AU-12(1) • SC-45(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 8.4
	CCE	• CCE-84846-5

10.26. Enable macOS Time Synchronization Daemon (timed)

The timed service *MUST* be enabled on all networked systems and configured to set time automatically from the approved time server.

This rule ensures the uniformity of time stamps for information systems with multiple system clocks and systems connected over a network.

To check the state of the system, run the following command(s):

```
/usr/bin/osascript -l JavaScript << EOS
$.NSUserDefaults.alloc.initWithSuiteName('com.apple.timed')\
.objectForKey('TMAutomaticTimeOnlyEnabled').js
EOS
```

If the result is not **true**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

The following settings are in the (com.apple.ManagedClient.preferences) payload. This payload requires the additional settings to be sub-payloads within, containing their defined payload types.

Create a configuration profile containing the following keys in the (com.apple.timed) payload type:

```
<key>TMAutomaticTimeOnlyEnabled</key>
<true/>
```

ID	sysprefs_time_server_enforce

References	**800-53r5**	• AU-12(1) • SC-45(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 8.4
	CCE	• CCE-84847-3

10.27. Disable Wi-Fi Interface

The macOS system must be configured with Wi-Fi support software disabled if not connected to an authorized trusted network.

Allowing devices and users to connect to or from the system without first authenticating them allows untrusted access and can lead to a compromise or attack. Since wireless communications can be intercepted it is necessary to use encryption to protect the confidentiality of information in transit.Wireless technologies include for example microwave packet radio (UHF/VHF) 802.11x and Bluetooth. Wireless networks use authentication protocols (e.g. EAP/TLS PEAP) which provide credential protection and mutual authentication.

If the system requires Wi-Fi to connect to an authorized network, this is not applicable.

To check the state of the system, run the following command(s):

```
/usr/sbin/networksetup -listallnetworkservices | /usr/bin/grep -c "*Wi-Fi"
```

If the result is not **1**, this is a finding.

Remediation Description

Perform the following to configure the system to meet the requirements:

To disable Wi-Fi on a macOS system, run the following command.

```
/usr/sbin/networksetup -setnetworkserviceenabled "Wi-Fi" off
```

ID	sysprefs_wifi_disable

References	**800-53r5**	• AC-18, AC-18(1), AC-18(3) • AC-4
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.2, 12.6
	CCE	• CCE-84938-0

Chapter 11. Inherent

This section reviews the controls that are built-in to macOS, and cannot be configured out of compliance.

11.1. Enforce Approved Authorization for Logical Access

The information system *IS* configured to enforce an approved authorization process before granting users logical access.

The inherent configuration of the macOS does not grant users logical access without authorization. Authorization is achieved on the macOS through permissions, which are controlled at many levels, from the Mach and BSD components of the kernel, through higher levels of the operating system and, for networked applications, through the networking protocols. Permissions can be granted at the level of directories, subdirectories, files or applications, or specific data within files or functions within applications.

https://developer.apple.com/library/archive/documentation/Security/Conceptual/AuthenticationAndAuthorizationGuide/Permissions/Permissions.html

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_logical_access	
References	**800-53r5**	• AC-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.3, 6.7

11.2. Ensure the System Implements Malicious Code Protection Mechanisms

The inherent configuration of the macOS *IS* in compliance as Apple has designed the system with three layers of protection against malware. Each layer of protection is comprised of one or more malicious code protection mechanisms, which are automatically implemented and which, collectively, meet the requirements of all applicable federal laws, Executive Orders, directives, policies, regulations, standards, and guidance for malicious code prevention.

1. This first layer of defense targets the distribution of malware; the aim is to prevent malware from ever launching. The following mechanisms are inherent to the macOS design and constitute the first layer of protection against malicious code:
    - The Apple App Store: the safest way to add new applications to a Mac is by downloading

them from the App Store; all apps available for download from the App Store have been reviewed for signs of tampering and signed by Apple to indicate that the app meets security requirements and does not contain malware.

- XProtect: a built-in, signature-based, anti-virus, anti-malware technology inherent to all Macs. XProtect automatically detects and blocks the execution of known malware.
- In macOS 10.15 and all subsequent releases, XProtect checks for known malicious content when:
- an app is first launched,
- an app has been changed (in the file system), and
- XProtect signatures are updated.
- YARA: another built-in tool (inherent to all Macs), which conducts signature-based detection of malware. Apple updates YARA rules regularly.
- Gatekeeper: a security feature inherent to all Macs; Gatekeeper scans apps to detect malware and/or revocations of a developer's signing certificate and prevents unsafe apps from running.
- Notarization: Apple performs regular, automated scans to detect signs of malicious content and to verify developer ID-signed software; when no issues are found, Apple notarizes the software and delivers the results of scans to the system owner.

2. The second layer of defense targets malware that manages to appear on a Mac before it runs; the aim is to quickly identify and block any malware present on a Mac in order to prevent the malware from running and further spreading. The following mechanisms are inherent to the macOS design and constitute the second layer of protection against malicious code:
   - XProtect (defined above).
   - Gatekeeper (defined above).
   - Notarization (defined above).
3. The third layer of defense targets infected Mac system(s); the aim is to remediate Macs on which malware has managed to successfully execute. The following mechanism is inherent to the macOS design and constitutes the third layer of protection against malicious code:
   - Apple's Malware Removal Tool (MRT): a technology included on all macOS systems. MRT is an agent that remediates based on automatic updates delivered from Apple. MRT will remove the malware upon receiving updated information and check for malware on restart and login.

https://support.apple.com/guide/security/protecting-against-malware-sec469d47bd8/1/web/1

https://support.apple.com/guide/security/app-security-overview-sec35dd877d0/web

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_malicious_code_prevention

References	**800-53r5**	• SI-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 10.1, 10.2, 10.5

11.3. Enforce multifactor authentication for network access to privileged accounts

The information system implements multifactor authentication for network access to privileged accounts.

For directory bound systems: The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_mfa_network_access	
References	**800-53r5**	• N/A
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.6

11.4. Obscure Passwords

The information system *IS* configured to obscure feedback of authentication information during the authentication process to protect the information from possible exploitation by unauthorized individuals.

The inherent configuration of a macOS uses NSSecureTextField for any text field that receives a password, which automatically obscures text which is entered.

https://developer.apple.com/documentation/appkit/nssecuretextfield

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_obscure_password

References	**800-53r5**	• IA-5 • IA-6
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.1

11.5. Encrypt Stored Passwords

The information system *IS* configured to encrypt stored passwords.

Passwords need to be protected at all times, and encryption is the standard method for protecting passwords. If passwords are not encrypted, they can be plainly read (i.e., clear text) and easily compromised.

https://developer.apple.com/documentation/opendirectory/kodattributetypeauthenticationauthority

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_store_encrypted_passwords	
References	**800-53r5**	• IA-5(1), IA-5(1)(c)
	CIS Benchmark	• N/A
	CIS Controls V8	• 3.11

11.6. Uniquely Identify Users and Processes

The macOS is a UNIX 03-compliant operating system. The system uniquely identifies and authenticates organizational users or processes.

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	os_unique_identification	
References	**800-53r5**	• IA-4
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.1, 6.1

11.7. Force Password Change at Next Logon

The macOS is able to be configured to force users to change their password at next logon.

Temporary passwords are often used for new users when accounts are created. However, once logged in to the system, users must be immediately prompted to change to a permanent password of their creation.

To for a user to change their password at next logon, run the following command:

```
/usr/bin/pwpolicy -u [USER] -setpolicy "newPasswordRequired=1"
```

Replace [USER] with the username that must change the password at next logon

The technology supports this requirement and cannot be configured to be out of compliance. The technology inherently meets this requirement.

ID	pwpolicy_force_password_change	
References	**800-53r5**	• IA-5(1)
	CIS Benchmark	• N/A
	CIS Controls V8	• 5.2

Chapter 12. Permanent Findings

This section contains the controls that are defined in NIST 800-53 revision 5 but are unable to be configured natively within macOS. It is recommended to implement a third-party solution to meet the controls in this section.

12.1. Must authenticate peripherals before establishing a connection

Organizational devices requiring unique device-to-device identification and authentication may be defined by type, by device, or by a combination of type/device. Information systems typically use either shared known information (e.g., Media Access Control [MAC] or Transmission Control Protocol/Internet Protocol [TCP/IP] addresses) for device identification or organizational authentication solutions (e.g., IEEE 802.1x and Extensible Authentication Protocol [EAP], Radius server with EAP-Transport Layer Security [TLS] authentication, Kerberos) to identify/authenticate devices on local and/or wide area networks. Organizations determine the required strength of authentication mechanisms by the security categories of information systems. Because of the challenges of applying this control on large scale, organizations are encouraged to only apply the control to those limited number (and type) of devices that truly need to support this capability.

The technology does support this requirement, however, third party solutions are required to implement at an infrastructure level.

ID	os_auth_peripherals	
References	**800-53r5**	• IA-3
	CIS Benchmark	• N/A
	CIS Controls V8	• 13.9

12.2. Secure Name Address Resolution Service

The information system requests and performs data origin authentication and data integrity verification on the name/address resolution responses the system receives from authoritative sources.

The technology does not support this requirement. This is an applicable-does not meet finding.

ID	os_secure_name_resolution

References	**800-53r5**	• SC-21
	CIS Benchmark	• N/A
	CIS Controls V8	• 4.9

Chapter 13. Not Applicable

This section contains the controls that are defined in the NIST 800-53 revision 5 but are not applicable when configuring a macOS system.

13.1. Access Control for Mobile Devices

A mobile device is a computing device that has a small form factor such that it can easily be carried by a single individual; is designed to operate without a physical connection; possesses local, non-removable or removable data storage; and includes a self-contained power source. Mobile device functionality may also include voice communication capabilities, on-board sensors that allow the device to capture information, and/or built-in features for synchronizing local data with remote locations. Examples include smart phones and tablets. Mobile devices are typically associated with a single individual. The processing, storage, and transmission capability of the mobile device may be comparable to or merely a subset of notebook/desktop systems, depending on the nature and intended purpose of the device. Protection and control of mobile devices is behavior or policy-based and requires users to take physical action to protect and control such devices when outside of controlled areas. Controlled areas are spaces for which organizations provide physical or procedural controls to meet the requirements established for protecting information and systems.

Due to the large variety of mobile devices with different characteristics and capabilities, organizational restrictions may vary for the different classes or types of such devices. Usage restrictions and specific implementation guidance for mobile devices include configuration management, device identification and authentication, implementation of mandatory protective software, scanning devices for malicious code, updating virus protection software, scanning for critical software updates and patches, conducting primary operating system (and possibly other resident software) integrity checks, and disabling unnecessary hardware.

Usage restrictions and authorization to connect may vary among organizational systems. For example, the organization may authorize the connection of mobile devices to its network and impose a set of usage restrictions, while a system owner may withhold authorization for mobile device connection to specific applications or impose additional usage restrictions before allowing mobile device connections to a system.

The technology does not support this requirement. This is an applicable-does not meet finding.

ID	os_access_control_mobile_devices	
References	**800-53r5**	• AC-19
	CIS Benchmark	• N/A
	CIS Controls V8	• 6.4

Chapter 14. Supplemental

This section provides additional information to support the guidance provided by the baselines.

14.1. Out of Scope Supplemental

There are several requirements defined in National Institute of Standards and Technology (NIST) Special Publication (SP) 800-53, Security and Privacy Controls for Information Systems and Organizations, Revision 5 that can be met by making configuration changes to the operating system. However, NIST SP 800-53 (Rev. 5) contains a broad set of guidelines that attempt to address all aspects of an information system or systems within an organization. Because the macOS Security Compliance Project is tailored specifically to macOS, some requirements defined in NIST SP 800-53 (Rev. 5) are not applicable.

This supplemental contains those controls that are assigned to a baseline in NIST SP 800-53 (Rev. 5) which cannot be addressed with a technical configuration for macOS. These controls can be accomplished though administrative or procedural processes within an organization or via integration of the macOS system into enterprise information systems which are configured to protect the systems within.

Family	Access Control (AC)
Controls	AC-1, AC-2, AC-3(14), AC-14, AC-17(4), AC-22

Family	Awareness and Training (AT)
Controls	AT-1, AT-2, AT-3, AT-4

Family	Audit and Accountability (AU)
Controls	AU-1, AU-6, AU-9(2)

Family	Security Assessment and Authorization (CA)
Controls	CA-1, CA-2, CA-3, CA-3(6), CA-5, CA-6, CA-7, CA-7(4), CA-9

Family	Configuration Management (CM)
Controls	CM-1, CM-4, CM-8, CM-10, CM-11

Family	Contingency Planning (CP)
Controls	CP-1, CP-2, CP-3, CP-4, CP-9, CP-10

Family	Identification and Authentication (IA)
Controls	IA-1, IA-8(1), IA-8(2), IA-8(3), IA-8(4)

Family	Incident Response (IR)
Controls	IR-1, IR-2, IR-4, IR-5, IR-6, IR-7, IR-8

Family	Maintenance (MA)
Controls	MA-1, MA-2, MA-5

Family	Media Protection (MP)
Controls	MP-1, MP-2, MP-6, MP-7

Family	Physical and Environmental Protection (PE)
Controls	PE-1, PE-2, PE-3, PE-6, PE-8, PE-12, PE-13, PE-14, PE-15, PE-16

Family	Planning (PL)
Controls	PL-1, PL-2, PL-4

Family	Personnel Security (PS)
Controls	PS-1, PS-2, PS-3, PS-4, PS-5, PS-6, PS-7, PS-8

Family	Risk Assessment (RA)
Controls	RA-1, RA-2, RA-3, RA-5

Family	System and Services Acquisition (SA)
Controls	SA-1, SA-2, SA-3, SA-4, SA-4(10), SA-5, SA-9

Family	System and Communications Protection (SC)
Controls	SC-1, SC-7(3), SC-7(7), SC-7(8), SC-7(18), SC-7(21), SC-12, SC-12(1), SC-20, SC-22, SC-23

Family	System and Information Integrity (SI)
Controls	SI-1, SI-4, SI-4(2), SI-4(4), SI-4(5), SI-4(12), SI-4(14), SI-4(20), SI-4(22), SI-5, SI-7(2), SI-8(2), SI-12

14.2. FileVault Supplemental

The supplemental guidance found in this section is applicable for the following rules: * sysprefs_filevault_enforce

In macOS 10.15 the internal Apple File System (APFS) volume (including both system and data storage) can be protected by FileVault.

FileVault uses an AES-XTS data encryption algorithm to protect full volumes of internal and external storage. Macs with a secure enclave (T2) utilize the hardware security features of the architecture.

FileVault is described in detail here: https://support.apple.com/guide/security/volume-encryption-with-filevault-sec4c6dc1b6e/web.

FileVault can be enabled in two ways within the macOS. It can be managed using the fdesetup command or by a Configuration Profile. When enabling FileVault via either of the aforementioned methods, you will be required to enter a username and password, which must be a local OpenDirectory account with a valid SecureToken password.

Using the fdesetup Command

When enabling FileVault via the command line in the Terminal application, you can run the following command.

```
/usr/bin/fdesetup enable
```

Running this command will prompt you for a username and password and then enable FileVault and return the personal recovery key. There are a number of management features available when managing FileVault via the command line that are not available when using a configuration profile. More information on these management features is available in the man page for `fdesetup`.

Apple has deprecated `fdesetup` command line tool from recognizing user name and password for security reasons and may remove the ability in future versions of macOS.

Using a Configuration Profile

When managing FileVault with a configuration profile, you must deploy a profile with the payload type `com.apple.MCX.FileVault2`. When using the Enable key to enable FileVault with a configuration profile, you must include 1 of the following:

```
<key>Enable</key>
<string>On</string>
<key>Defer</key>
<true />
```

```
<key>Enable</key>
<string>On</string>
<key>UserEntersMissingInfo</key>
<true/>
```

If using the Defer key it will prompt for the user name and password at logout.

The `UserEntersMissingInfo` key will only work if installed through manual installation, and it will prompt for the username and password immediately.

When using a configuration profile, you can escrow the Recovery key to a Mobile Device Management (MDM) server. Documentation for that can be found on Apple's Developer site: https://developer.apple.com/documentation/devicemanagement/fderecoverykeyescrow.

It's recommended that you use a Personal Recovery key instead of an Institutional key as it will generate a specific key for each device.

FileVault currently only uses password-based authentication and cannot be done using a smartcard or any other type of multi-factor authentication.

14.3. Packet Filter (pf) Supplemental

The supplemental guidance found in this section is applicable for the following rules:

- os_firewall_default_deny_require

macOS contains an application layer firewall (ALF) and a packet filter (PF) firewall.

- The ALF can block incoming traffic on a per-application basis and prevent applications from gaining control of network ports, but it cannot be configured to block outgoing traffic.
  - More information on the ALF can be found here: https://support.apple.com/en-ca/HT201642
- The PF firewall can manipulate virtually any packet data and is highly configurable.
  - More information on the BF firewall can be found here: https://www.openbsd.org/faq/pf/index.html

Below is a script that configures ALF and the PF firewall to meet the requirements defined in NIST SP 800-53 (Rev. 5). The script will make sure the application layer firewall is enabled, set logging to "detailed", set built-in signed applications to automatically receive incoming connections, and set downloaded signed applications to automatically receive incoming connections. It will then create a custom rule set and copy `com.apple.pfctl.plist` from `/System/Library/LaunchDaemons/` into the `/Library/LaunchDaemons` folder and name it `800-53.pfctl.plist`. This is done to not conflict with the system's pf ruleset.

The custom pf rules are created at `/etc/pf.anchors/800_53_pf_anchors`.

The ruleset will block connections on the following ports:

Port	Service
548	Apple File Protocol (AFP)
1900	Bonjour
79	Finger
20, 21	File Transfer Protocol (FTP)
80	HTTP
icmp	ping
143	Internet Message Access Protocol (IMAP)
993	Internet Message Access Protocol over SSL (IMAPS)
3689	Music Sharing

Port	Service
5353	mDNSResponder
2049	Network File System (NFS)
49152	Optical Media Sharing
110	Post Office Protocol (POP3)
995	Post Office Protocol Secure (POP3S)
631	Printer Sharing
3031	Remote Apple Events
5900	Screen Sharing
137, 138, 138, 445	Samba (SMB)
25	Simple Mail Transfer Protocol (SMTP)
22	Secure Shell (SSH)
23	Telnet
69	Trivial File Transfer Protocol (TFTP)
540	Unix-to-Unix Copy (UUCP)

For more on configuring the PF firewall check out the man pages on `pf.conf` and `pfctl`.

```
#!/bin/bash

#enabling macos application firewall
enable_macos_application_firewall () {

    /usr/libexec/ApplicationFirewall/socketfilterfw --setglobalstate on
    /usr/libexec/ApplicationFirewall/socketfilterfw --setloggingopt detail
    /usr/libexec/ApplicationFirewall/socketfilterfw --setallowsigned on
    /usr/libexec/ApplicationFirewall/socketfilterfw --setallowsignedapp on

}

#enabling pf firewall with macsec rules
enable_pf_firewall_with_macsec_rules () {
    macsec_pfctl_plist="/Library/LaunchDaemons/macsec.pfctl.plist"

    if [[ -e "$macsec_pfctl_plist" ]]; then
        echo "LaunchDaemon already exists, flushing and reloading rules..."
        pfctl -e 2> /dev/null
        pfctl -f /etc/pf.conf 2> /dev/null
        return 0
    fi

    # copy system provided launchd for custom ruleset
    cp "/System/Library/LaunchDaemons/com.apple.pfctl.plist" "$macsec_pfctl_plist"
```

```
    #allow pf to be enabled when the job is loaded
    /usr/libexec/PlistBuddy -c "Add :ProgramArguments:1 string -e" $macsec_pfctl_plist
    #use new label to not conflict with System's pfctl
    /usr/libexec/PlistBuddy -c "Set :Label macsec.pfctl" $macsec_pfctl_plist

    # enable the firewall
    pfctl -e 2> /dev/null

    #make pf run at system startup
    launchctl enable system/macsec.pfctl
    launchctl bootstrap system $macsec_pfctl_plist

    pfctl -f /etc/pf.conf 2> /dev/null #flush the pf ruleset (reload the rules)

}

# append the macsec anchors to pf.conf
configure_pf_config_add_macsec_anchors () {

    # check to see if macsec anchors exists
    anchors_exist=$(grep -c '^anchor "macsec_pf_anchors"' /etc/pf.conf)

    if [[ $anchors_exist == "0" ]];then
        echo 'anchor "macsec_pf_anchors"' >> /etc/pf.conf
        echo 'load anchor "macsec_pf_anchors" from
"/etc/pf.anchors/macsec_pf_anchors"' >> /etc/pf.conf
    else
        echo "macsec anchors exist, continuing..."
    fi

}


# Create /etc/pf.anchors/macsec_pf_anchors
create_macsec_pf_anchors () {
if [[ -e /etc/pf.anchors/macsec_pf_anchors ]]; then
    echo "macsec Anchor file exists, deleting and recreating..."
    rm -f /etc/pf.anchors/macsec_pf_anchors
fi


cat > /etc/pf.anchors/macsec_pf_anchors <<'ENDCONFIG'

anchor macsec_pf_anchors

#default deny all in, allow all out and keep state
block in all
pass out all keep state

## Allow DHCP
pass in inet proto udp from port 67 to port 68
```

```
pass in inet6 proto udp from port 547 to port 546

## Allow incoming SSH
pass in proto tcp to any port 22

#apple file service --port 548-- pf firewall rule
block in log proto tcp to any port { 548 }

#bonjour component SSDP --port 1900-- pf firewall rule
block log proto udp to any port 1900

#finger --port 79-- pf firewall rule
block log proto tcp to any port 79

#ftp --ports 20 21-- pf firewall rule
block in log proto { tcp udp } to any port { 20 21 }

#http --port 80-- pf firewall rule
block in log proto { tcp udp } to any port 80

#icmp pf firewall rule
block in log proto icmp

#imap --port 143-- pf firewall rule
block in log proto tcp to any port 143

#imaps --port 993-- pf firewall rule
block in log proto tcp to any port 993

#iTunes sharing --port 3689-- pf firewall rule
block log proto tcp to any port 3689

#mDNSResponder --port 5353-- pf firewall rule
block log proto udp to any port 5353

#nfs --port 2049-- pf firewall rule
block log proto tcp to any port 2049

#optical drive sharing --port 49152-- pf firewall rule
block log proto tcp to any port 49152

#pop3 --port 110-- pf firewall rule
block in log proto tcp to any port 110

#pop3s --port 995-- pf firewall rule
block in log proto tcp to any port 995

#remote apple events --port 3031-- pf firewall rule
block in log proto tcp to any port 3031

#screen_sharing --port 5900-- pf firewall rule
```

```
block in log proto tcp to any port 5900
#allow screen sharing from localhost while tunneled via SSH
pass in quick on lo0 proto tcp from any to any port 5900

#smb --ports 139 445 137 138-- pf firewall rule
block in log proto tcp to any port { 139 445 }
block in log proto udp to any port { 137 138 }

#smtp --port 25-- pf firewall rule
block in log proto tcp to any port 25

#telnet --port 23-- pf firewall rule
block in log proto { tcp udp } to any port 23

#tftp --port 69-- pf firewall rule
block log proto { tcp udp } to any port 69

#uucp --port 540-- pf firewall rule
block log proto tcp to any port 540

ENDCONFIG
}

####

enable_macos_application_firewall
create_macsec_pf_anchors
configure_pf_config_add_macsec_anchors
enable_pf_firewall_with_macsec_rules
```

14.4. Password Policy Supplemental

The supplemental guidance found in this section is applicable for the following rules:

- pwpolicy_lower_case_character_enforce
- pwpolicy_upper_case_character_enforce
- pwpolicy_account_inactivity_enforce
- pwpolicy_minimum_lifetime_enforce

Password policies should be enforced as much as possible via Configuration Profiles. However, the following policies are currently not enforceable via Configuration Profiles, and must therefore be enabled using the `pwpolicy` command:

- Enforcing at least 1 lowercase character
- Enforcing at least 1 uppercase character
- Disabling an account after 35 days of inactivity
- Password minimum lifetime

To set the local policy to meet these requirements, save the following XML password policy to a file.

```
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
  <plist version="1.0">
  <dict>
    <key>policyCategoryAuthentication</key>
    <array>
      <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>300</integer>
          <key>policyAttributeMaximumFailedAuthentications</key>
          <integer>3</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributeLastAuthenticationTime &gt; policyAttributeCurrentTime
- (policyAttributeInactiveDays * 24 * 60 * 60)</string>
        <key>policyIdentifier</key>
        <string>Inactive Account</string>
        <key>policyParameters</key>
        <dict>
          <key>policyAttributeInactiveDays</key>
          <integer>35</integer>
        </dict>
      </dict>
    </array>
    <key>policyCategoryPasswordChange</key>
    <array>
      <dict>
        <key>policyContent</key>
        <string>policyAttributeCurrentTime &gt; policyAttributeLastPasswordChangeTime
+ (policyAttributeExpiresEveryNDays * 24 * 60 * 60)</string>
        <key>policyIdentifier</key>
        <string>Password Expires after 60 days</string>
        <key>policyParameters</key>
        <dict>
          <key>policyAttributeExpiresEveryNDays</key>
          <integer>60</integer>
        </dict>
```

```
      </dict>
    </array>
    <key>policyCategoryPasswordContent</key>
    <array>
      <dict>
        <key>policyContent</key>
        <string>policyAttributePassword matches '(.*[A-Z].*){1,}+'</string>
        <key>policyIdentifier</key>
        <string>Must have at least 1 uppercase letter</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumAlphaCharactersUpperCase</key>
          <integer>1</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributeLastPasswordChangeTime &lt; policyAttributeCurrentTime
- (policyAttributeMinimumLifetimeHours * 60 * 60)</string>
        <key>policyIdentifier</key>
        <string>Minimum Password Lifetime</string>
        <key>policyParameters</key>
        <dict>
          <key>policyAttributeMinimumLifetimeHours</key>
          <integer>24</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributePassword matches '.{15,}+'</string>
        <key>policyIdentifier</key>
        <string>Must be at least 15 characters</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumLength</key>
          <integer>15</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributePassword matches '(.*[0-9].*){1,}+'</string>
        <key>policyIdentifier</key>
        <string>Must have at least 1 numeric value</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumNumericCharacters</key>
          <integer>2</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
```

```
        <string>policyAttributePassword matches '(.*[a-z].*){1,}+'</string>
        <key>policyIdentifier</key>
        <string>Must have at least 1 lowercase letter</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumAlphaCharactersLowerCase</key>
          <integer>1</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributePassword matches '(.*[A-Z].*){1,}+'</string>
        <key>policyIdentifier</key>
        <string>Must have at least 1 uppercase letter</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumAlphaCharacters</key>
          <integer>1</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>policyAttributePassword matches '(.*[^a-zA-Z0-9].*){1,}+'</string>
        <key>policyIdentifier</key>
        <string>Must have at least 1 special characters</string>
        <key>policyParameters</key>
        <dict>
          <key>minimumSymbols</key>
          <integer>1</integer>
        </dict>
      </dict>
      <dict>
        <key>policyContent</key>
        <string>none policyAttributePasswordHashes in
policyAttributePasswordHistory</string>
        <key>policyIdentifier</key>
        <string>Cannot match the last 5 passwords</string>
        <key>policyParameters</key>
        <dict>
          <key>policyAttributePasswordHistoryDepth</key>
          <integer>5</integer>
        </dict>
      </dict>
    </array>
  </dict>
  </plist>
```

Run the following command to load the new policy file, substituting the path to the file in place of "$pwpolicy_file".

```
/usr/bin/pwpolicy setaccountpolicies $pwpolicy_file
```

If directory services is being utilized, password policies should come from the domain.

14.5. Smartcard Supplemental

The supplemental guidance found in this section is applicable for the following rules:

- auth_ssh_password_authentication_disable
- auth_smartcard_enforce
- auth_smartcard_certificate_trust_enforce_moderate
- auth_smartcard_certificate_trust_enforce_high
- auth_smartcard_allow
- auth_pam_sudo_smartcard_enforce
- auth_pam_su_smartcard_enforce
- auth_pam_login_smartcard_enforce

macOS supports smartcards, such as U.S. Personal Identity Verification (PIV) cards and U.S. Department of Defense Common Access Cards (CAC). Smartcards can be used on a macOS for the following:

- Authentication (Loginwindow, Screensaver, SSH, PKINIT, Safari, Finder, and PAM Authorization (`sudo`, `login`, and `su`))
- Digital Encryption
- Digital Signing
- Remote Access (VPN:L2TP)
- Port-based Network Access Control (802.1X)
- Keychain Unlock

macOS has built-in support for USB CCID class-compliant smartcard readers.

Smartcard Pairing

The default method for using smartcards in macOS is a method called "local account pairing". Local account pairing is automatically initiated when a user inserts a smartcard into the Mac. The user is prompted to pair their smartcard with their account. If a user receives a new smartcard, the previous card must be unpaired, and the new card paired to the account. Local account pairing employs fixed key mapping with the hash of a public key on the user's smartcard with a local account.

Smartcard Attribute Mapping

Smartcards can be used to authenticate against a directory via attribute mapping configured in `/private/etc/SmartcardLogin.plist`. This file takes precedence over local account pairing. Attribute mapping matches the configured certificate field values from the smart card to the value in a directory. This may be used with network accounts, mobile accounts, or local accounts.

Smartcard Management in macOS

The following settings are available to manage smartcards (com.apple.security.smartcard):

Key	Type	Value
userPairing	bool	If false, users will not get the pairing dialog, although existing pairings will still work.
allowSmartCard	bool	If false, the SmartCard is disabled for logins, authorizations, and screensaver unlocking. It is still allowed for other functions, such as signing emails and web access. A restart is required for a change of setting to take effect.
checkCertificateTrust	int	Valid values are 0-3: • 0: certificate trust check is turned off • 1: certificate trust check is turned on. Standard validity check is being performed but this does not include additional revocation checks. • 2: certificate trust check is turned on, and a soft revocation check is performed. Until the certificate is explicitly rejected by CRL/OCSP, it is considered valid. This implies that unavailable/unreachable CRL/OCSP allows this check to succeed. • 3: certificate trust check is turned on, plus a hard revocation check is performed. Unless CRL/OCSP explicitly states that “this certificate is OK”, the certificate is considered invalid. This is the most secure value for this setting.
oneCardPerUser	bool	If true, a user can pair with only one smartcard, although existing pairings will be allowed if already set up.
enforceSmartCard	bool	If true, a user can only login or authenticate with a smartcard.
tokenRemovalAction	int	If 1, the screen saver will automatically when the smartcard is removed.
allowUnmappedUsers	int	If 1, allows users who are in a directory group to be exempt from smartcard-only enforcement. The group allowed for exemption is defined in /private/etc/SmartcardLogin.plist

A custom configuration profile (`com.apple.loginwindow`) should be created to disable automatic login when FileVault is enabled. This ensures that authorized users boot their Macs, enter a password at the pre-boot screen (which decrypts the boot volume), and are then presented with a login window

where they can authenticate with a smartcard.

Key	Type	Value
DisableFDEAutoLogin	bool	If true, both Extensible Firmware Interface (EFI) login password and loginwindow PIN are required.

Trusted Authorities

The macOS allows users to specify which certificate authorities (CA) can be used for trust evaluation during smartcard authentication. Only CAs listed in the TrustedAuthorities section of the `SmartcardLogin.plist` will be evaluated as trusted. This setting only works if `checkCertificateTrust` is set to either 1, 2, or 3 in `com.apple.security.smartcard`.

To get the SHA-256 hash in the correct format, run the following command within terminal:

```
/usr/bin/openssl x509 -noout -fingerprint -sha256 -inform pem -in <issuer cert> |
/usr/bin/awk -F '=' '{print $2}' |  /usr/bin/sed 's/://g'
```

To configure Trusted Authorities, the `SmartcardLogin.plist` should be minimally configured as below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>AttributeMapping</key>
    <dict>
          <key>fields</key>
          <array>
              <string>NT Principal Name</string>
          </array>
          <key>formatString</key>
          <string>Kerberos:$1</string>
          <key>dsAttributeString</key>
          <string>dsAttrTypeStandard:AltSecurityIdentities</string>
    </dict>
    <key>TrustedAuthorities</key>
  <array>
      <string>SHA256_HASH_OF_CERTDOMAIN_1,SHA256_HASH_OF_CERTDOMAIN_2</string>
  </array>
</dict>
</plist>
```

Smartcard Enforcement Exemption

Group Exemption

Starting in macOS 10.15, enforcement on a system can be granularly configured by adding a field to `/private/etc/SmartcardLogin.plist`. The `NotEnforcedGroup` can be added to the file to list a Directory group that will not be included in smartcard enforcement. In order to activate this feature, `enforceSmartCard` and `allowUnmappedUsers` must be applied via a configuration profile (`com.apple.security.smartcard`).

To configure the `NotEnforcedGroup`, the `SmartcardLogin.plist` should be minimally configured as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>AttributeMapping</key>
    <dict>
          <key>fields</key>
          <array>
              <string>NT Principal Name</string>
          </array>
          <key>formatString</key>
          <string>Kerberos:$1</string>
          <key>dsAttributeString</key>
          <string>dsAttrTypeStandard:AltSecurityIdentities</string>
    </dict>
    <key>TrustedAuthorities</key>
  <array>
      <string>SHA256_HASH_OF_CERTDOMAIN_1,SHA256_HASH_OF_CERTDOMAIN_2</string>
  </array>
    <key>NotEnforcedGroup</key>
    <string>EXEMPTGROUP</key>
</dict>
</plist>
```

Once a system is configured for the `NotEnforcedGroup` a user can be added to the assigned group by running the following:

```
/usr/sbin/dseditgroup -o edit -a <exempt_user> -t user <notenforcegroup>
```

User Exemption

Alternatively, if a single user needs to be exempt for a period of time, `kDSNativeAttrTypePrefix:SmartCardEnforcement` can be set in the user's Open Directory record. The following values can be set:

- 0 - The system default is respected.

- 1 - Smartcard enforcement is enabled.
- 2 - Smartcard enforcement is disabled.

In Active Directory environments, the value of the `userAccountControl` attribute is respected.

Run the following command to set the exemption when booted from macOS:

```
/usr/bin/dscl . -append /Users/<username> SmartCardEnforcement 2
```

Run the following command to set the exemption when booted from Recovery:

```
/usr/bin/defaults write /Volumes/Macintosh\
HD/var/db/dslocal/nodes/Default/users/<username> SmartCardEnforcement -array-add 2
```

Pluggable Authentication Module (PAM)

Terminal sessions in macOS can be configured for smartcard enforcement by modifying the PAM modules for `sudo`, `su`, and `login`.

```
/etc/pam.d/sudo
# sudo: auth account password session
auth        sufficient    pam_smartcard.so
auth        required      pam_opendirectory.so
auth        required      pam_deny.so
account     required      pam_permit.so
password    required      pam_deny.so
session     required      pam_permit.so
```

```
/etc/pam.d/su
# su: auth account password session
auth        sufficient    pam_smartcard.so
auth        required      pam_rootok.so
auth        required      pam_group.so no_warn group=admin,wheel ruser root_only
fail_safe
account     required      pam_permit.so
account     required      pam_opendirectory.so no_check_shell
password    required      pam_opendirectory.so
session     required      pam_launchd.so
```

```
/etc/pam.d/login
# login: auth account password session
auth        sufficient    pam_smartcard.so
auth        optional      pam_krb5.so use_kcminit
```

```
auth        optional      pam_ntlm.so try_first_pass
auth        optional      pam_mount.so try_first_pass
auth        required      pam_opendirectory.so try_first_pass
auth        required      pam_deny.so
account     required      pam_nologin.so
account     required      pam_opendirectory.so
password    required      pam_opendirectory.so
session     required      pam_launchd.so
session     required      pam_uwtmp.so
session     optional      pam_mount.so
```

Screen Sharing and Screen Recording

macOS will disable support for TouchID, Watch, or Smartcard authentication when being watched or recorded. This can cause certain portions of the system to not recognize your smartcard.

In Unified Logging you'll notice an entry such as

```
2022-07-14 16:45:46.880038-0400 0x2F97 Info 0xC8D2 1600 SecurityAgent: (SecurityAgent)
[com.apple.Authorization:SecurityAgent] Screen is being watched, no Touch ID, Watch or
SmartCard support is allowed
```

This can be remediated by writing the preference domain com.apple.authorization with the key ignoreARD.

`defaults write com.apple.Authorization ignoreARD -bool true`

Or applied system wide with a configuration profile named `com.apple.security.authorization.mobileconfig` in the project's `includes` folder.

```
<key>PayloadType</key>
<string>com.apple.security.authorization</string>
<key>ignoreArd</key>
<true/>
```