macOS用アーティファクト収集ツールと簡易マルウェア解析サンドボックスの実装と利用方法

Japan Security Analyst Conference 2020

株式会社インターネットイニシアティブ
小林 稔

Who am I?

小林 稔

- 株式会社インターネットイニシアティブ セキュリティ本部 セキュリティ情報統括室
  技術調査、社内インシデントレスポンス
- 外部活動
  セキュリティキャンプ全国大会講師 2017-2019
  Japan Security Analyst Conference 2018
  Black Hat USA 2018 Briefing
- Twitter: @unkn0wnbit

Agenda

1. モチベーション
2. macOS Artifact Collector
3. Norimaci
4. bgiparser
5. Future Work
6. まとめ

1 モチベーション

モチベーション (1)

- macOSセキュリティの解析者もツールもWindowsと比べて少ない
  - OSのシェアや需要の大きさの違い
  - 解析者、ツールが少ないことが悪循環か
- 情報源となるのはわずかな人物や組織のみ
  - Sarah Edwards (mac4n6.com)
  - Patrick Wardle (objective-see.com)
  - Yogesh Khatri (swiftforensics.com)
  - 川崎 隆哉 (padawan-4n6.hatenablog.com)
  - FireEye
  - CrowdStrike
  - SentinelOne
  - Twitterの“#mac4n6”ハッシュタグ

モチベーション (2)

- 要望を満たすツールが見つからない
  - アーティファクト収集を行うツールがほとんどない
  - 最新のmacOSが動作するサンドボックスがない
- 誰も作ってくれなさそうなので自分で作った
- 作成したツールを公開することで、少しでも解析に関わってくれる人を増やしたい

2

macOS Artifact Collector (macosac)

Forensic Artifact Collection Tool for macOS

既存のアーティファクト解析・収集ツールの問題点 (1)

- 既存の解析・収集ツール
  - アーティファクト収集専用のツールはほとんど存在しない
  - 様々なアーティファクトのパーサをパッケージングした総合解析ツールがほとんどを占める
- 問題点
  - 解析ツールは結果をすぐに確認できる反面、パーサのバグで解析が停止する可能性がある
  - パーサのバグが発生した場合、マルウェア解析など、他の作業に遅れが生じる
  - Time Machineバックアップやローカルスナップショット内のアーティファクトを解析・収集しない
  - アーティファクトファイルの拡張属性(Extended Attributes)を収集の際に保持しない

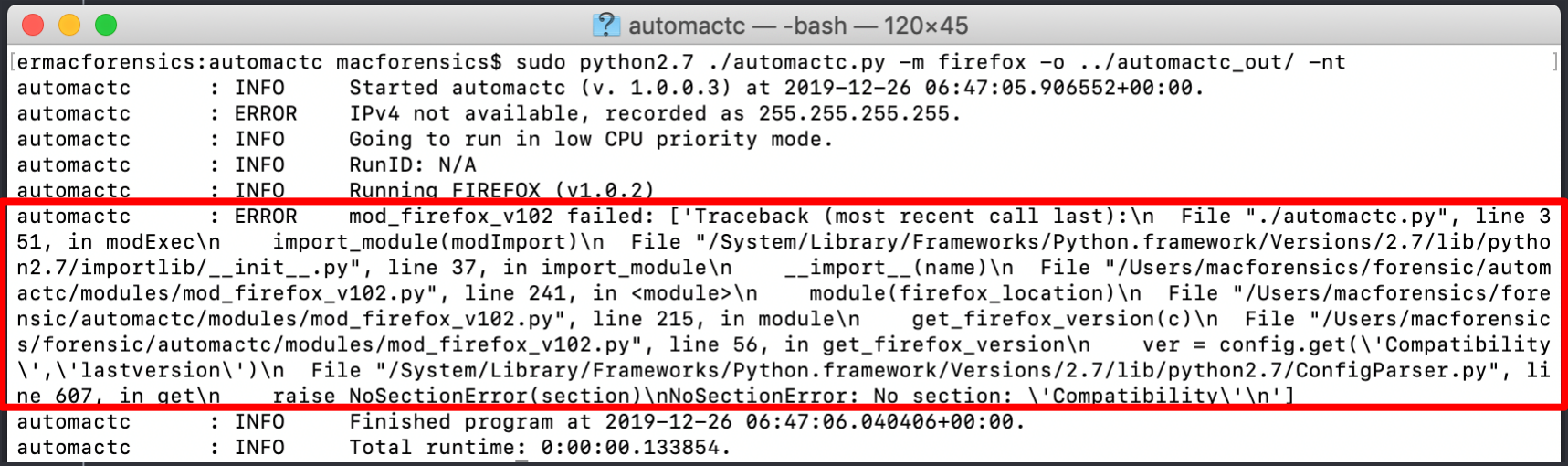
既存のアーティファクト解析・収集ツールの問題点 (2)

	アーティファクト解析	アーティファクト収集	バックアップを処理対象に含める	備考
mac_apt	✓	✓	✗	ディレクトリ構造と拡張属性を保持しない
AutoMacTC	✓	✗	✗	-
 osxcollector	This repository has been archived by the owner. It is now read-only.			osxcollector開発停止
Venator	✓	✗	✗	ライブシステムのみが対象
CyLR	✗	✓	✗	拡張属性を保持しない

Icon made by Freepik from www.flaticon.com

既存のアーティファクト解析・収集ツールの問題点 (3)

- AutoMacTCのFirefoxモジュールでエラーが発生している例



既存のアーティファクト解析・収集ツールの問題点 (4)

- アーティファクト収集と解析のプロセスを分けた方が安全
  - 解析ツールのバグ対応を考慮した作業手順が必要ないため、解析対象ホストで行う作業がシンプルになる
  - インシデントレスポンスの情報収集は専任ではない人が実行する可能性があることを考慮すると複雑な作業手順は避けるべき
  - 収集専用ツールはバグが発生しにくい
  - 収集したアーティファクトは既存のツールでオフライン解析を行う
    - バグがあっても修正して正しいデータを取得できる
    - 解析ツールの仕様に合うような収集方法を検討する

macosacの実装 (1) – 目標

- Python 2.7で実装する
  - macOSに標準でインストールされている
- 収集するアーティファクトを設定ファイルで簡単に指定できる
- ディレクトリ構造を保持したままアーティファクトを収集する
  - 多くの解析ツールはデフォルトのディレクトリ構造を想定している
- バックアップからもアーティファクトを収集する
  - Time Machineバックアップ
  - ローカルスナップショット
- HFS+やAPFSの拡張属性(Extended Attributes)を保持する
- USBメモリ等から実行することを前提とする
  - デフォルトインストールされているコマンドやライブラリのみで実装する
- 解析対象ホストのストレージに極力書き込みを行わずにアーティファクトを収集する

macosacの実装 (2) – 処理の流れ

アーティファクトファイルリストアップ

Time Machineバックアップ、
ローカルスナップショットも対象

アーティファクトファイルを保全先にコピー

ディレクトリ構造、拡張属性を保持

設定ファイル
（アーティファクトリスト）

macOS Artifact
Collector

別のホストでアーティファクトを解析 —— AutoMacTCなど

macosacの実装 (3) – 設定ファイルフォーマット

- macosac.ini
  - macosacと同じディレクトリに配置する
  - macosacの起動時に自動で読み込まれる
  - デフォルトのカテゴリリストはAppendix 1を参照

```
[mru]
file1_1 = /Users/*/Library/Application Support/com.apple.sharedfilelist/*.sfl
file1_2 = /Users/*/Library/Application Support/com.apple.sharedfilelist/*/*.sfl
file1_3 = /private/var/*/Library/Application Support/com.apple.sharedfilelist/*/*.sfl
file2_1 = /Users/*/Library/Application Support/com.apple.sharedfilelist/*.sfl2
file2_2 = /Users/*/Library/Application Support/com.apple.sharedfilelist/*/*.sfl2
file2_3 = /private/var/*/Library/Application Support/com.apple.sharedfilelist/*/*.sfl2
file3_1 = /Users/*/Library/Containers/*/Data/Library/Preferences/*.securebookmarks.plist
file3_2 = /private/var/*/Library/Containers/*/Data/Library/Preferences/*.securebookmarks.plist
file4_1 = /Users/*/Library/Preferences/com.apple.finder.plist
file4_2 = /private/var/*/Library/Preferences/com.apple.finder.plist
file5_1 = /Users/*/Library/Preferences/com.apple.sidebarlists.plist
file5_2 = /private/var/*/Library/Preferences/com.apple.sidebarlists.plist
```

カテゴリ名

アーティファクトファイルのフルパス
（ワイルドカード表現が可能）

macosacの実装 (4) – アーティファクトのコピー方法

- 起動時にアーティファクトのコピー方法を指定する
- rsync（デフォルト）
  - macOSにデフォルトでインストールされている
  - ファイル同期の定番ツールであり信頼性が高い
  - ファイルのタイムスタンプの保持や拡張属性のコピーにも対応している
  - “-aREL”オプションで実行
  - 一時ファイルをコピー先に作成する (--temp-dir)
  - コピーしたファイル名をログに記録する (--log-file)
- ビルトインコピー機能
  - 何らかの理由でrsyncが使えない場合に使用する
  - メタデータもrsyncとほぼ同等にコピーできる

- macosacの実装 (5) – バックアップデータへのアクセス手法
  - Time Machineバックアップの確認

```
$ tmutil listbackups
/Volumes/TimeMachine/Backups.backupdb/ermacforensics/2019-06-13-195104
/Volumes/TimeMachine/Backups.backupdb/ermacforensics/2019-07-04-154843
/Volumes/TimeMachine/Backups.backupdb/ermacforensics/2019-07-19-194030
/Volumes/TimeMachine/Backups.backupdb/ermacforensics/2019-07-25-120411
/Volumes/TimeMachine/Backups.backupdb/ermacforensics/2019-08-01-004740
```

バックアップ日時

  - ローカルスナップショットの確認

```
$ tmutil listlocalsnapshots /
com.apple.TimeMachine.2019-12-26-121648
com.apple.TimeMachine.2019-12-26-131643
com.apple.TimeMachine.2019-12-26-141643
com.apple.TimeMachine.2019-12-26-151643
com.apple.TimeMachine.2019-12-26-161644
```

バックアップ日時

macosacの実装 (6) – バックアップデータへのアクセス手法

- ローカルスナップショットのマウント
  - /Volumes以下にマウントされる

```
$ tmutil mountlocalsnapshots /
Mounted local snapshots: (
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-151643/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-141643/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-191644/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-131643/Macintosh HD",
```

- ローカルスナップショットのアンマウント

```
$ tmutil unmountlocalsnapshots /
Unmounted local snapshots: (
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-191644/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-181644/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-27-104519/Macintosh HD",
    "/Volumes/com.apple.TimeMachine.localsnapshots/Backups.backupdb/ermacforensics/2019-12-26-121648/Macintosh HD",
```

macosacの実装 (7) – バックアップデータへのアクセス手法

- バックアップデータの取得
  - Time Machineバックアップもローカルスナップショットも“/Volumes”以下にマウントされる
    - ディレクトリ構造を保持するため、バックアップ内のアーティファクトはコピー先ディレクトリの“Volumes”以下にコピーされる
  - バックアップ日時を基に、必要なバックアップからアーティファクトをコピーする

macosacの実装 (8) – アーティファクトの保存方法

- 3種類の保存方法から選択
  1. 指定した保存先に単純コピー（デフォルト）
     - ホスト名と実行日時の名前を持つサブディレクトリが作成される
  2. 指定した保存先にHFS+のファイルシステムを持つDMGを作成して、そこにコピー
     - ホスト名と実行日時の名前を持つDMGが作成される
  3. 2.の方式でコピーした後、DMGを読み取り専用形式に変換

macosacの実装 (9) – アーティファクトの保存方法

- DMGに保存する方法がお勧め
  - 保存先のファイルシステムを意識しなくて良い
  - パーミッションや拡張属性などのメタデータも保持できる
  - DMG形式が扱いにくい場合、E01(EWF)形式に変換できる
- 保存先がexFAT等の場合に、1の方式で保存
  - 拡張属性はAppleDouble形式で保存される
  - 解析ツールの仕様によっては拡張属性が解析されないかも
- 保存先がHFS+またはAPFSの場合は、どの方式を選択しても問題ない

macosacの実装 (10) – アーティファクトの保存方法

- DMGからE01への変換コマンド例

```
$ brew install libewf
$ hdiutil attach -nomount ermacforensics_20191010_171704.dmg
$ diskutil list →マウントしたDMGのデバイス名を確認
$ ewfacquire -t evidence -v /dev/disk4s1 →/dev/disk4s1からevidence.E01を作成
$ hdiutil detach /dev/disk4
```

  - 解析対象以外のホストで実行すること

User Consentによるアクセス制限 (1)

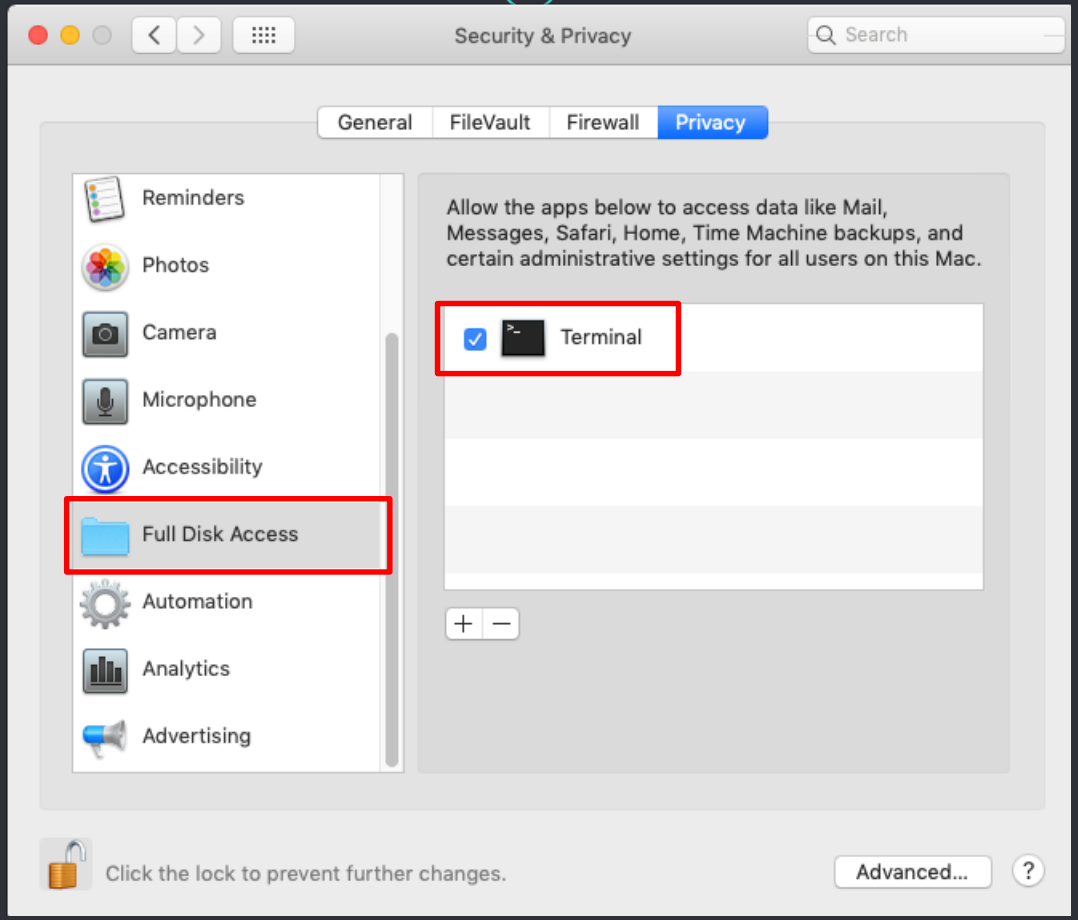
- 一部のフォルダにはUser Consentによるアクセス制限がかかっている
  - アプリケーションがユーザのプライバシーに関わるデータにアクセスする際に、ユーザの許可を必要とするセキュリティフレームワーク
  - macOS 10.13 High Sierra以降で採用
- root権限でもアクセスすることはできない

```
ermacforensics:~ macforensics$ sudo ls -al ~/Library/Safari/
Password:
ls: : Operation not permitted
```

User Consentによるアクセス制限 (2)

- macOS 10.14 Mojave以降、保護範囲が拡大され、バックアップやアンチウイルス等のアプリケーションには、「フルディスクアクセス」の許可が必要となる
- macosacの場合、Terminal.appにフルディスクアクセスを許可する必要がある
  - ただし、この設定を行っても一部のフォルダにはアクセスできない

User Consentによるアクセス制限 (3)



macosacコマンドラインオプション (1)

- -o / --outputdir
  - 出力先ディレクトリ（デフォルト：macosac.pyと同じディレクトリ）
- -t / --outputtype
  - 保存形式 dir/dmg/ro-dmg（デフォルト : dir）
- -l / --list
  - macosac.iniファイルに記述されているカテゴリを表示する
- -c / --categories
  - 指定したカテゴリのアーティファクトファイルのみ収集する（デフォルト : all）
- -ls / --localsnapshots
  - APFSローカルスナップショットを収集対象に含める
- -tm / --timemachine
  - Time Machineバックアップを収集対象に含める
- -ts / --timestamp
  - 収集を始めるバックアップ日時を指定する
- --use-builtincopy
  - rsyncの代わりにビルトインコピー機能を使用する

macosacコマンドラインオプション (2)

- macosac実行例

```
$ sudo python2.7 ./macosac.py -t ro-dmg -o ./out/ -tm -ts 2019-12-01-090000
```

  - 全てのカテゴリのアーティファクトを収集する
  - outディレクトリ以下にDMGディスクイメージを作成する
  - 現在ストレージにあるアーティファクトと2019年12月1日 9時以降に作成されたTime Machineバックアップ内のアーティファクトをDMGにコピーする
  - コピー完了後、DMGを読み込み専用に変換する

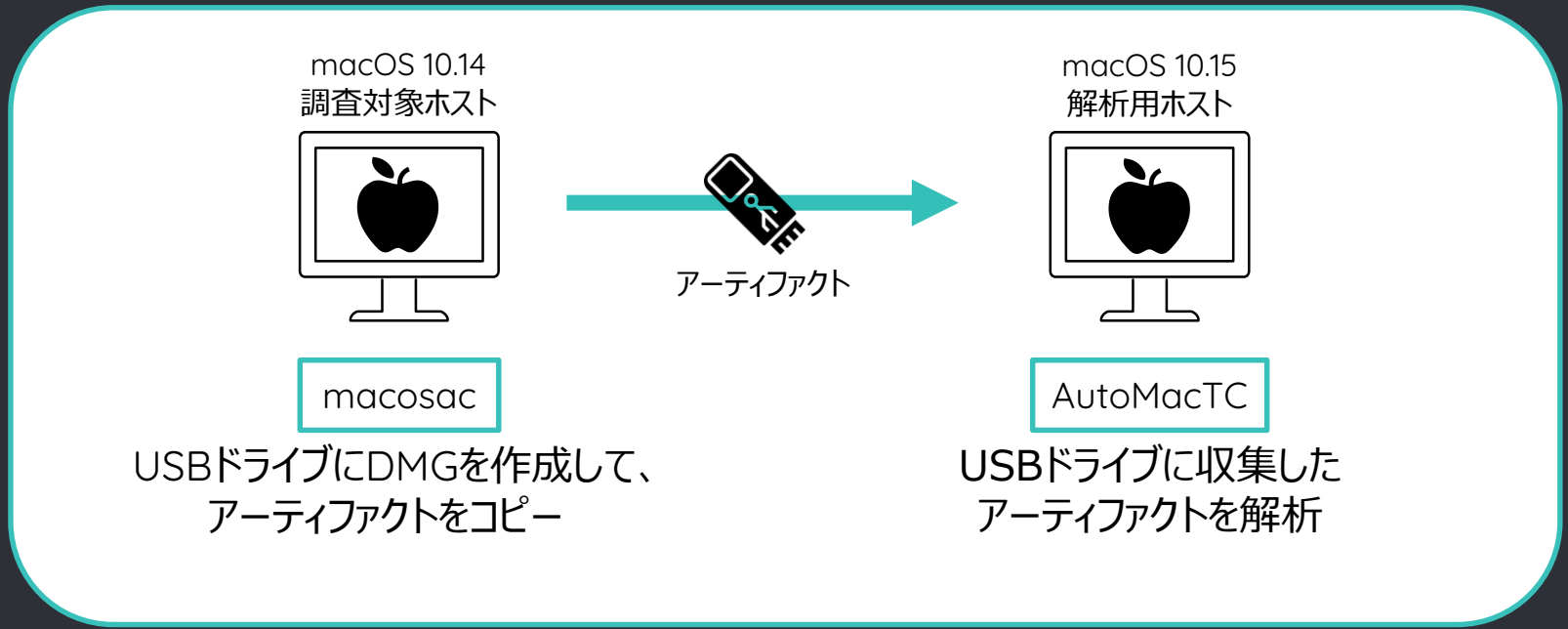
D

macOS Artifact Collector Demo

macosacとAutoMacTCの連携

macOS Artifact Collector Demo (1)

デモ環境(VMware Fusion)

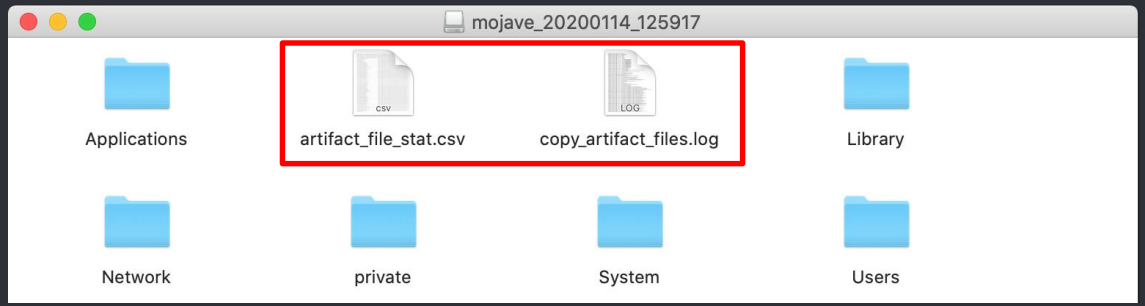


Icon made by phatplus from www.flaticon.com

DEMO MOVIE 1

macOS Artifact Collector Demo (2)

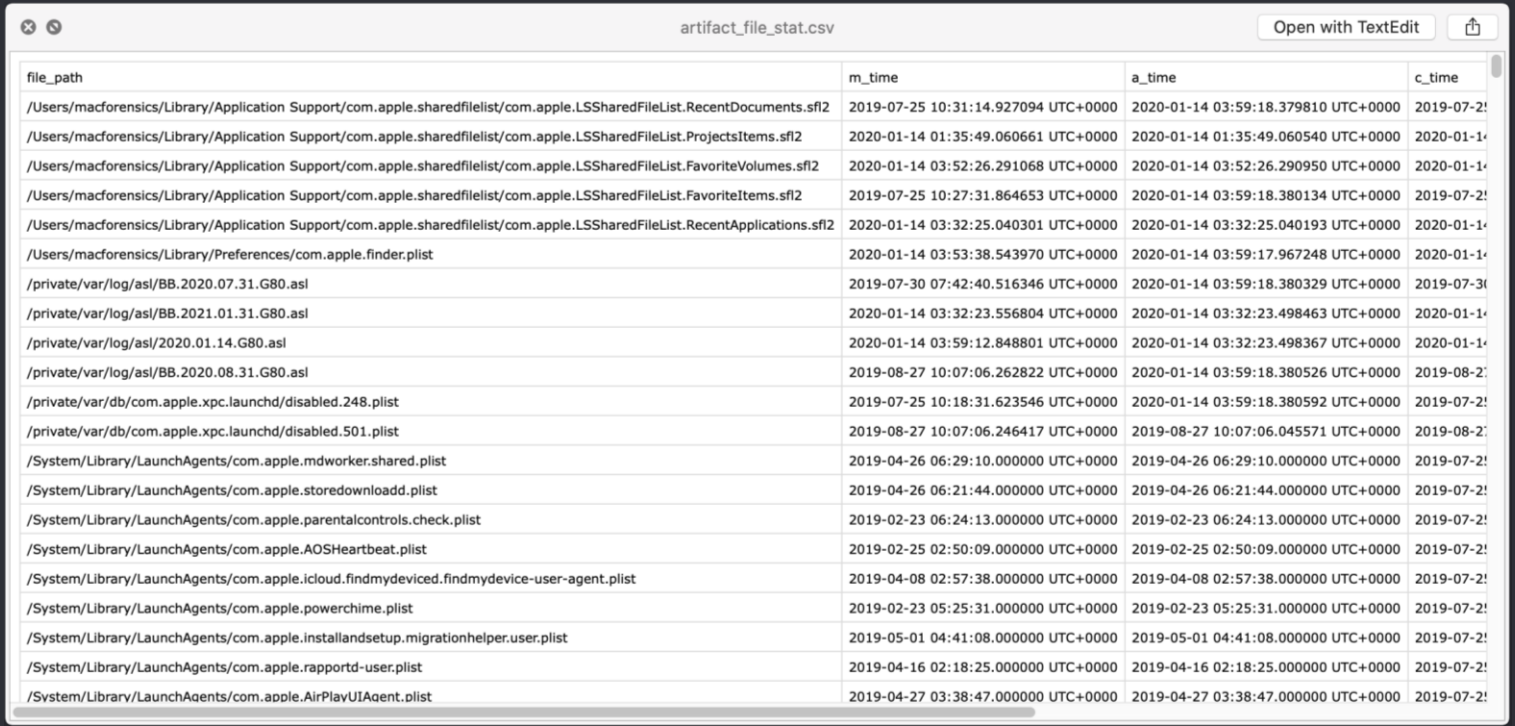
- 収集したアーティファクト例



- artifact_file_stat.csv
  - コピー前のアーティファクトのMACBタイムなどのメタ情報
- copy_artifact_files.log
  - アーティファクトコピー時の動作ログ

macOS Artifact Collector Demo (3)

- artifact_file_stat.csv

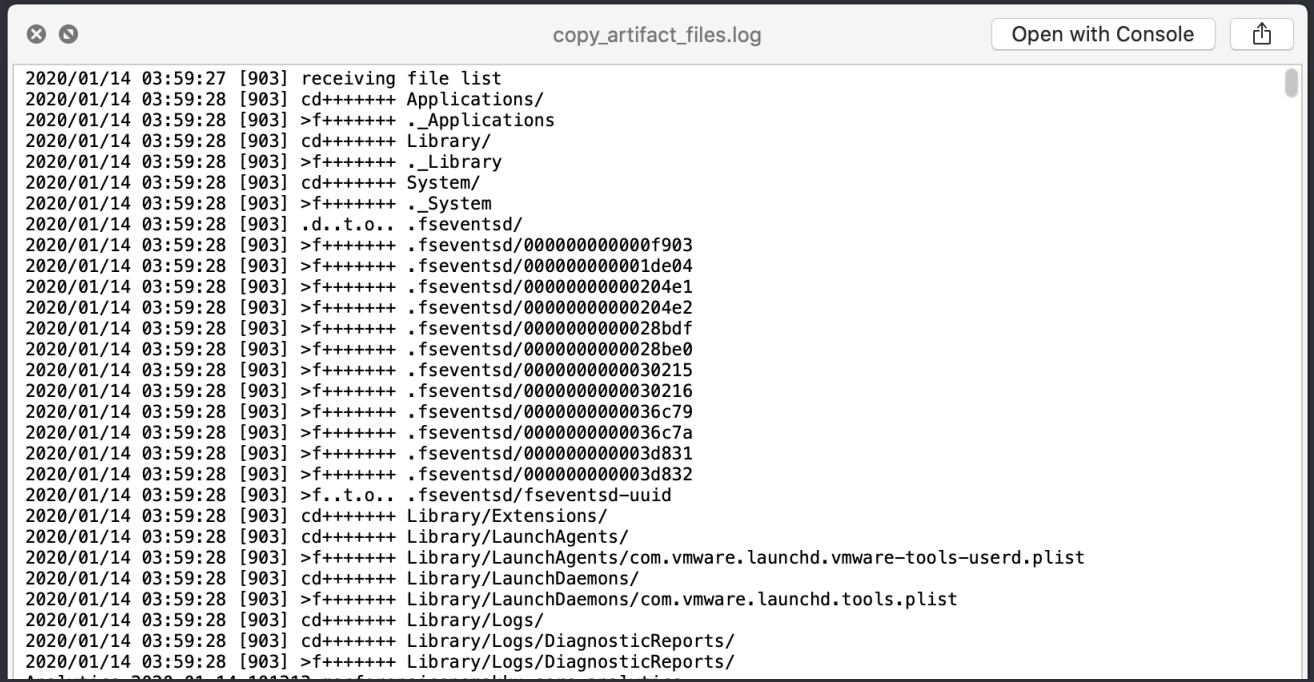


artifact_file_stat.csv — Open with TextEdit

file_path	m_time	a_time	c_time
/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentDocuments.sfl2	2019-07-25 10:31:14.927094 UTC+0000	2020-01-14 03:59:18.379810 UTC+0000	2019-07-25
/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.ProjectsItems.sfl2	2020-01-14 01:35:49.060661 UTC+0000	2020-01-14 01:35:49.060540 UTC+0000	2020-01-14
/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteVolumes.sfl2	2020-01-14 03:52:26.291068 UTC+0000	2020-01-14 03:52:26.290950 UTC+0000	2020-01-14
/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.FavoriteItems.sfl2	2019-07-25 10:27:31.864653 UTC+0000	2020-01-14 03:59:18.380134 UTC+0000	2019-07-25
/Users/macforensics/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl2	2020-01-14 03:32:25.040301 UTC+0000	2020-01-14 03:32:25.040193 UTC+0000	2020-01-14
/Users/macforensics/Library/Preferences/com.apple.finder.plist	2020-01-14 03:53:38.543970 UTC+0000	2020-01-14 03:59:17.967248 UTC+0000	2020-01-14
/private/var/log/asl/BB.2020.07.31.G80.asl	2019-07-30 07:42:40.516346 UTC+0000	2020-01-14 03:59:18.380329 UTC+0000	2019-07-30
/private/var/log/asl/BB.2021.01.31.G80.asl	2020-01-14 03:32:23.556804 UTC+0000	2020-01-14 03:32:23.498463 UTC+0000	2020-01-14
/private/var/log/asl/2020.01.14.G80.asl	2020-01-14 03:59:12.848801 UTC+0000	2020-01-14 03:32:23.498367 UTC+0000	2020-01-14
/private/var/log/asl/BB.2020.08.31.G80.asl	2019-08-27 10:07:06.262822 UTC+0000	2020-01-14 03:59:18.380526 UTC+0000	2019-08-27
/private/var/db/com.apple.xpc.launchd/disabled.248.plist	2019-07-25 10:18:31.623546 UTC+0000	2020-01-14 03:59:18.380592 UTC+0000	2019-07-25
/private/var/db/com.apple.xpc.launchd/disabled.501.plist	2019-08-27 10:07:06.246417 UTC+0000	2019-08-27 10:07:06.045571 UTC+0000	2019-08-27
/System/Library/LaunchAgents/com.apple.mdworker.shared.plist	2019-04-26 06:29:10.000000 UTC+0000	2019-04-26 06:29:10.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.storedownloadd.plist	2019-04-26 06:21:44.000000 UTC+0000	2019-04-26 06:21:44.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.parentalcontrols.check.plist	2019-02-23 06:24:13.000000 UTC+0000	2019-02-23 06:24:13.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.AOSHeartbeat.plist	2019-02-25 02:50:09.000000 UTC+0000	2019-02-25 02:50:09.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.icloud.findmydeviced.findmydevice-user-agent.plist	2019-04-08 02:57:38.000000 UTC+0000	2019-04-08 02:57:38.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.powerchime.plist	2019-02-23 05:25:31.000000 UTC+0000	2019-02-23 05:25:31.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.installandsetup.migrationhelper.user.plist	2019-05-01 04:41:08.000000 UTC+0000	2019-05-01 04:41:08.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.rapportd-user.plist	2019-04-16 02:18:25.000000 UTC+0000	2019-04-16 02:18:25.000000 UTC+0000	2019-07-25
/System/Library/LaunchAgents/com.apple.AirPlayUIAgent.plist	2019-04-27 03:38:47.000000 UTC+0000	2019-04-27 03:38:47.000000 UTC+0000	2019-07-25

macOS Artifact Collector Demo (4)

- copy_artifact_files.log



解析ツールとの連携

- 収集したアーティファクトは解析ツールで解析できることを確認済み
  - AutoMacTC 1.0.0.3
  - mac_apt 0.4.1
  - APOLLO 08252019
    - アーティファクト(SQLite DB)が更新されるため、読み取り専用DMG以外で保存する
  - BlackLight 2019 R2
    - DMGで保存した場合、E01への変換が必要
  - AXIOM 3.8.0
    - DMGで保存した場合、E01への変換が必要

将来のmacOSでの課題と対策 (1)

- 将来のmacOSにはPython, Ruby, Perl等のスクリプト言語が含まれない
- これらのスクリプト言語を採用しているツールはmacOSで実行できなくなる
- macosacも含め、多くのセキュリティ関連ツールが影響を受ける

Use of Python 2.7 isn't recommended as this version is included in macOS for compatibility with legacy software. Future versions of macOS won't include Python 2.7. Instead, it's recommended that you run python3 from within Terminal. (51097165)

https://developer.apple.com/documentation/macos_release_notes/macos_catalina_10_15_release_notes

将来のmacOSでの課題と対策 (2)

- PyInstallerによるパッケージング
  - PythonがmacOSにインストールされていなくても スクリプトを実行できる
- パッケージング例

```
$ sudo pip2 install virtualenv
$ virtualenv ~/envs/macosac
$ source ~/envs/macosac/bin/activate
$ pip2 install pyinstaller xattr --user
$ pyinstaller --onefile --runtime-tmpdir ./ macosac.py
$ cp ./dist/macosac ./macosac.ini /Volumes/thumbdrive/
```

2

Norimaci

Simple Malware Analysis Sandbox

既存のマルウェア解析サンドボックスの問題点 (1)

- 既存のサンドボックス
  - CuckooベースのmacOS向けのオープンソースのサンドボックスはいくつか存在する
  - Cuckoo: OS X 10.10 Yosemiteまでサポート
  - Mac-A-Mal: macOS 10.12 Sierraまでサポート
- 問題点
  - 積極的にメンテナンスが行われておらず、最新のmacOSで解析を行うのは難しい可能性がある

既存のマルウェア解析サンドボックスの問題点 (2)

- Cuckoo
  - アクティビティの監視にDTraceを使用している
  - OS X 10.11 El Capitan以降、Security Integrity Policy (SIP)が導入され、デフォルトでDTraceは使用できなくなった
- Mac-A-Mal
  - アクティビティ監視用のカーネル拡張を提供しているが、macOS 10.13以降はサポートしていない
  - カーネル拡張をメンテナンスし続けるコストが高いのではないか

Norimaciの実装 (1) – 目標

- サンドボックスとしてのメンテナンスコストを抑えた構成
  - サードパーティのアクティビティ監視ツールを積極的に利用する
    - SIPと互換性があるとベター
  - 将来のmacOSでも動作することが期待できるシンプルなツール構成を目指す
- これらを満たすリファレンス実装: Noriben

Norimaciの実装 (2)

- Noriben Malware Analysis Sandbox
  - https://github.com/Rurik/Noriben
  - Windows用のシンプルなマルウェア解析サンドボックス
  - アクティビティ監視にSysinternalsのProcess Monitorを利用する
  - Pythonスクリプトでアクティビティログの解析とレポーティングを行う

Noriben = のり弁

https://www.slideshare.net/bbaskin/bh15-arsenal-noriben

Norimaciの実装 (3)

- Why “Norimaci”?
  - 春山さん(@cci_forensics)のアドバイス
  - “Noriben”-like tool on “Macintosh”
    → Norimaci (発音: norimaki)
    = のり巻き
  - のり巻きはのり弁と同じように
    最小限の食材で作られている
  - Norimaciも最小限の機能を提供する

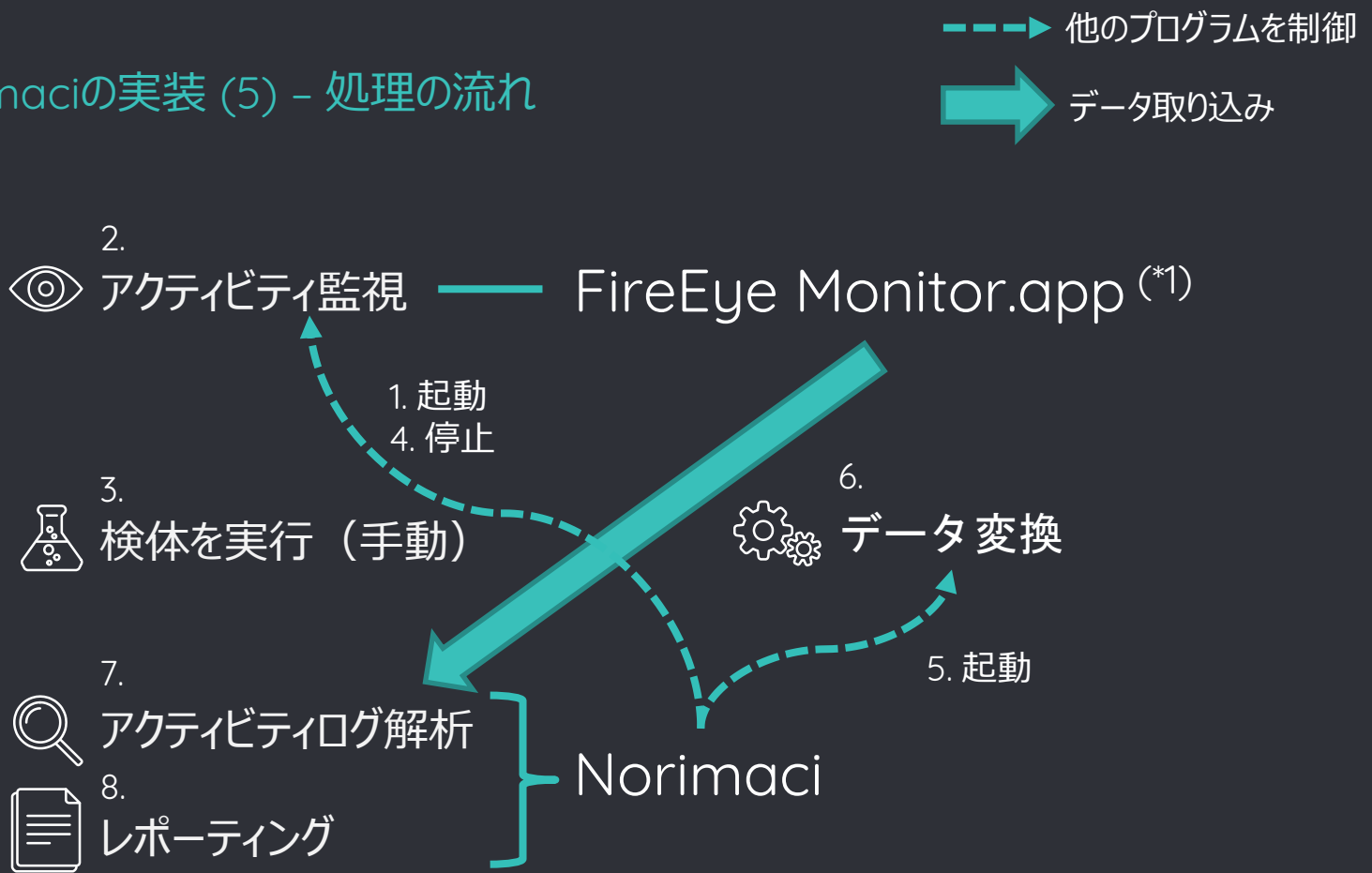
- Noribenのレポートフォーマットを踏襲している
  - Noriben作者の了承済み
  - Thanks Brian ☺

https://commons.wikimedia.org/wiki/File:Futomaki_zushi_in_201902.jpg

Norimaciの実装 (4) – アクティビティ監視

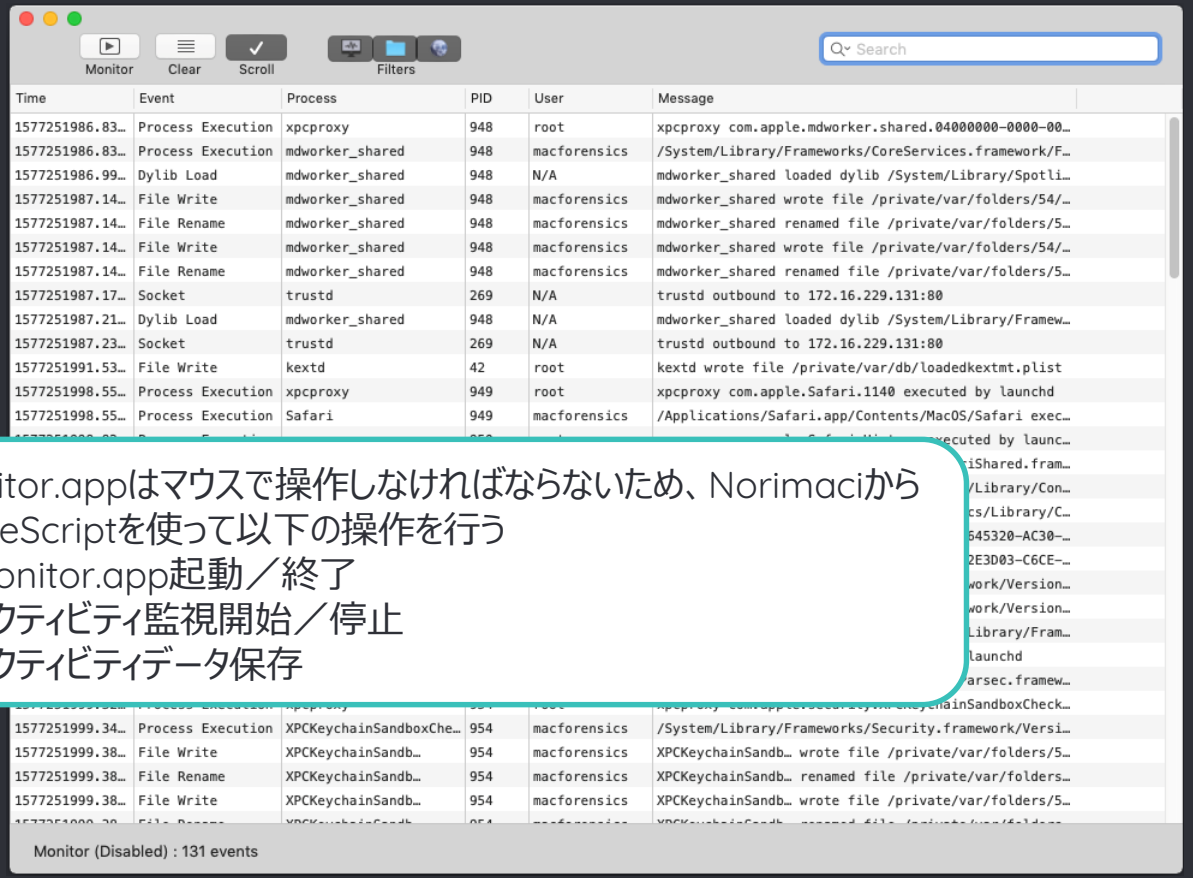
- macOS用アクティビティ監視ツール
  - FireEye Monitor.app
  - Fortinet FortiAppMonitor
  - これらのツールはSIP互換性あり
- 今回はMonitor.appを利用する
  - FortiAppMonitorはアクティビティログを保存できない

Norimaciの実装 (5) – 処理の流れ



(*1) https://www.fireeye.com/services/freeware/monitor.html

Norimaciの実装 (6) – Monitor.app

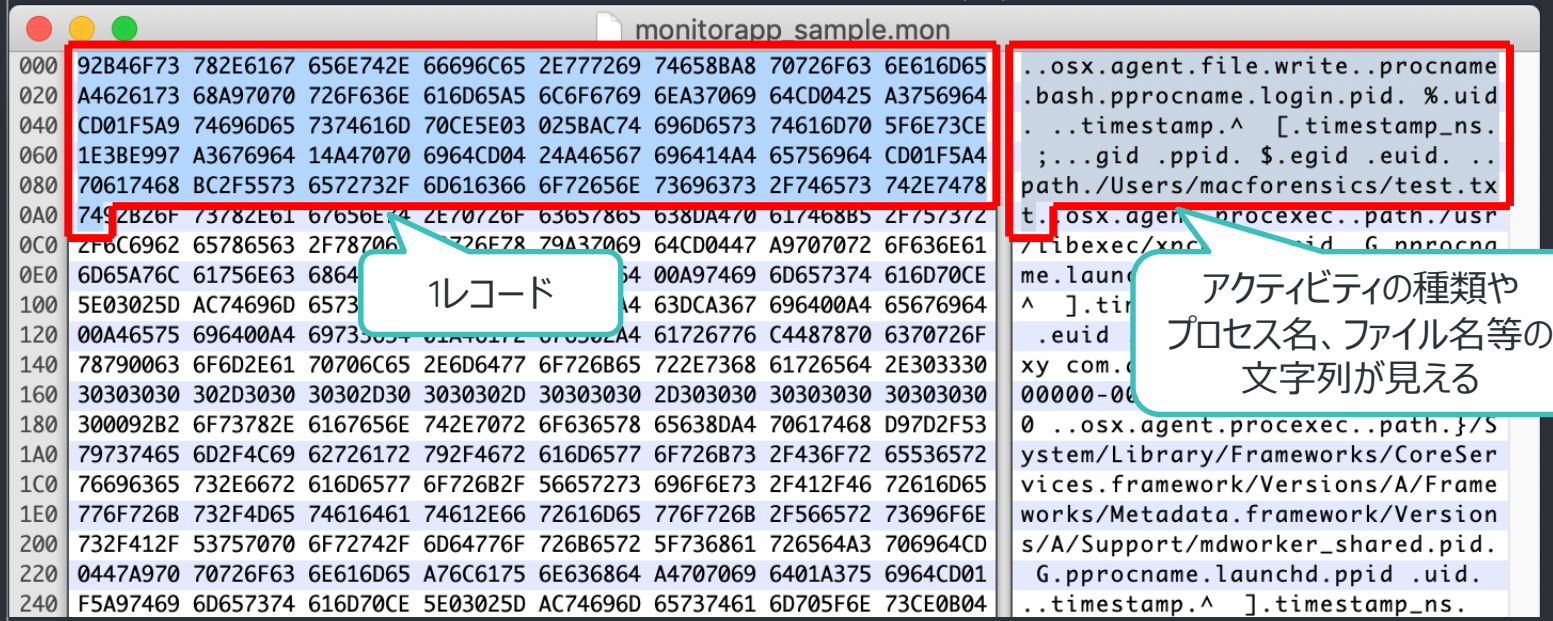


Monitor.appはマウスで操作しなければならないため、Norimaciから
AppleScriptを使って以下の操作を行う

- Monitor.app起動／終了
- アクティビティ監視開始／停止
- アクティビティデータ保存

Norimaciの実装 (8) – Monitor.appのデータ

- Monitor.appのデータはバイナリ
  - データフォーマットの詳細はAppendix 2参照



Norimaciの実装 (9) – Monitor.appのデータ変換

- monitorappconv.py
  - アクティビティログがバイナリでは扱いにくいため、JSONフォーマットに変換する


```
{"record_num": 0, "record_type": "file_write", "procname": "bash", "pprocname":
"login", "pid": 1061, "uid": 501, "timestamp": 1577255515, "timestamp_ns":
507242903, "gid": 20, "ppid": 1060, "egid": 20, "euid": 501, "path":
"/Users/macforensics/test.txt"}
{"record_num": 1, "record_type": "procexec", "path": "/usr/libexec/xpcproxy",
"pid": 1095, "pprocname": "launchd", "ppid": 1, "uid": 0, "timestamp": 1577255517,
"timestamp_ns": 178545628, "gid": 0, "egid": 0, "euid": 0, "is64": 1, "argc": 2,
"argv": "xpcproxy¥u0000com.apple.mdworker.shared.03000000-0000-0000-0000-00000000
0000¥u0000"}
```

Norimaciの実装 (10) – アクティビティログの解析 (1)

- Monitor.appはシステム全体のアクティビティを記録する
- 検体と関連性が低いプログラムや定常的に作成されるファイルのアクティビティをフィルタリングする
  - Spotlightなどのデーモンや.DS_Storeのファイル操作をホワイトリストとして定義する
  - リストの更新頻度は少ないため、ツールの中にハードコーディングしている

Norimaciの実装 (11) – アクティビティログの解析 (2)

- ホワイトリストの定義（一部）
  - JSONのエレメントでフィルタを設定する
  - 正規表現使用可能

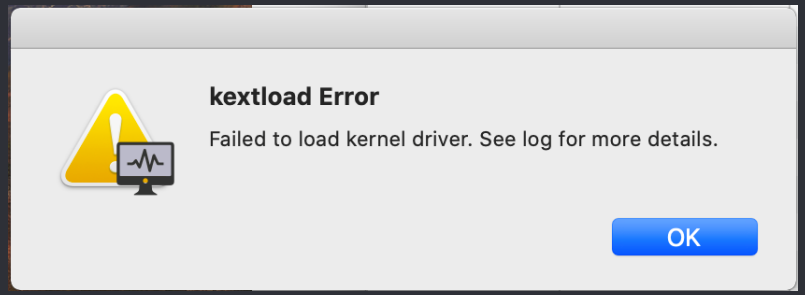
```
50  whitelist_process = [
51      {'record_type': 'info'},
52      {'record_type': 'procexec', 'path': r'/usr/libexec/xpcproxy', 'ppid': 1},  # for Monitor.app
53      {'record_type': 'procexec', 'procname': r'/sbin/launchd', 'path': r'xpcproxy', 'ppid': 1},  # for OpenBSM
54      {'record_type': 'procexec', 'path': r'(/System/Library/CoreServices/)?iconservicesagent?', 'ppid': 1},
55      {'record_type': 'file_write', 'procname': r'(/System/Library/CoreServices/)?iconservicesagent?', 'ppid': 1},
56      {'record_type': 'file_rename', 'procname': r'(/System/Library/CoreServices/)?iconservicesagent?', 'ppid': 1},
57      {'record_type': 'folder_create', 'procname': r'(/System/Library/CoreServices/)?iconservicesagent?', 'ppid': 1},
58      {'record_type': 'folder_delete', 'procname': r'(/System/Library/CoreServices/)?iconservicesagent?', 'ppid': 1},
59      {'record_type': 'procexec', 'path': r'(/System/Library/CoreServices/)?iconservicesd', 'ppid': 1},
60      {'record_type': 'file_write', 'procname': r'(/System/Library/CoreServices/)?iconservicesd', 'ppid': 1},
61      {'record_type': 'file_rename', 'procname': r'(/System/Library/CoreServices/)?iconservicesd', 'ppid': 1},
62      {'record_type': 'procexec', 'path': r'/usr/libexec/periodic-wrapper', 'ppid': 1},
63      {'record_type': 'file_write', 'procname': r'(/usr/sbin/)?cfprefsd', 'ppid': 1},
```

Norimaciの実装 (12) – レポーティング

- 2種類のレポートを生成する
- アクティビティのタイムライン
- カテゴリごとのアクティビティ
  - プロセス生成
  - ファイルアクティビティ
  - dylib読み込み
  - カーネル拡張読み込み
  - ネットワークアクティビティ
  - DNSクエリ
  - 通信先ホスト
  - パーシステンス関連ファイルへのアクセス
  - TTYオープン、クローズ

Monitor.appがmacOS 10.15 Catalinaで動作しない

- Monitor.appはmacOS 10.15で動作しない



- 別の方法のアクティビティ監視を検討する必要がある

macOSでアクティビティ監視に使えるフレームワーク

	SIP互換性	サポートしている macOSバージョン	アクティビティ監視として使えるか？
DTrace	✗	**Up to 10.10** (*2)	
OpenBSM	✓	**10.6 or later**	
EndpointSecurity Framework	✓	**10.15 or later**	

- 今回は、SIPと互換性があり、多くのmacOSバージョンでサポートされているOpenBSMを使用する
- EndpointSecurity Frameworkのサポート対象は現状少なすぎる
  - 今後開発が進む新しいフレームワークなので今後も採用すべきか検討は必要

(*2) SIPを無効化すれば、OS X 10.11以降でも使用可能

OpenBSM (1) – OpenBSMとは

- SolarisのBasic Security Module (BSM)と呼ばれるシステム監査用APIとファイルフォーマットのオープンソース実装
  - 主にBSD系OSで採用されている
  - McAfeeによってmacOSに移植された
  - システムコールとその引数や返り値、プロセスのIDやEUID等を取得することができる
  - システムコールをトレースすることでアプリケーションの動作を監視することができる
- Monitor.appと同じ内容のアクティビティ監視は行えない
  - ライブラリ読み込み、カーネル拡張読み込み、DNSクエリ、TTYオープン／クローズ

OpenBSM (2)

- 設定ファイル
  - /etc/security/audit_control
    - 監査対象のシステムコールのクラスやログファイルサイズを指定
  - /etc/security/audit_event
    - システムコールとクラスの対応付け
- 監査ログ取得コマンド
  - sudo praudit –ls /dev/auditpipe

OpenBSM (3) – 設定ファイル

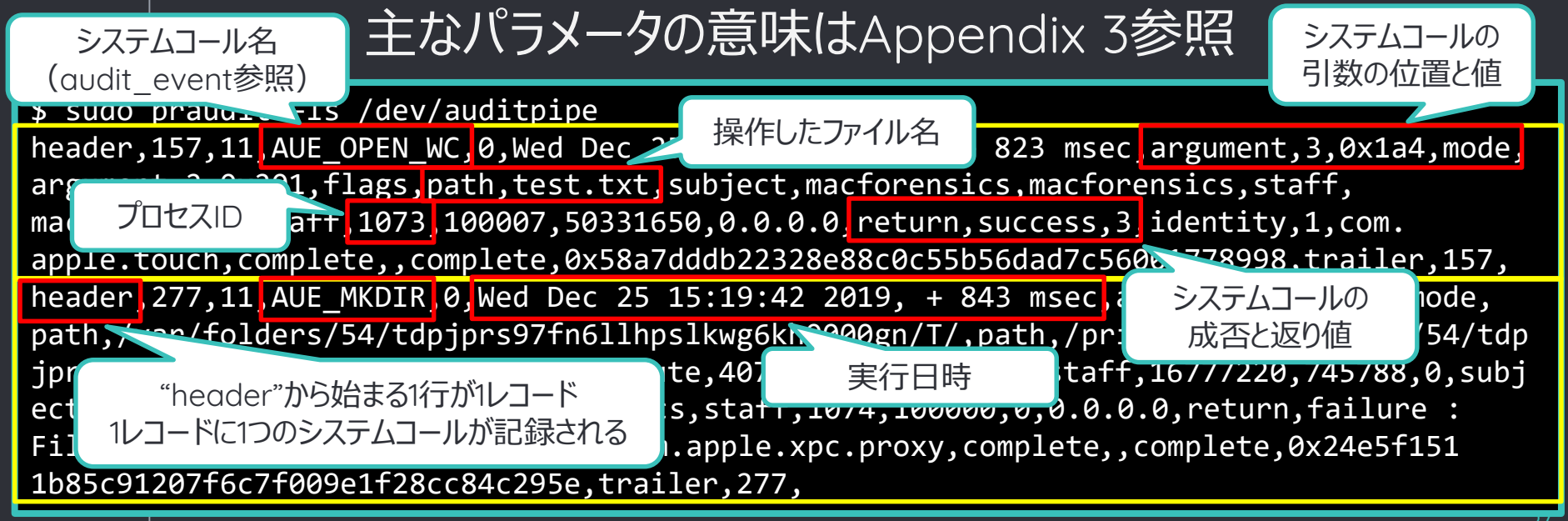
- /etc/security/audit_control

```
#
# $P4: //depot/projects/trustedbsd/openbsm/etc/audit_control#8 $
#
dir:/var/audit
flags:lo,aa,fc,fd,pc,nt,ex
minfree:5
naflags:lo,aa,fc,fd,pc,nt,ex
policy:cnt,argv
filesz:2M
expire-after:10M
superuser-set-sflags-mask:has_authenticated,has_console_access
superuser-clear-sflags-mask:has_authenticated,has_console_access
member-set-sflags-mask:
member-clear-sflags-mask:has_authenticated
```

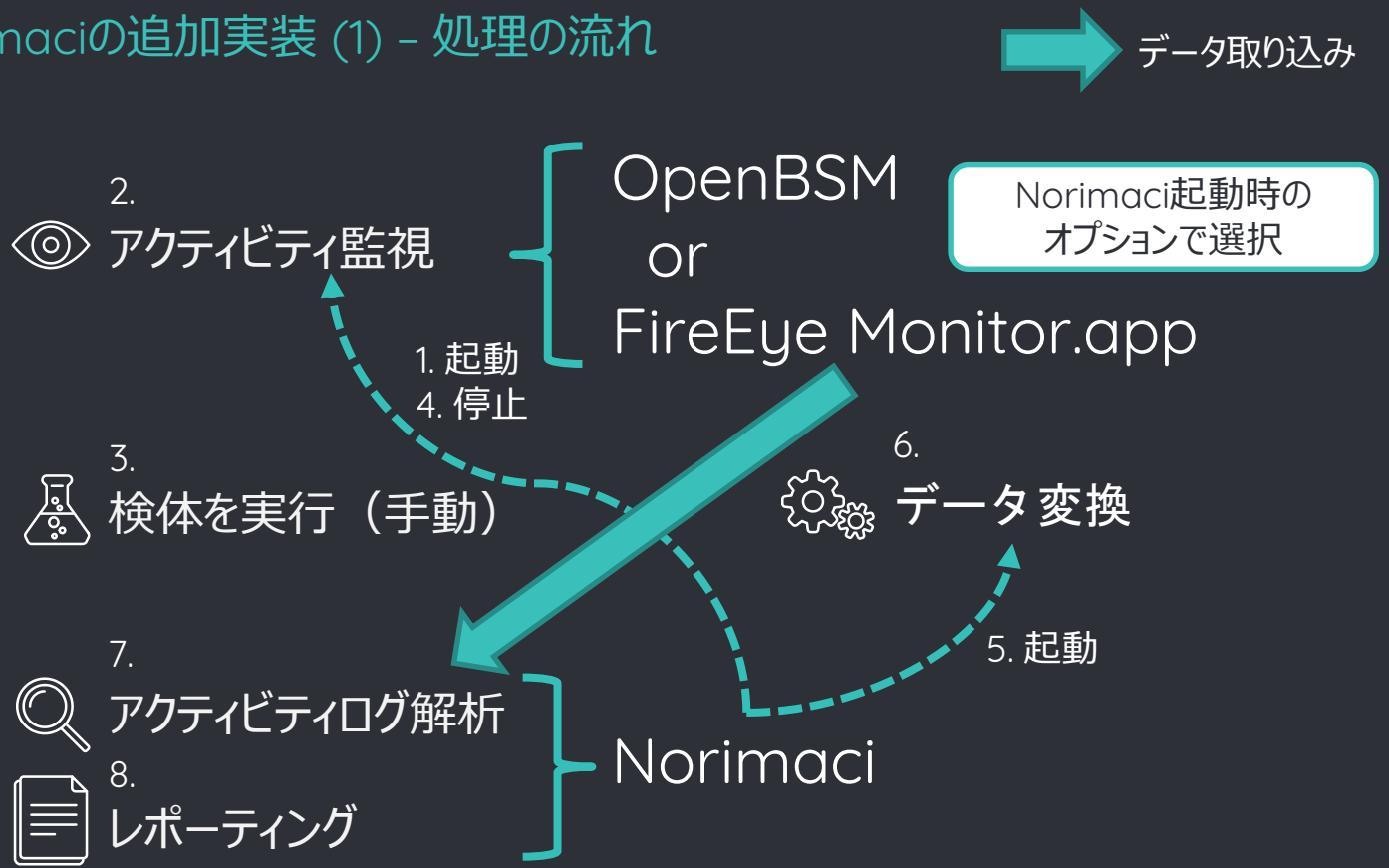
Norimaci向けにファイル作成、ファイル削除、プロセス、ネットワークのクラスを追加する（設定後に再起動が必要）

OpenBSM (4) – 監査ログ出力例

- sudo praudit –ls /dev/auditpipe
  - CSVフォーマットで出力
    主なパラメータの意味はAppendix 3参照



Norimaciの追加実装 (1) – 処理の流れ



Norimaciの追加実装 (2) – OpenBSM監査ログ変換

- openbsmconv.py
  - 監査ログをmonitorappconv互換のJSONフォーマットに変換するコンバータ


```
{"record_num": 0, "record_type": "file_write", "procname": "/usr/bin/touch",
"pprocname": "-bash", "timestamp": 1577254781, "timestamp_ns": 823000000, "pid":
1073, "ppid": 670, "uid": 501, "gid": 20, "egid": 501, "euid": 20, "path":
"test.txt"}
{"record_num": 1, "record_type": "file_write", "procname":
"/usr/libexec/biokitaggdd", "pprocname": "/sbin/launchd", "timestamp": 1577254782,
"timestamp_ns": 159000000, "pid": 271, "ppid": 1, "uid": 501, "gid": 20, "egid":
501, "euid": 20, "path": "/Users/macforensics/Library/Saved Application
State/com.apple.Terminal.savedState/window_5.data"}
```

Norimaciコマンドラインオプション (1)

- -m / --monitor
  - アクティビティ監視に使用するプログラム
  - ‘openbsm’ or ‘monitorapp’
- -o / --output
  - レポートの出力先ディレクトリ
- -j / --json
  - JSONフォーマットのアクティビティログ
  - openbsmconv.py/monitorapp.pyの出力結果
- -bl / --openbsm-log
  - CSV形式のOpenBSM監査ログ
- -p / --proclist
  - openbsmconv.pyが保存するプロセスリスト
- -ml / --monitorapp-log
  - Monitor.appのデータ

基本的にこの2つのオプション
のみを使用する

その他は既存のログを
解析する場合に使用する

Norimaciコマンドラインオプション (2)

- Norimaci実行例

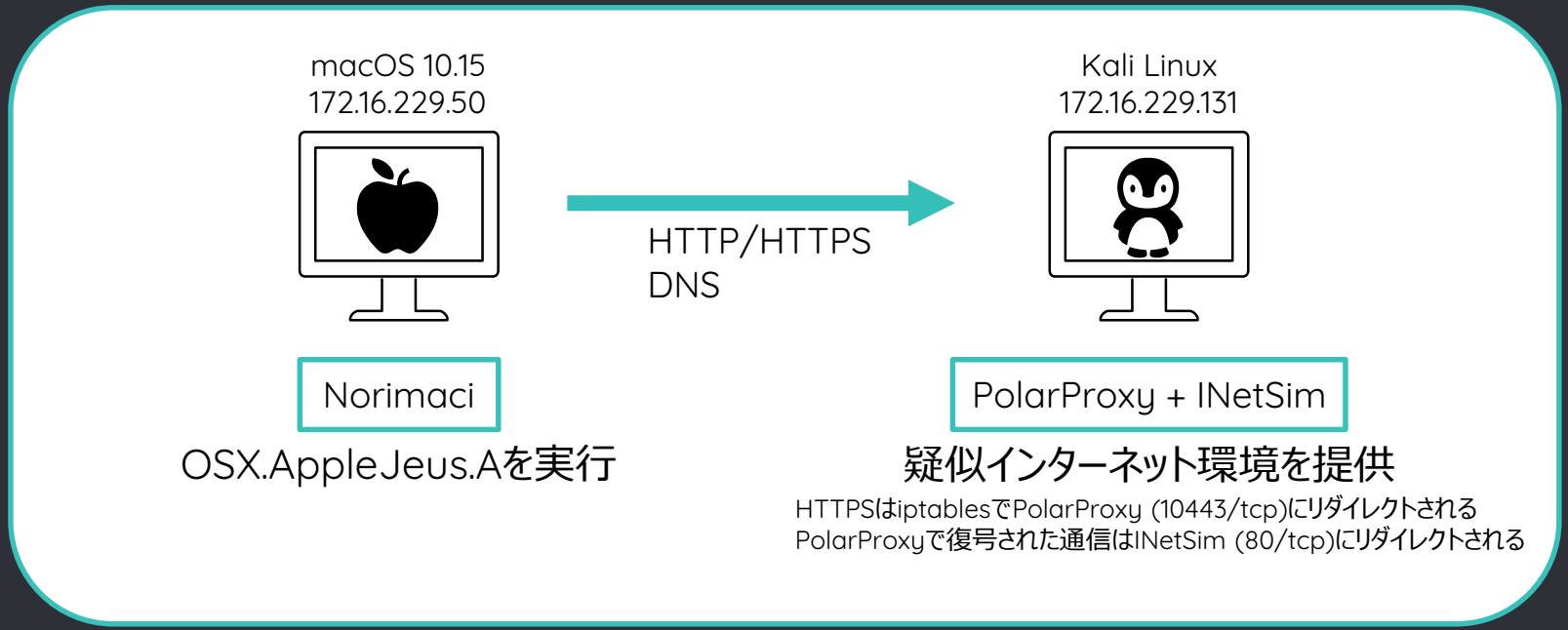
```
$ sudo python3 ./norimaci.py -m openbsm -o ./out/
```

  - OpenBSMでアクティビティ監視を行う
  - outディレクトリに解析レポートを保存する
- マルウェアを実行後、適当なタイミングでNorimaciを「Ctrl + C」で停止させる
  - アクティビティ監視アプリケーションも停止される

Norimaci Demo

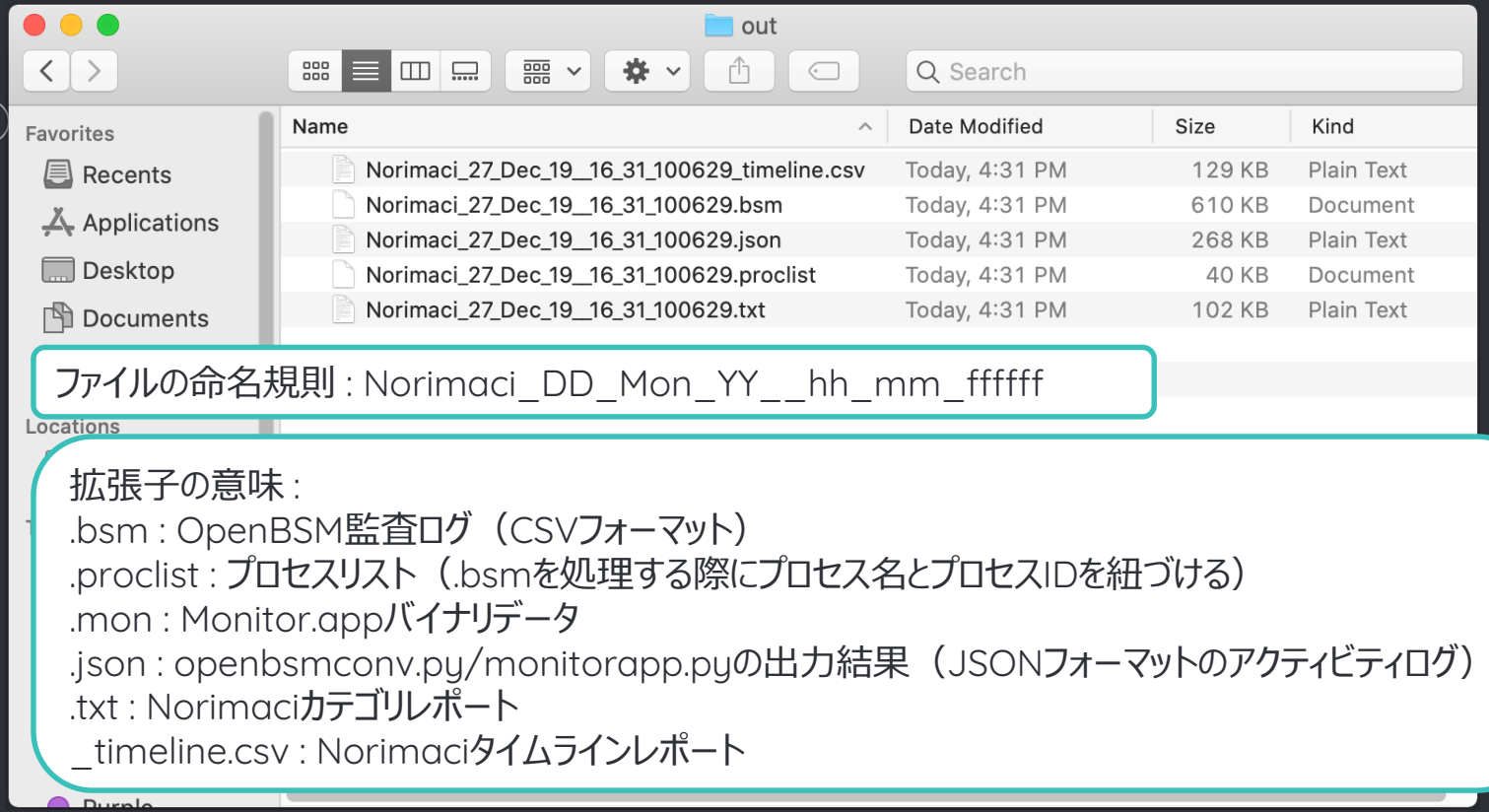
Norimaci Demo (1)

デモ環境(VMware Fusion)



DEMO MOVIE 2

Norimaci Demo (2) – 生成されるファイル



ファイルの命名規則 : Norimaci_DD_Mon_YY__hh_mm_ffffff

拡張子の意味 :
.bsm : OpenBSM監査ログ（CSVフォーマット）
.proclist : プロセスリスト（.bsmを処理する際にプロセス名とプロセスIDを紐づける）
.mon : Monitor.appバイナリデータ
.json : openbsmconv.py/monitorapp.pyの出力結果（JSONフォーマットのアクティビティログ）
.txt : Norimaciカテゴリレポート
_timeline.csv : Norimaciタイムラインレポート

Norimaci Demo (3)

- タイムラインレポート
  - 流れがつかみにくい場合はアプリケーション名やプロセスIDなどでgrepしてみる

```
2019-12-27 16:31:12.737000000,Process,CreateProcess,/sbin/launchd,1,/System/Library/Frameworks/Quartz.
framework/Frameworks/QuickLookUI.framework/Resources/QuickLookUIHelper.app/Contents/MacOS/QuickLook
UIHelper,4737
2019-12-27 16:31:12.842000000,Process,CreateProcess,/sbin/launchd,1,/System/Library/CoreServices/
DiskImageMounter.app/Contents/MacOS/DiskImageMounter,4738
2019-12-27 16:31:12.999000000,Process,CreateProcess,/sbin/launchd,1,/System/Library/PrivateFrameworks/
DiskImages.framework/Resources/hdiejectd,4739
2019-12-27 16:31:13.177000000,File,CreateFile,/System/Library/PrivateFrameworks/DiskImages.framework/
Resources/hdiejectd,4739,/var/run/hdiejectd.pid,SHA256,7897af5e7c52ddfa6d91d5bb33cd57a0c39a790f8e2cdd0
528f23516bf9deee8,,
2019-12-27 16:31:13.190000000,Process,CreateProcess,/System/Library/CoreServices/DiskImageMounter.app/
Contents/MacOS/DiskImageMounter,4738,/System/Library/PrivateFrameworks/DiskImages.framework/Resources/
diskimages-helper -uuid AE3A32E9-B721-4B23-AD33-86ABACBA3352,4740
2019-12-27 16:31:13.204000000,Process,CreateProcess,/System/Library/PrivateFrameworks/DiskImages.
framework/Resources/diskimages-helper,4740,/System/Library/PrivateFrameworks/DiskImages.framework/
Resources/diskimages-helper -uuid AE3A32E9-B721-4B23-AD33-86ABACBA3352 -post-exec 4,4741
(snip)
```

Norimaci Demo (4) – Norimaciタイムラインレポート

```
$ grep CelasTradePro Norimaci_27_Dec_19__16_31_100629_timeline.csv | grep -v installd
2019-12-27 16:31:13.770000000,File,CreateFolder,/usr/libexec/diskarbitrationd,120,/Volumes/CelasTradeP
2019-12-27 16:31:13.775000000,Process,CreateProcess,/usr/libexec/diskarbitrationd,120,/sbin/mount -t h
 -g=20 -m=755 nodev noowners nosuid /dev/disk2s1 /Volumes/CelasTradePro,4745
2019-12-27 16:31:13.787000000,Process,CreateProcess,/sbin/mount,4745,/System/Library/Filesystems/hfs.fs/Contents/
Resources/mount_hfs -u 501 -g 20 -m 755 -o nodev -o noowners -o nosuid /dev/disk2s1 /Volumes/CelasTradePro,4746
2019-12-27 16:31:29.781000000,File,RenameFile,/System/Library/PrivateFrameworks/PackageKit.framew
shove,4762,/Library/InstallerSandboxes/.PKInstallSandboxManager/BF9CB5EF-0D74-4E44-9696-2AF27D439
Root/Applications/CelasTradePro.app,/Applications/CelasTradePro.app
2019-12-27 16:31:29.847000000,Process,CreateProcess,/sbin/launchd,661,/bin/sh /tmp/PKInstallSandbox.qRskRK/Scripts/
com.celasllc.pkg.CelasTradePro.gvY1vq/postinstall /Volumes/CelasTradePro/CelasTradePro.pkg / / /,4763
2019-12-27 16:31:29.916000000,Process,CreateProcess,/bin/bash,4763,/bin/mv /Applications/Ce
Resources/.com.celastradepro.plist /Library/LaunchDaemons/com.celastradepro.plist,4764
2019-12-27 16:31:29.918000000,File,RenameFile,/bin/mv,4764,/Applications/CelasTradePro.app/Contents/Resources/
.com.celastradepro.plist,/Library/LaunchDaemons/com.celastradepro.plist
2019-12-27 16:31:29.918000000,File,Persistence,/bin/mv,4764,/Applications/CelasTradePro.app/Contents/Resources/
.com.celastradepro.plist,/Library/LaunchDaemons/com.celastradepro.plist
2019-12-27 16:31:29.929000000,Process,CreateProcess,/bin/bash,4763,/Appl
Updater CheckUpdate,4765
2019-12-27 16:31:30.182000000,File,CreateFile,/Applications/CelasTradePro.app/Contents/MacOS/Updater,4765,/private/
var/db/mds/system/mds.lock,SHA256,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,,
2019-12-27 16:31:30.653000000,Network,TCP Send,/Applications/CelasTradePro.app/Contents/MacOS/Updater,4765,
172.16.229.131,443
2019-12-27 16:31:30.671000000,Network,TCP Send,/Applications/CelasTradePro.app/Contents/MacOS/Updater,4765,
172.16.229.131,443
```

DMGをマウント

パッケージインストール

パーシステンスファイルの設置

CelasTradeProのコンポーネントが443/tcpに通信

Norimaci Demo (5) – PolarProxyとINetSimのログ

- PolarProxyログ

```
$ grep "Dec 27 16:31:30" /var/log/syslog
Dec 27 16:31:30 kali PolarProxy[85782]: [10443] 172.16.229.51 -> N/A Connection from: 172.16.229.51:50962
Dec 27 16:31:30 kali PolarProxy[85782]: [10443] 172.16.229.51 -> www.celasllc.com Connection request for:
www.celasllc.com from 172.16.229.51:50962
Dec 27 16:31:30 kali PolarProxy[85782]: Creating certificate for www.celasllc.com
Dec 27 16:31:30 kali PolarProxy[85782]: [10443] 172.16.229.51 -> www.celasllc.com Terminating TLS session from
172.16.229.51:50962 and forwarding decrypted data to 172.16.229.131:80
```

www.celasllc.com宛てのHTTPS通信

復号したHTTPSを
INetSimにリダイレクト

- INetSimログ

```
$ grep "2019-12-27 16:31:30" /var/log/inetsim/service.log
[2019-12-27 16:31:30] [71009] [dns_53_tcp_udp 71014] [172.16.229.50] connect
[2019-12-27 16:31:30] [71009] [dns_53_tcp_udp 71014] [172.16.229.50] recv: Query Type A, Class IN, Name www.celasllc.com
[2019-12-27 16:31:30] [71009] [dns_53_tcp_udp 71014] [172.16.229.50] send: www.celasllc.com 3600 IN A 172.16.229.131
[2019-12-27 16:31:30] [71009] [dns_53_tcp_udp 71014] [172.16.229.50] disconnect
[2019-12-27 16:31:30] [71009] [dns_53_tcp_udp 71014] [172.16.229.50] stat: 1 qtype=A qclass=IN qname=www.celasllc.com
[2019-12-27 16:31:30] [71009] [https_80_tcp 80530] [172.16.229.50:49946] connect
[2019-12-27 16:31:30] [71009] [https_80_tcp 80530] [172.16.229.50:49946] info: Client sent no data
[2019-12-27 16:31:30] [71009] [https_80_tcp 80530] [172.16.229.50:49946] disconnect
```

www.celasllc.comの名前解決

復号された通信内容

Norimaci Demo (6)

- カテゴリレポート

```
(snip)
Network Traffic:
==================
[TCP|UDP] /usr/sbin/syslogd:93 > 127.0.0.1:32376
[UDP] 127.0.0.1:60170 > /System/Library/CoreServices/Installer.app/Contents/MacOS/Installer:4756
[TCP] /usr/libexec/syspolicy
[UDP] /usr/sbin/mDNSResponde
[UDP] 172.16.229.131:53 > /usr/sbin/mDNSResponder:219
[TCP] /Applications/CelasTradePro.app/Contents/MacOS/Updater:4765 > 172.16.229.131:443

(snip)

Persistence:
==================
[Persistence] /bin/mv 4764 > /Applications/CelasTradePro.app/Contents/Resources/.com.celastradepro.plist
=> /Library/LaunchDaemons/com.celastradepro.plist
(snip)
```

CelasTradeProのコンポーネントが443/tcpに通信

mvでパーシステンスファイルを設置

元ファイルは不可視

コンピュータ起動時にロードされる

Norimaci Demo (7) – com.celastradepro.plist の調査

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
        "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Label</key>
        <string>com.celastradepro</string>
        <key>ProgramArguments</key>
        <array>
                <string>/Applications/CelasTradePro.app/Contents/MacOS/Updater</string>
                <string>CheckUpdate</string>
        </array>
        <key>RunAtLoad</key>
        <true/>
        <!-- Uncomment to debug
        <key>StandardOutPath</key>
        <string>/tmp/tmpctp.log</string>
        <key>StandardErrorPath</key>
        <string>/tmp/tmpctp.log</string>
        <key>Debug</key>
        <true/>
        -->
</dict>
</plist>
```

プログラムのパスと引数

ロード時に実行

4

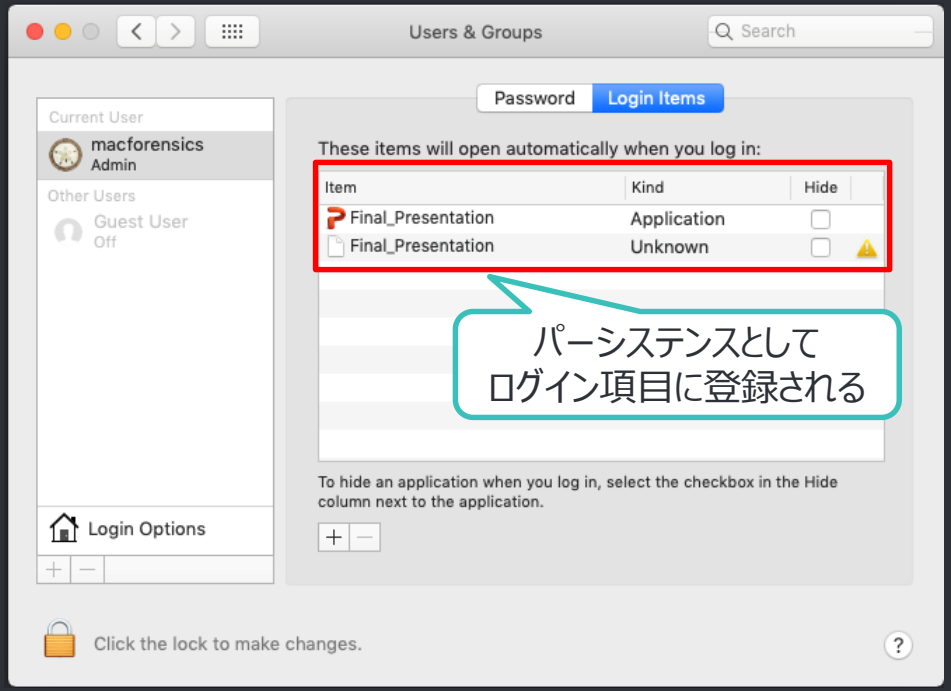
bgiparser

A parsing tool for backgrounditems.btm

backgrounditems.btm (1)

- 各ユーザの「ログイン項目」を保存しているファイル
  - ユーザログイン時に実行するアプリケーションを指定する
- macOS 10.12 Sierra以前
  - ~/Library/Preferences/com.apple.loginitems.plist
- macOS 10.13 HighSierra以降
  - ~/Library/Application Support/com.apple.backgroundtaskmanagementagent/backgrounditems.btm
- パーシステンスとして利用される
- AutoMacTCやmac_aptは、backgrounditems.btmに対応していない
- Venatorはライブ環境であれば解析できるが、オフライン解析はできない

- backgrounditems.btm (2)
  - OSX.WindTail on macOS 10.14



backgrounditems.btm (3)

```
$ file backgrounditems.btm
backgrounditems.btm: Apple binary property
$ plutil -p backgrounditems.btm
{
  "$archiver" => "NSKeyedArchiver"
  "$objects" => [
    0 => "$null“
    1 => {
      "$class" => <CFKeyedArchiverUID 0x7ffe7b7077c0 [0x7fff8c8888e0]>{value = 28}
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ffe7b707660 [0x7fff8c8888e0]>{value = 2}
        1 => <CFKeyedArchiverUID 0x7ffe7b707680 [0x7fff8c8888e0]>{value = 3}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ffe7b707740 [0x7fff8c8888e0]>{value = 4}
        1 => <CFKeyedArchiverUID 0x7ffe7b707760 [0x7fff8c8888e0]>{value = 5}
      ]
    }
    2 => "version“
    3 => "backgroundc
    4 => 2
(snip)
```

Binary plistフォーマット

NSKeyedArchiverフォーマットで保存されている

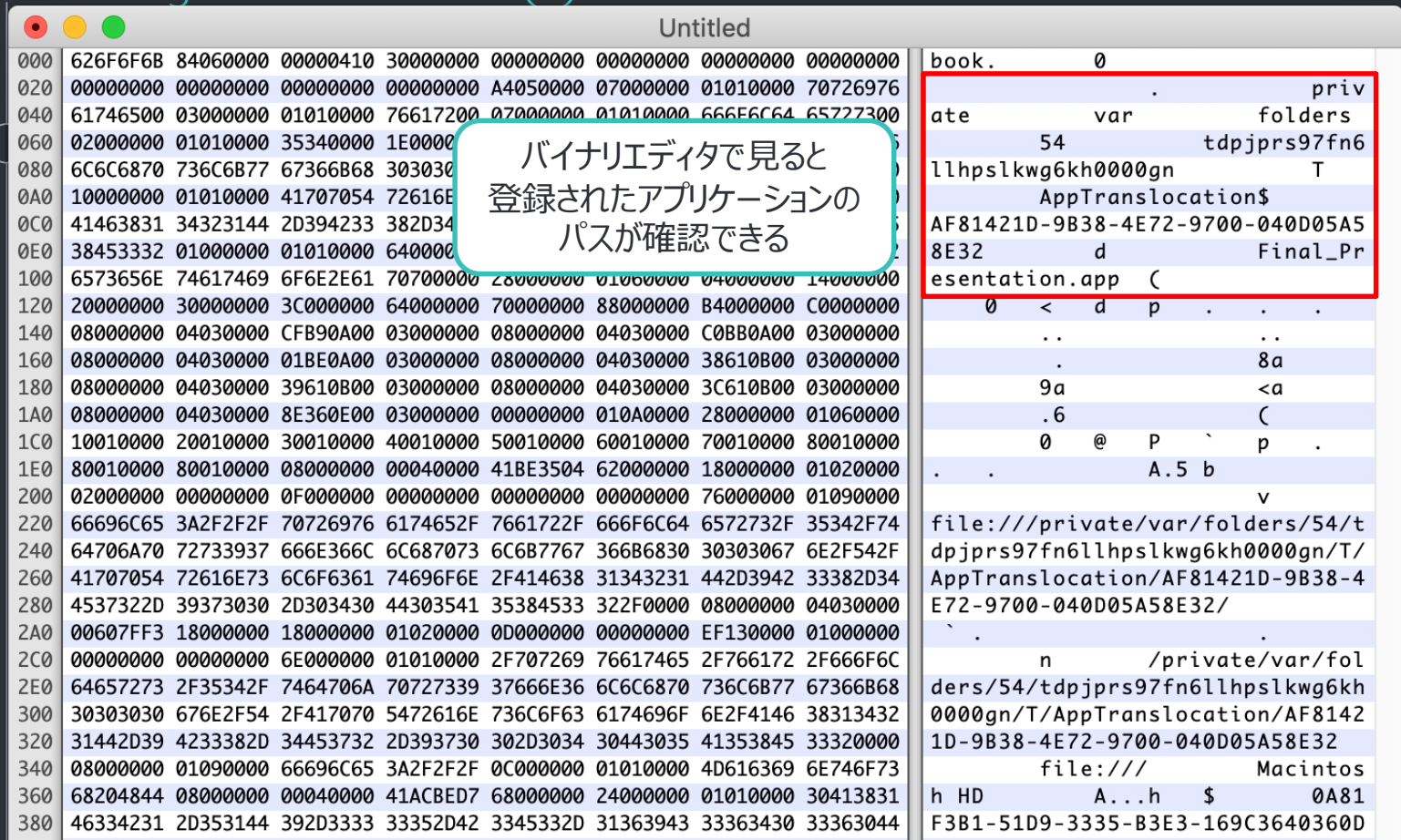
NSKeyedArchiverの解析方法
https://www.mac4n6.com/blog/2016/1/1/manual-analysis-of-nskeyedarchiver-formatted-plist-files-a-review-of-the-new-os-x-1011-recent-items

backgrounditems.btm (4)

```
(snip)
    12 => {
      "$class" => <CFKeyedArchiverUID 0x7ffe7b708040 [0x7fff8c8888e0]>{value = 15}
      "data" => <CFKeyedArchiverUID 0x7ffe7b708060 [0x7fff8c8888e0]>{value = 14}
      "identifier" => <CFKeyedArchiverUID 0x7ffe7b708080 [0x7fff8c8888e0]>{value = 13}
    }
    13 => {
      "$class" => <CFKeyedArchiverUID 0x7ffe7                                  = 9}
      "NS.uuidbytes" => <9f70e89b 9420428d aa
    }
    14 => <626f6f6b 84060000 00000410 30000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 a4050000 07000000 01010000 70726976 61746500 03000000
01010000 76617200 07000000 01010000 666f6c64 65727300 02000000 01010000 35340000 1e000000
01010000 7464706a 70727339 37666e36 6c6c6870 736c6b77 67366b68 30303030 676e0000 01000000
01010000 54000000 10000000 01010000 41707054 72616e73 6c6f6361 74696f6e 24000000 01010000
41463831 34323144 2d394233 382d3445 37322d39 3730302d 30343044 30354135 38453332 01000000
01010000 64000000 16000000 01010000 46696e61 6c5f5072 6573656e 74617469 6f6e2e61 70700000
28000000 01060000 04000000 14000000 20000000 30000000 3c000000 64000000 70000000 88000000
b4000000 c0000000 08000000 04030000 cfb90a00 03000000 08000000 04030000 c0bb0a00 03000000
08000000 04030000 01be0a00 03000000 08000000 04030000 38610b00 03000000 08000000 04030000
(snip)
```

登録されたアプリケーションの
パス等が保存されているキー

backgrounditems.btm (5)



backgrounditems.btm (6)

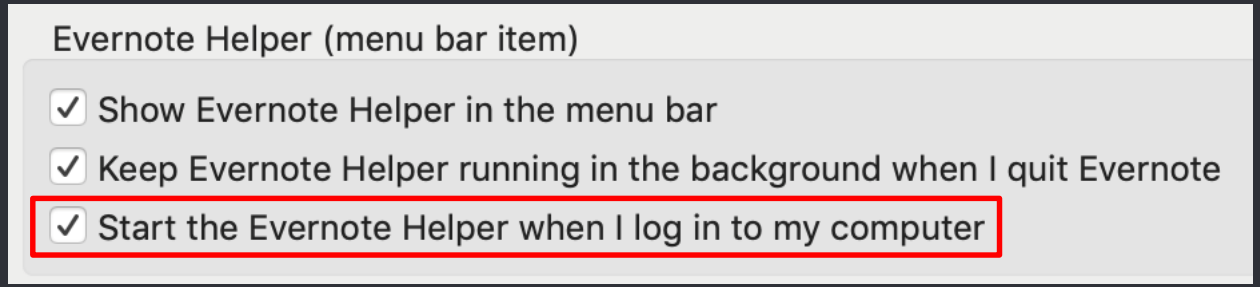
- このデータはBookmarkData構造体
  - BookmarkData構造体のパースの仕方
    - http://michaellynn.github.io/2015/10/24/apples-bookmarkdata-exposed/
- bgiparser.py
  - backgrounditems.btmを解析して、登録されているアプリケーションの名前とパスをJSONフォーマットで出力する

- backgrounditems.btm (7)
  - bgiparser.pyの実行結果

Gatekeeper Path Randomizationにより
テンポラリディレクトリのパスが設定されているため、
パーシステンスの設定としては失敗している
（ユーザログイン時に実行されない）

```
[
    {
        "name": "Final Presentation",
        "path": "/private/var/folders/54/tdpjprs97fn6llhpslkwg6kh0000gn/T/AppTrans
location/AF81421D-9B38-4E72-9700-040D05A58E32/d/Final_Presentation.app"
    },
    {
        "name": "Final_Presentation",
        "path": "/private/var/folders/54/tdpjprs97fn6llhpslkwg6kh0000gn/T/AppTrans
location/8212B23C-D9C2-4C66-B478-1C30A136DF06/d/Final_Presentation.app"
    }
]
```

- backgrounditems.btm (8)
  - アプリケーションで「ログイン時に起動」を設定すると、nameを空で設定するものがある




```
 {
        "name": "",
         "path": "/Applications/Evernote.app/Contents/Library/LoginItems/Evernote
Helper.app"
    }
```

アプリケーション名が空で設定されている

5 Future Work

Future Work

- macOS Artifact Collector
  - 解析ツールとの互換性を高める
  - Python 3対応
    - 将来的にmacOSにPythonが同梱されなくなるため、Python 2.7系で実装する必要性がなくなる
- Norimaci
  - Noribenが提供している機能の実装を目指す
    - Yara
    - VirusTotal
  - macOSバージョンアップに伴って、アクティビティ監視ツールを定期的に見直す

6 まとめ

まとめ

- macOS Artifact Collector
  - 解析ツールが対応していないアーティファクトも収集可能
  - USBメモリ等から起動可能
  - 各種解析ツールで解析できるディスクイメージに変換可能
  - https://github.com/mnrkbys/macosac
- Norimaci
  - 多くのバージョンのmacOSで動作可能
  - プロセス生成、ファイルアクティビティ、ネットワークアクティビティ等をレポーティング可能
  - 実際のマルウェアのアクティビティをレポートで確認できた
  - https://github.com/mnrkbys/norimaci
- bgiparser
  - macOS 10.13以降のログイン項目の解析が可能
  - https://github.com/mnrkbys/bgiparser

Thank you for listening!

Any questions?

Appendix 1

Default categories of macOS Artifact Collector

Default categories of macOS Artifact Collector (1)

- MRU (Most Recently Used)
  - mru
  - msoffice
  - quicklook : Can not access >= macOS 10.14
  - spotlight
- Execution History
  - bash
  - zsh
  - cmdhistory
  - ard
  - terminal
- Auto Execution
  - autoruns

Default categories of macOS Artifact Collector (2)

- Logs
  - asl
  - syslog
  - utmpx
  - unifiedlogs
- Usage Statistics
  - coreanalytics
  - knowledgec_db
  - netusage_db
- Application List
  - installhistory
  - applist

Default categories of macOS Artifact Collector (3)

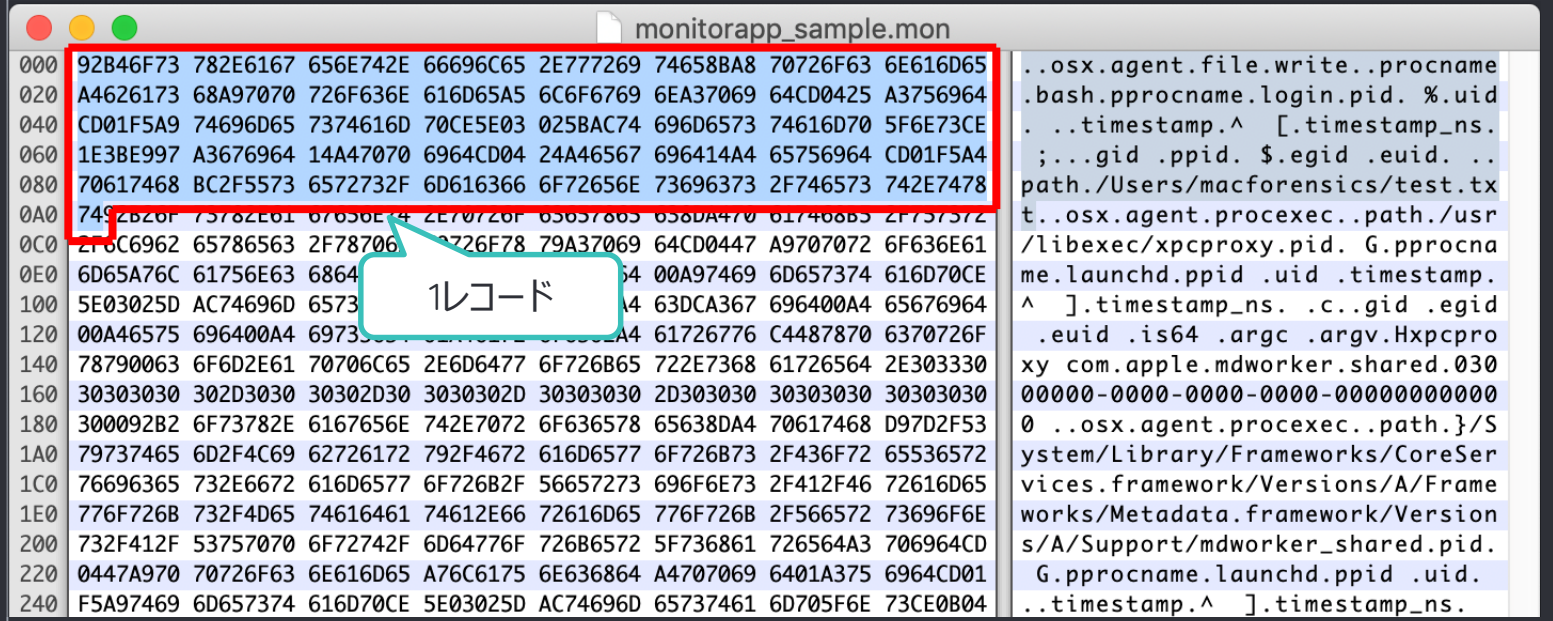
- SSH
  - ssh
- Quarantines
  - quarantines
- FSevents
  - fsevents
- Miscellaneous
  - systeminfo
  - users
  - Networking
- Preparing more categories…

Appendix 2

Monitor.app data format

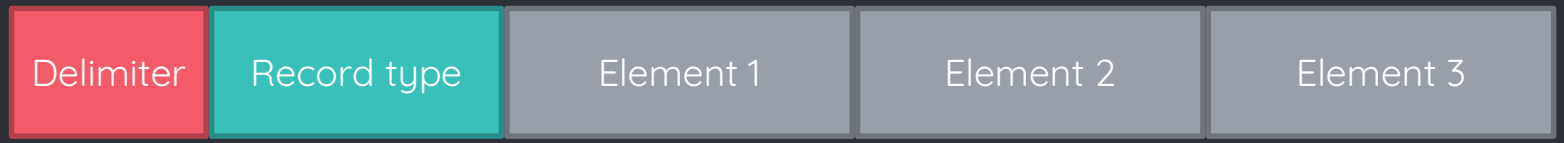
Monitor.appデータレコードフォーマット (1)

- 下図の青い部分が1レコードを表す
- 文字列で記録されるフィールドが多いため、おおよその内容は見て取れるが、プログラム処理するにはフォーマットを理解する必要がある



Monitor.appデータレコードフォーマット (2)

- レコードフォーマット
  - レコードデリミタで始まり、直後にレコードタイプが記録される
  - レコードタイプに必要なエレメントが連続して記録される



Monitor.appデータレコードフォーマット (3)

- レコード区切り
  - 1レコードの始まりを表す
  - これだけでは誤検知が発生するので、レコードタイプ文字列に含まれる “osx.agent” と併せて判別する

Record delimiter	Meaning
92	Record delimiter

Appendix 2内の表中の数字は16進数を表す

Monitor.appデータレコードフォーマット (4)

- レコードタイプ
  - 該当レコードの種類を表す
  - 最初の1バイトは文字列長を0xA0でXORした値

Record type	Meaning
AE osx.agent.info 81	Information
B4 osx.agent.file.write 8B	File write
B5 osx.agent.file.rename 8C	File rename
B4 osx.agent.dylib.load 87	Dylib load
B2 osx.agent.procexec 8D	Process execution

- レコードタイプ（続き）

Record type	Meaning
BB osx.agent.socket.connection 8A	Socket
B3 osx.agent.kext.load 87	Kext load
BC osx.agent.socket.dns.request 86	DNS query
BA osx.agent.socket.dns.reply 86	DNS reply
AD osx.agent.tty 87	TTY create/close

Monitor.appデータレコードフォーマット (6)

- エレメントタイプ（共通）
  - レコード内の要素（プロセス名、PID等）を表す
  - 最初の1バイトは文字列長を0xA0でXORした値

Element type	Meaning
A8 procname	Process name
A9 pprocname	Parent process name
A3 pid	Process id
A4 ppid	Parent process id

◦ エレメントタイプ（共通）続き

Element type	Meaning
A3 uid	User id
A3 gid	Group id
A9 timestamp	Timestamp (UTC)
AC timestamp_ns	Timestamp nanoseconds (UTC)
A4 euid	Effective user id
A4 egid	Effective group id
A5 bytes	Byte array

Monitor.appデータレコードフォーマット (8)

- エレメントデータタイプ
  - エレメントタイプ直後の1バイトで、格納されるデータタイプとデータ長（ビッグエンディアン）を表す
  - 以下の表にない場合、エレメントタイプ直後の1バイトを数値データとして扱う

Element data type	Meaning
CC XX	Value of 1-byte (XX)
CD XX XX	Value of 2-byte (XX XX)
CE XX XX XX XX	Value of 4-byte (XX XX XX XX)
C4 XX <byte array>	Byte array length of 1-byte value (XX)
D9 XX <string>	String length of 1-byte value (XX)
DA XX XX <string>	String length of 2-byte value (XX XX)

Monitor.appデータレコードフォーマット (9)

- エレメントタイプ（インフォメーション）

Element type	Meaning
A3 msg	Information about Monitor.app

Monitor.appデータレコードフォーマット (10)

- エレメントタイプ（ファイル／プロセス）

Element type	Meaning
A4 path	File path
A7 oldpath	Old file path
A7 newpath	New file path
A4 is64	64bit process or not
A4 argc	Number of arguments at program execution
A4 argv	String of arguments at program execution

◦ エレメントタイプ（ソケット）

Element type	Meaning
A7 version	Internet Protocol version
A9 direction	Direction of communication
A5 srctip	Source IP address
A7 srcport	Source port
A5 dstip	Destination IP address
A7 dstport	Destination port
A5 proto	Protocol

Monitor.appデータレコードフォーマット (12)

- エレメントタイプ (TTY)

Element type	Meaning
A3 dev	TTY number
A9 operation	An operation against TTY

Appendix 3

OpenBSM audit log parameters

OpenBSM監査ログパラメータ

Parameters	Meaning
argument,<POSITION>,<VALUE>,<DESC>	POSITION: A position of the argument VALUE: A value of the argument DESC: Short description
path,<PATH> (This parameter may be present multiple times in one record.)	PATH: File path
subject,-1,<EUName>,<EGName>,<UName>,<GName>,<PID>, num,num,0.0.0.0	EUName: Effective user name EGName: Effective group name UName: Real User name GName: Real Group name PID: Process ID
return,<STR>,<RETVAL>	STR: String of “success” or “failure” RETVAL: A return value of system call
Identity,1,<STR>	STR: An identity string for process

References / resources

References and resources (1) – Documents / Articles

- Get Cozy with OpenBSM Auditing
  - https://objective-see.com/talks/Wardle_ShmooCon2018.pdf
- MacOS monitoring the open source way | Dropbox Tech Blog
  - https://blogs.dropbox.com/tech/2018/04/4696/
- Audit in a OS X System
  - https://www.scip.ch/en/?labs.20150108
- Monitoring Process Creation via the Kernel (Part I) – Synack
  - https://www.synack.com/blog/monitoring-process-creation-via-the-kernel-part-i/
- Manual Analysis of 'NSKeyedArchiver' Formatted Plist
  - https://www.mac4n6.com/blog/2016/1/1/manual-analysis-of-nskeyedarchiver-formatted-plist-files-a-review-of-the-new-os-x-1011-recent-items
- Apple's BookmarkData - exposed! – mikeymikey blogs here
  - http://michaellynn.github.io/2015/10/24/apples-bookmarkdata-exposed/

References and resources (2) – Software (1)

- AutoMacTC: Automating Mac Forensic Triage
  - https://www.crowdstrike.com/blog/automating-mac-forensic-triage/
- macOS Artifact Parsing Tool
  - https://github.com/ydkhatri/mac_apt
- Venator
  - https://posts.specterops.io/introducing-venator-a-macos-tool-for-proactive-detection-34055a017e56
- CyLR
  - https://github.com/orlikoski/CyLR

References and resources (2) – Software (2)

- Noriben
  - https://github.com/Rurik/Noriben
- Monitor.app
  - https://www.fireeye.com/services/freeware/monitor.html
- FortiAppMonitor
  - https://www.fortinet.com/blog/threat-research/fortiappmonitor--a-powerful-utility-for-monitoring-system-activi.html
- PolarPorxy
  - https://www.netresec.com/?page=PolarProxy
- INetSim
  - https://www.inetsim.org/

References and resources (3) – Mac Malware Samples

- Mac Malware of 2017
  - https://objective-see.com/blog/blog_0x25.html
- The Mac Malware of 2018
  - https://objective-see.com/blog/blog_0x3C.html
- The Mac Malware of 2019 
  - https://objective-see.com/blog/blog_0x53.html

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
- R.I.P icon made by Freepik from www.flaticon.com
- USB thumb drive icon made by phatplus from www.flaticon.com