

Smartphone Malware Forensics — An Introduction

Viktor Schlüter & Janik Besendorf

Digital Security Lab - Reporter ohne Grenzen

Who we are and what we do:

Digital Security Lab

- founded 1.5 years ago
- **IT security trainings** for journalists
- **Analysis** of digital attacks
- **active (re-)search** for spyware attacks on journalists

Digital Security Lab

@

RSF **REPORTER
OHNE GRENZEN**



Kaspersky Says New Zero-Day Malware Hit iPhones—Including Its Own

On the sale of thousands



GREECE - 2021/04/26: In this photo illustration, an Apple logo seen displayed on a smartphone screen with a computer keyboard in the background. (Photo Illustration by Nikolas Joao Kokovis/SOPA Images/LightRocket via Getty Images)

Hacking Meduza: Pegasus

spying
Putin



PUBLISHED

< RESEARCH

October 6, 2023

Predator Files: Technical deep-dive into Intellexa Alliance's surveillance products

On 5 October 2023, a major global investigation – the “[Predator Files](#)” – was published exposing the proliferation of surveillance technologies around the world and the failure of governments and the European Union (EU) to properly regulate the industry. The Security Lab at Amnesty International is a technical partner in the “Predator

< ASIA AND THE PACIFIC

December 28, 2023

India: Damning new forensic investigation reveals repeated use of Pegasus spyware to target high-profile journalists



India: Damning new forensic investigation reveals repeated use of Pegasus spyware to target high-profile journalists

Forensic appendix: Pegasus zero-click exploit threatens journalists in India

Ahmed Mansoor: the poet who spoke truth to power and paid a heavy price

Amnesty International

Fundamentals

Computer vs. Smartphone Forensics



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

<https://www.flickr.com/photos/30478819@N08/36059117945>

Malware vs. Spyware



Phone malware: The significance of **exploits**



Sandbox

with exploits



without exploits

Phone malware: The significance of **exploits**





Sandbox

with exploits



without exploits

Exploits in different operating systems

		
scams / stalkerware <u>without</u> exploits	<ul style="list-style-type: none">• no sideloading (yet)• limited app permissions• jailbreaks sometimes possible	<ul style="list-style-type: none">• sideloading• broader app permissions• rooting often possible
spyware <u>with</u> exploits	<ul style="list-style-type: none">• possible to detect• very hard to get binary	<ul style="list-style-type: none">• rarely detected• little info for analysis

Mobile state actor forensics: An eternal cat and mouse game



Mobile state actor forensics: The eternal cat and mouse game



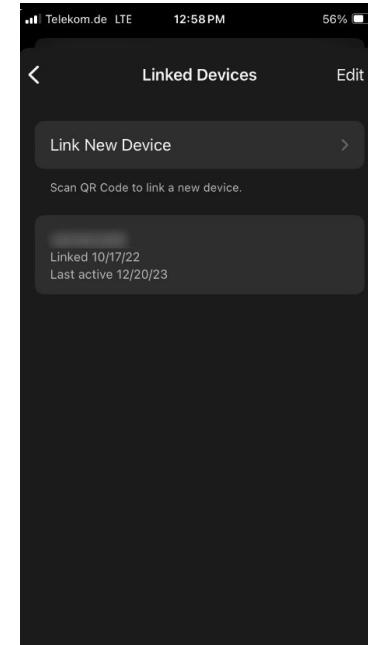
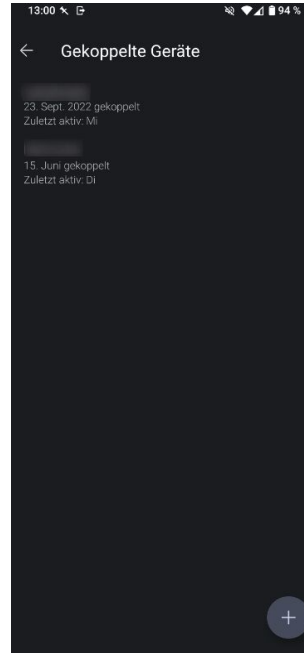
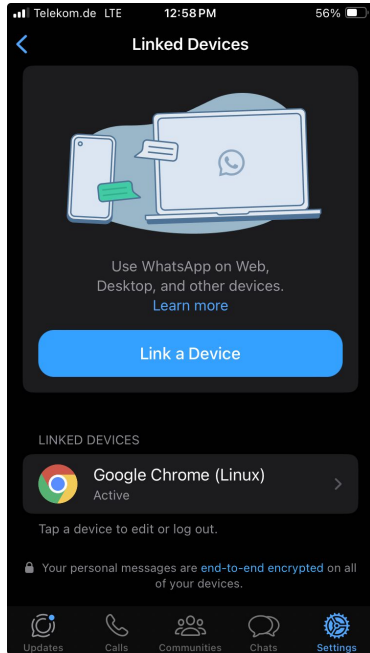
Helpful websites for cyber stalking victims

<https://antistalking.haecksen.org>

<https://stopstalkerware.org/information-for-survivors>

Methodology

Linked Devices



<https://netzpolitik.org/2021/ohne-staatstrojaner-polizei-und-geheimdienste-koennen-whatsapp-mitlesen/>

Methodology for iOS

0. optional: jailbreak device

1. Create an iPhone Backup \longrightarrow `idevicebackup2 backup --full <backup_folder>`

2. decrypt mvt Backup \longrightarrow `mvt-ios decrypt-backup -d <decrypted_dir> <backup_folder>`

3. analyse data with mvt \longrightarrow `mvt-ios check-backup -o <mvt_output> <decrypted_dir>`

4. Look through resulting data,
additional analyses



Methodology for **Android**

0. optional: root device

1. extract data with mvt → `mvt-android check-adb --output /path/to/results`

2. **download apk's** → `mvt-android download-apks --output /path/to/folder`

3. **upload apk's to Virustotal** → `MVT_VT_API_KEY=<key> mvt-android download-apks --output /path/to/folder --virustotal`

4. Look through **resulting data**,
additional analyses



Primary findings vs. secondary findings

Primary findings contain the full chain of evidence:

vendor → infrastructure
→ network traffic
→ smart phone
→ Proof of execution on phone

(plus ideally the binary)

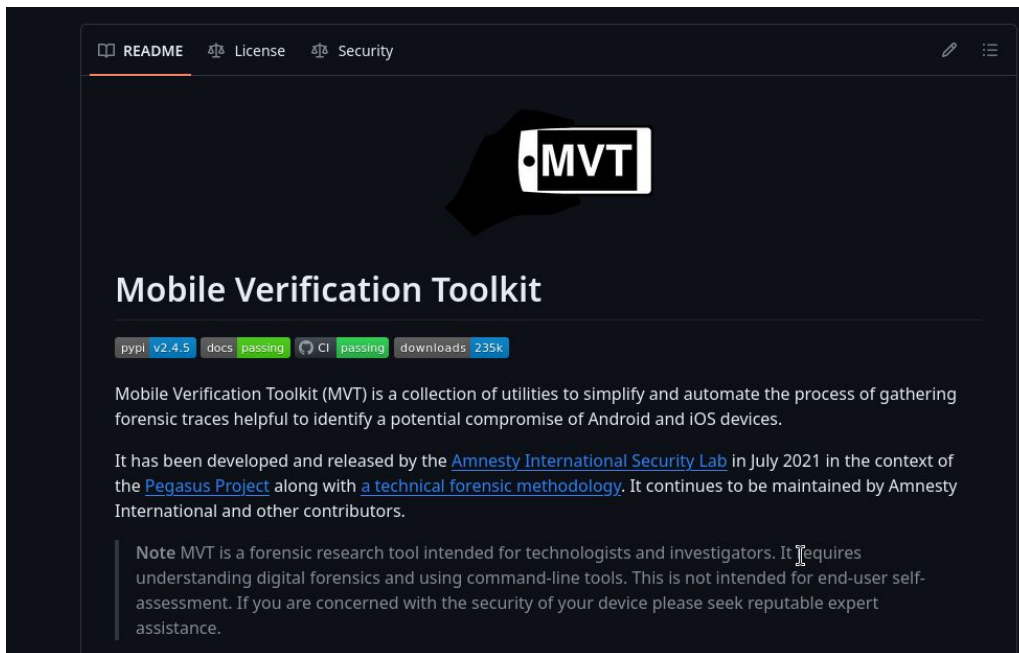
Secondary findings correlate with a primary finding:

- Identifying behaviour is the same:
 - file names
 - process names
 - Same way to “clean up”
 - same traces in logs

Indicators of Compromise (IoC)

- useful for easy secondary finding
- recommended IoC lists
 - <https://github.com/AssoEchap/stalkerware-indicators>
 - <https://github.com/mvt-project/mvt-indicators>
 - <https://github.com/AmnestyTech/investigations>

Meet the toolbox



or if you're wild, your favourite sqlite parser ;)

Traffic Analysis

- spyware needs to **transmit data** back to the surveillant
- this **traffic can be intercepted** and analyzed
- **encryption** and **obfuscation** can make this tricky

TinyCheck

- **Tool** by Kaspersky to **analyze traffic** in order to find stalkerware
- runs on a **raspberrypi** (optional)
- opens a dedicated wifi
- generates a **report** with suspicious connections
- <https://tiny-check.com>



Catching iOS malware Part 1: **Stalkerware**

- Since iOS 15.7 no jailbreak
- **Stalkerware without jailbreak difficult to impossible**
 - Has to go through app store
 - Has to comply to app store rules
 - Notifier when camera / microphone are activated
 - No access to data from other apps
- Sometimes: iCloud data crawling services



Catching iOS malware Part 1: **Stalkerware**

TCC (Transparency, Consent, and Control)

- path: /Library/TCC/TCC.db
- Interesting Table: access

in the database:

	service	client	ent_ty	th_val	auth_reason	u_ver	csreq	policy_ic	ject_ide	_object_id	ect_cc	flags	ast_modified
	Filter	Filter	Filter	Filter	Filter	Fi...	Filter	Filter	Filter	Filter	Filter	Fil...	Filter
16	kTCCServiceMicrophone	org.whispersystems.signal	0	2	2	1	NULL	NULL	NULL	UNUSED	NULL	0	1664975441

```
$ date -d @+1664975441  
Wed Oct 5 03:10:41 PM CEST 2022
```

mvt output:

```
2023-12-18 11:36:49,629 - mvt.ios.modules.mixed.tcc - INFO -  
Found client "org.whispersystems.signal" with access allowed to microphone on  
2022-10-05 13:10:41.000000 by user_consent
```



Catching iOS malware Part 1: **Stalkerware**

Data Usage

- `Library/Databases/DataUsage.sqlite`
- Interesting Table: `ZLIVEUSAGE`, `ZPROCESS`



Catching iOS malware Part 1: **Stalkerware**

Data Usage

- Library/Databases/DataUsage.sqlite
- Interesting Table: ZLIVEUSAGE, ZPROCESS

Easy to remember query:
(copied from mvt)

```
SELECT
ZPROCESS.ZFIRSTTIMESTAMP,
ZPROCESS.ZTIMESTAMP,
ZPROCESS.ZPROCNAME,
ZPROCESS.ZBUNDLENAME,
ZPROCESS.Z_PK,
ZLIVEUSAGE.ZWIFIIN,
ZLIVEUSAGE.ZWIFIOUT,
ZLIVEUSAGE.ZWWANIN,
ZLIVEUSAGE.ZWWANOUT,
ZLIVEUSAGE.Z_PK,
ZLIVEUSAGE.ZHASPROCESS,
ZLIVEUSAGE.ZTIMESTAMP
FROM ZLIVEUSAGE
LEFT JOIN ZPROCESS ON ZLIVEUSAGE.ZHASPROCESS = ZPROCESS.Z_PK
UNION
SELECT ZFIRSTTIMESTAMP, ZTIMESTAMP, ZPROCNAME, ZBUNDLENAME, Z_PK,
NULL, NULL, NULL, NULL, NULL, NULL, NULL
FROM ZPROCESS WHERE Z_PK NOT IN
(SELECT ZHASPROCESS FROM ZLIVEUSAGE);
```



Catching iOS malware Part 1: **Stalkerware**

Data Usage

- Library/Databases/DataUsage.sqlite
- Interesting Table: ZLIVEUSAGE, ZPROCESS

mvt converts this to:

```
"first_isodate": "2022-11-23"  
17:47:22.267285",  
  "isodate": "2022-11-28"  
17:20:04.212211",  
  "proc_name":  
"mDNSResponder/ph.telegra.Telegraph",  
  "bundle_id": "ph.telegra.Telegraph",  
  "proc_id": 131,  
  "wifi_in": 0.0,  
  "wifi_out": 0.0,  
  "wwan_in": 8940.0,  
  "wwan_out": 4260.0,  
  "live_id": 3110,  
  "live_proc_id": 131,  
  "live_isodate": "2022-11-23"  
17:47:22.266572"
```



Catching iOS malware Part 1: **Stalkerware**

Installed Applications

- Info.plist
- parse plists with plistutil
- bundle id is contained in
apple app store html source:



Catching iOS malware Part 1: **Stalkerware**

Installed Applications

- Info.plist
- parse plists with plistutil
- bundle id is contained in apple app store html source:

```
es\":true,\"bundleId\":\\\"ph.telegra.Telegraph\\\",\\\"hasMessagesExtension\\\":false,\\  
ph.telegra.Telegraph ^ v Highlight All Match Case
```



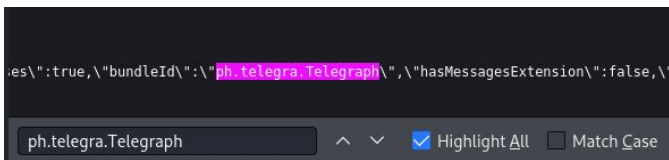
<https://apps.apple.com/de/app/telegram-messenger/id686449807>



Catching iOS malware Part 1: Stalkerware

Installed Applications

- Info.plist
- parse plists with plistutil
- bundle id is contained in apple app store html source:



<https://apps.apple.com/de/app/telegram-messenger/id686449807>

```
{
  "name": "ph.telegra.Telegraph",
  "DeviceBasedVPP": false,
  "artistName": "Telegram FZ-LLC",
  "bundleShortVersionString": "9.1.1",
  "bundleVersion": "24496",
  "com.apple.iTunesStore.downloadInfo": {
    "accountInfo": {
      "AltDSID": "001575-08-1f46f614-3fd6-4660-ad8d-1cfaadc37c58",
      "AppleID": "jb@reporter-ohne-grenzen.de",
      "DSPersonID": 20633932829,
      "DownloaderID": 0,
      "FamilyID": 0,
      "PurchaserID": 20633932829
    },
    "purchaseDate": "2022-11-23T17:47:07Z"
  },
  "gameCenterEnabled": false,
  "gameCenterEverEnabled": false,
  "genre": "Soziale Netze",
  "genreId": 6005,
  "hasMessagesExtension": false,
  "hasOrEverHasHadIAP": true,
  "iad-attribution": "0",
  "is-auto-download": false,
  "is-purchased-redownload": false,
  "isFactoryInstall": false,
  "itemId": 686449807,
  "itemName": "Telegram Messenger",
  "kind": "software",
  "launchProhibited": false,
  "rating": {
    "Label": "17+",
    "rank": 600
  },
  "redownload-params": "productType=C6price=06salableAdamId=686449807pricingParameters=STDR",
  "releaseDate": "2013-08-14T07:00:00Z",
  "s": 143443,
  "sideloadedDeviceBasedVPP": false,
  "softwareVersionBundleId": "ph.telegra.Telegraph",
  "softwareVersionExternalIdentifier": 853263055,
  "sourceApp": "com.apple.AppStore",
  "storeCohort": "10|date=1669224600006sf=1434438pgtp=Search6pgid=3d990042-1710-4d73-b58f-c3c9bc8ec0086prpg=Genre_1791836ctxt=Search6issrch=16imptyp=card&kind=iosSoftware6itpptyp=id=4"
}
```

Catching iOS malware Part 1: **Stalkerware**

The screenshot shows the GitHub repository 'stalkerware-indicators' (Public). The repository has 22 watches, 37 forks, and 217 stars. The main branch is 'master', with 2 other branches and 0 tags. The repository is managed by 'Te-k' and has 753 commits. The file structure is as follows:

File/Folder	Description	Last Update
generated	Updating generated indicator files and README	last week
tools	Adds tools	11 months ago
vendors	Updating generated indicator files and README	4 months ago
.flake8	Flake8 checks in scripts	last year
.gitignore	Update generation script and add linter	last year
README.md	Use Wayback Machine URL (#123)	last week
ioc.yaml	Updating generated indicator files and README	last month
quad9_blocklist.txt	Update readme and blocklist	last year
rules.yar	Updating generated indicator files and README	11 months ago
samples.csv	Updating generated indicator files and README	2 weeks ago
watchware.yaml	Updating generated indicator files and README	3 months ago

The repository also includes a README section titled 'Stalkerware Indicators of Compromise'. The repository statistics on the right show 217 stars, 22 watching, and 37 forks. There are no releases or packages published. The contributors section shows 12 contributors, and the languages section shows YARA at 51.9% and Python at 48.1%.

→ included in mvt

but: false positives are possible



Catching iOS malware Part 2: **State-sponsored Spyware**

Safari History, redirects (similar for Firefox, Chrome)

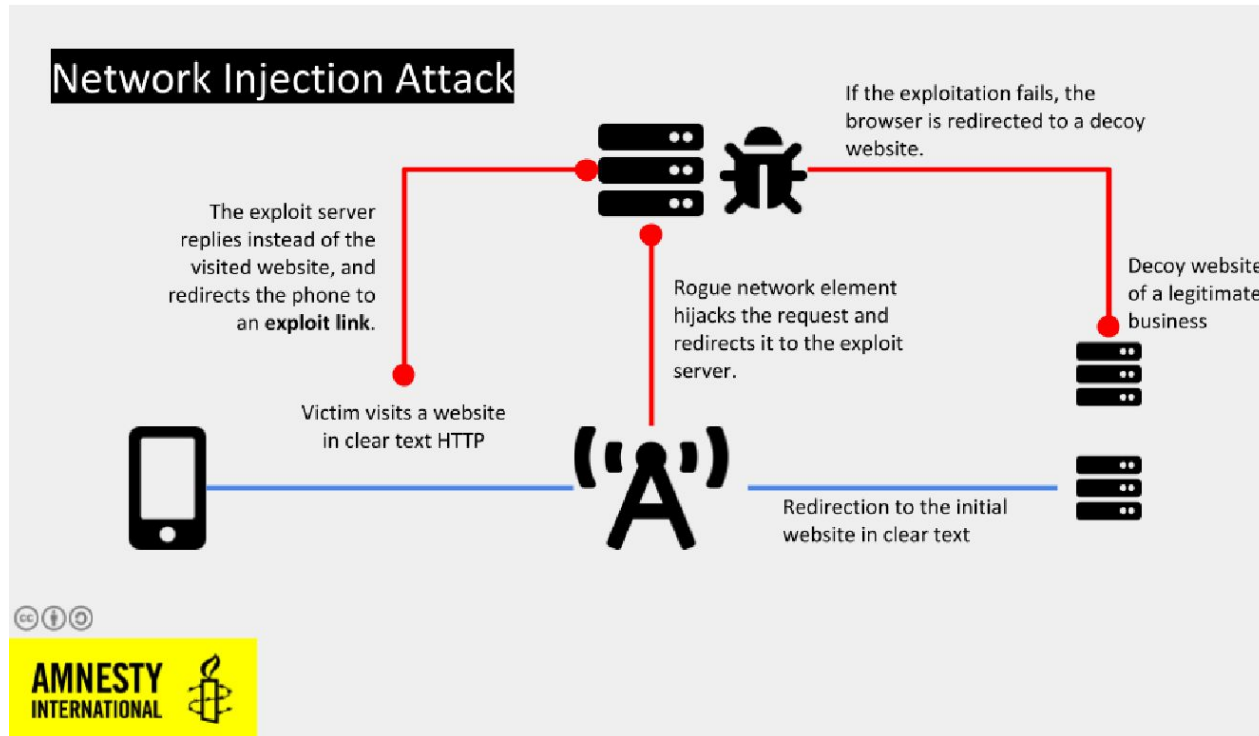
- `Library/Safari/History.db`
- `history_items, history_visits`

→ **browser redirects** quickly after the original request can be an indicator for network injection attacks

→ **one click** exploit URLs can be found here



Network Injection Attacks



Catching iOS malware Part 2: **State-sponsored Spyware**

Incomplete Trace Removals in DBs

- Which DB? It depends...

Example: (credit to Citizenlab[1])

- Pegasus deleted data usage entries in ZPROCESS but not in ZLIVEUSAGE in 2021
- By checking for inconsistencies and anomalies you can find **indirect malware traces**

[1] <https://citizenlab.ca/2021/09/forcedentry-nso-group-imessage-zero-click-exploit-captured-in-the-wild/>



Catching iOS malware Part 2: **State-sponsored Spyware**

Data Usage

- `Library/Databases/DataUsage.sqlite`
- Interesting Table: `ZLIVEUSAGE`, `ZPROCESS`



Catching iOS malware Part 2: **State-sponsored Spyware**

Data Usage

- Library/Databases/DataUsage.sqlite
- Interesting Table: ZLIVEUSAGE, ZPROCESS

Easy to remember query:
(copied from mvt)

```
SELECT
ZPROCESS.ZFIRSTTIMESTAMP,
ZPROCESS.ZTIMESTAMP,
ZPROCESS.ZPROCNAME,
ZPROCESS.ZBUNDLENAME,
ZPROCESS.Z_PK,
ZLIVEUSAGE.ZWIFIIN,
ZLIVEUSAGE.ZWIFIOUT,
ZLIVEUSAGE.ZWWANIN,
ZLIVEUSAGE.ZWWANOUT,
ZLIVEUSAGE.Z_PK,
ZLIVEUSAGE.ZHASPROCESS,
ZLIVEUSAGE.ZTIMESTAMP
FROM ZLIVEUSAGE
LEFT JOIN ZPROCESS ON ZLIVEUSAGE.ZHASPROCESS = ZPROCESS.Z_PK
UNION
SELECT ZFIRSTTIMESTAMP, ZTIMESTAMP, ZPROCNAME, ZBUNDLENAME, Z_PK,
NULL, NULL, NULL, NULL, NULL, NULL, NULL
FROM ZPROCESS WHERE Z_PK NOT IN
(SELECT ZHASPROCESS FROM ZLIVEUSAGE);
```



Catching iOS malware Part 2: **State-sponsored Spyware**

Data Usage

- Library/Databases/DataUsage.sqlite
- Interesting Table: ZLIVEUSAGE, ZPROCESS

mvt converts this to:

```
"first_isodate": "2022-11-23"
17:47:22.267285",
  "isodate": "2022-11-28"
17:20:04.212211",
  "proc_name":
"mDNSResponder/ph.telegra.Telegraph",
  "bundle_id": "ph.telegra.Telegraph",
  "proc_id": 131,
  "wifi_in": 0.0,
  "wifi_out": 0.0,
  "wwan_in": 8940.0,
  "wwan_out": 4260.0,
  "live_id": 3110,
  "live_proc_id": 131,
  "live_isodate": "2022-11-23"
17:47:22.266572"
```



Catching iOS malware Part 2: **State-sponsored Spyware**

Time stamps of iOS Backups

- `decrypt Backup` with `mvt-ios decrypt-backup`



```
└─$ ls
00 0b 16 21 2c 37 42 4d 58 63 6e 79 84 8f 9a a5 b0 bb c6 d1 dc          e6 f1 fc
01 0c 17 22 2d 38 43 4e 59 64 6f 7a 85 90 9b a6 b1 bc c7 d2 dd          e7 f2 fd
02 0d 18 23 2e 39 44 4f 5a 65 70 7b 86 91 9c a7 b2 bd c8 d3 de          e8 f3 fe
03 0e 19 24 2f 3a 45 50 5b 66 71 7c 87 92 9d a8 b3 be c9 d4 decrypted e9 f4 ff
04 0f 1a 25 30 3b 46 51 5c 67 72 7d 88 93 9e a9 b4 bf ca d5 df          ea f5 Info.plist
05 10 1b 26 31 3c 47 52 5d 68 73 7e 89 94 9f aa b5 c0 cb d6 e0          eb f6 Manifest.db
06 11 1c 27 32 3d 48 53 5e 69 74 7f 8a 95 a0 ab b6 c1 cc d7 e1          ec f7 Manifest.plist
07 12 1d 28 33 3e 49 54 5f 6a 75 80 8b 96 a1 ac b7 c2 cd d8 e2          ed f8 Status.plist
08 13 1e 29 34 3f 4a 55 60 6b 76 81 8c 97 a2 ad b8 c3 ce d9 e3          ee f9
09 14 1f 2a 35 40 4b 56 61 6c 77 82 8d 98 a3 ae b9 c4 cf da e4          ef fa
0a 15 20 2b 36 41 4c 57 62 6d 78 83 8e 99 a4 af ba c5 d0 db e5          f0 fb
```

Catching iOS malware Part 2: **State-sponsored Spyware**

Time stamps of iOS Backups

- **decrypt Backup** with `mvt-ios decrypt-backup`
- **Reconstruct backup** to original paths.
 - <https://github.com/inflex/ideviceunback>
- In the backup folder, **sort by timestamps**
 - `find -printf "%TY-%Tm-%Td %TT %p\n" | sort -n`
- If you have an **interesting time range**, look there



Catching iOS malware Part 2: **State-sponsored Spyware**

Attack artifacts: Crash logs, Attachments

- Library/SMS/**Attachments**
 - Example: .gif files in FORCEDENTRY exploits[1] (credits to Citizenlab!)
- Crashlogs can also contain interesting artifacts
 - many crashes of the same component in a short timeframe?

[1]: <https://citizenlab.ca/2021/09/forcedentry-nso-group-imessage-zero-click-exploit-captured-in-the-wild/>



Catching iOS malware Part 2: **State-sponsored Spyware**

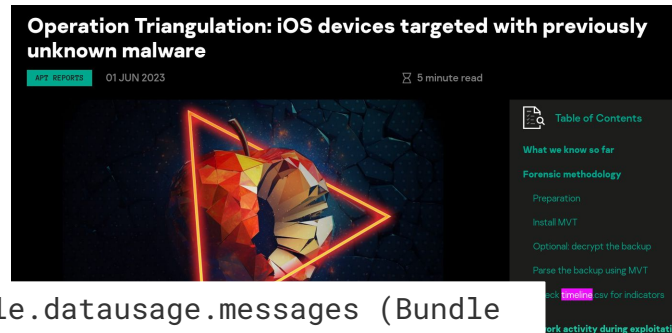
Correlation of events (in mvt timeline)

Example from Triangulation Case:

2022-09-13 10:04:11.890351Z Datausage **IMTransferAgent**/com.apple.datausage.messages (Bundle ID: com.apple.datausage.messages, ID: 127) WIFI IN: 0.0, WIFI OUT: 0.0 - WWAN IN: 76281896.0, WWAN OUT: 100956502.0

2022-09-13 10:04:54.000000Z Manifest **Library/SMS/Attachments/65/05** - MediaDomain

2022-09-13 10:05:14.744570Z Datausage **BackupAgent** (Bundle ID: , ID: 710) WIFI IN: 0.0, WIFI OUT: 0.0 - WWAN IN: 734459.0, WWAN OUT: 287912.0



Source: <https://securelist.com/operation-triangulation/109842/>, Kaspersky



Attack Vectors: **Zero Click Exploits**

First step: Tracing points of contact

- **Zero click** exploits:
 - Goal of attackers → get the device to process complex data
 - What are **interesting targets**?
 - Baseband
 - Messengers
 - Browsers
 - All apps that can be triggered to pull data / invites
 - Bluetooth, WiFi

Attack Vectors: **One Click Exploits**

First step: Tracing points of contact

- **One click exploits**
 - Attackers have to get the victim to interact with them / **click a link**
 - How can they reach the victim?
 - Messengers
 - E-mail
 - Content of web pages
- Advantage: Sometimes the victim might remember a “**strange/unusual message**”

Analysis for **Android**

→ Searching for **possible indicators**:

- Installed apps and their permissions
 - Accessibility Services
 - Full device management permissions
- Microphone and location indicators
- Root: new permissions
- running processes
- Apks and tier hashes
- check suspicious app installers
- URLs in WhatsApp messages



Analysis for **Android**: Installed Applications

- `adb shell pm list packages -u -i -f`

```
package:/data/app/~np14KhTyYF2TUaa77CbPQ==/info.metadude.android.congress.schedule-riJV6nMV3RctH6dWk9TzDw==/base.apk=info.metadude.android.congress.schedule installer=com.google.android.packageinstaller
```

- check installer
 - `com.google.android.packageinstaller`
 - None
 - `com.samsung.android.app.omcagent`



Analysis for **Android**: Installed Applications

- `adb shell pm list packages`
 - `-s` system packages
 - `-3` 3rd party packages
 - `-d` disabled packages



Analysis for **Android**: Installed Applications

- `adb shell dumpsys package com.example.package`
 - `firstInstallTime=2023-12-28 16:40:13`
 - `lastUpdateTime=2023-12-28 16:40:13`
 - `requested permissions:`
 - `android.permission.INTERNET`
 - `android.permission.ACCESS_NETWORK_STATE`
 - `android.permission.ACCESS_WIFI_STATE`
 - `android.permission.CHANGE_WIFI_MULTICAST_STATE`
 - `android.permission.CHANGE_NETWORK_STATE`



Analysis for **Android:** Intents

- Intents notify apps when certain events occur
- `adb shell dumpsys package com.example.package`

```
Action: "androidx.profileinstaller.action.BENCHMARK_OPERATION"  
android.intent.action.BOOT_COMPLETED:
```

- `android.provider.Telephony.NEW_OUTGOING_SMS`
- `android.intent.action.DATA_SMS_RECEIVED`
- `android.intent.action.NEW_OUTGOING_CALL`



Analysis for **Android**: Installed Applications MVT

- `mvt-android check-adb --module Packages --output /path/to/results/folder/`

```
[
  {
    "package_name": "com.shannon.qualifiednetworksservice",
    "file_name": "/system_ext/priv-app/ShannonQualifiedNetworksService/ShannonQualifiedNetworksService.apk",
    "installer": null,
    "disabled": false,
    "system": true,
    "third_party": false,
    "files": [
      {
        "path": "/system_ext/priv-app/ShannonQualifiedNetworksService/ShannonQualifiedNetworksService.apk",
        "md5": "76368e7eefa975ff298922cae1b98f8e",
        "sha1": "f3c7c4f58659e24f94b5d4d235b0e7204fbabb45",
        "sha256": "a722576115e1501a5f691de4e70c13ba314df7dce3264b21353642c32eb4deaa",
        "sha512": "5d00639973f71aa3ffc81161425b8a7da3200fb789ecc7928df435078b411dd024673f5f5565b4f7f478f8e34fc643206cfa874cd675400db6833458a47476d7"
      }
    ],
    "uid": "10189",
    "version_name": "Ic68d80cbe7b2a68ca4e862d127228b2de989a5ab",
    "version_code": "109 minSdk=32 targetSdk=32",
    "timestamp": "2009-01-01 01:00:00",
    "first_install_time": "2009-01-01 01:00:00",
    "last_update_time": "2009-01-01 01:00:00",
    "permissions": [
```



Analysis for **Android**: Download apk

- Get package name
 - `adb shell pm list packages`
- Get apk path
 - `adb shell pm path com.example.package`
- pull the apk
 - `adb pull`
`/data/app/~~Ihc8WADRQ_Qi0w2SqZ7-Nw==/org.fdroid.fdroid-ENnqn`
`1o_sh9NYMd7fBbx4A==/base.apk`



Analysis for **Android**: Download apk MVT

- `mvt-android download-apks -output /path/to/apks/folder/`
- `MVT_VT_API_KEY=<key> mvt-android download-apks --output /path/to/folder --virustotal`

```
MVT - Mobile Verification Toolkit
https://mvt.re
Version: 2.4.5
Indicators updates checked recently, next automatic check in 12 hours

17:19:02 INFO [mvt.android.cmd_download_apks] Retrieving list of installed packages...
17:20:45 INFO [mvt.android.cmd_download_apks] Third-party package 'com.sec.provider.mobile.android' requested 11 potentially dangerous
permissions
INFO [mvt.android.cmd_download_apks] Found non-system package with name "info.metadude.android.congress.schedule" installed by
"com.google.android.packageinstaller" on 2023-12-28 16:40:58
INFO [mvt.android.cmd_download_apks] Found non-system package with name "com.sec.provider.mobile.android" installed by "None" on
2022-07-17 01:08:10
INFO [mvt.android.cmd_download_apks] Found non-system package with name "org.fdroid.fdroid" installed by
"com.google.android.packageinstaller" on 2023-12-28 16:40:11
Looking up 3 files... 100% 0:00:01
VirusTotal Packages Detections
```


Package name	File path	Detections
info.metadude.android.congress.schedule	/data/app/~~np14KhTyYF2TUnaa77CbPQ==/info.metadude.android.congress.schedule-riJV6nMV3...	0/76
com.sec.provider.mobile.android	/data/app/~~BhAL6PzJV2A-9_hkRSHZsw==/com.sec.provider.mobile.android-aIwm6LjDyavP449XU...	39/76
org.fdroid.fdroid	/data/app/~~Ihc8WADRQ_Qi0w2SqZ7-Nw==/org.fdroid.fdroid-ENnqn1o_sh9NYM7fBbx4A==/base.a...	0/76



Analysis for **Android**: Accessibility Services

Removing Malware That Uses Accessibility to Prevent Uninstallation

Product/Version includes: Mobile Security For Enterprise , Trend Vision One , [View More](#)

 Update Date: 2023/03/10

 Article Number: 000292475

 Category: Configure, Troubleshoot

 Rating: 1

Analysis of a malware exploiting Android accessibility services



By [Roxane Suau](#) on February, 28 2023



Analysis for **Android:** Accessibility Services

- adb shell dumpsys accessibility

```
shortcut key:{}
button:{}
button target:{null}
  Bound services:{Service[label=AlcoTrack DrunkProtect Hintergrunddienst, feedbackType[FEEDBACK_SPOKEN, FEEDBACK_HAPTIC, FEEDBACK_AUDIBLE, FEEDBACK_VISUAL, FEEDBACK_GENERIC, FEEDBACK_BRAILLE], capabilities=1, eventTypes=TYPE_WINDOW_CONTENT_CHANGED, notificationTimeout=100, requestA11yBtn=false]}
  Enabled services:{{com.felixheller.alcdroid/com.felixheller.alcdroid.drunkprotect.DrunkProtectAccessibilityService_}}
  Binding services:{}
  Crashed services:{}
  Client list info:{
    Client list callbacks: 18
    Client list killed: false
    Client list broadcasts count: -1
    Registered clients:{
```



Analysis for **Android:** Accessibility Services MVT

```
$ mvt-android check-adb --module DumpsysAccessibility
INFO      [mvt.android.modules.adb.dumpsys_accessibility] Running module
DumpsysAccessibility...
INFO      [mvt.android.modules.adb.dumpsys_accessibility] Found installed
accessibility service
"com.samsung.accessibility/.universalswitch.UniversalSwitchService"
INFO      [mvt.android.modules.adb.dumpsys_accessibility] Found installed
accessibility service
"com.samsung.android.accessibility.talkback/com.samsung.android.marvin.talkba
ck.TalkBackService"
INFO      [mvt.android.modules.adb.dumpsys_accessibility] Found installed
accessibility service "com.malicious.package/.bad.Service"
```



Analysis for **Android**: Running processes

- `adb shell ps -A`

```
root      437      2      0      0 0      0 1 [ext4-fsv-conver]
logd      440      1 10901672 4848 0      0 S logd
lmkd      441      1 10817268 3336 0      0 S lmkd
system    442      1 10895544 3960 0      0 S servicemanager
system    443      1 10831408 4212 0      0 S hw servicemanager
system    444      1 10885112 2384 0      0 S vnd servicemanager
root      445      2      0      0 0      0 S [psimon]
root      446      2      0      0 0      0 S [sugov:0]
root      450      2      0      0 0      0 S [sugov:4]
root      451      2      0      0 0      0 S [sugov:6]
root      454      1 10779116 1696 0      0 S watchdogd
root      457      1 10906224 4128 0      0 S vold
root      458      1 11057356 3540 0      0 S twoshay
```



Analysis for **Android**: Running processes MVT

- `mvt-android check-adb --module Processes --output /path/to/results/folder/`

```
[
  {
    "user": "root",
    "pid": "1",
    "parent_pid": "0",
    "vsize": "2324912",
    "rss": "5136",
    "wchan": "0",
    "pc": "0",
    "name": "init"
  },
  {
    "user": "root",
    "pid": "2",
    "parent_pid": "0",
    "vsize": "0",
    "rss": "0",
    "wchan": "0",
    "pc": "0",
    "name": "[kthreadd]"
  },

```



Questions ?

Contact:

viktor.schlueter@reporter-ohne-grenzen.de

[@schluevik@chaos.social](https://chaos.social/@schluevik)

janik.besendorf@reporter-ohne-grenzen.de

[@besendorf@chaos.social](https://chaos.social/@besendorf)

Workshop: 14:15

Stage H

