

Sysdiagnose Forensic Analysis

By Sirack Hailu

April 2024

Table of Contents

[Table of Contents](#)

[Executive Summary](#)

[Methodology](#)

[Key Findings](#)

[Content Analysis](#)

[Device](#)

[Apps](#)

[Network](#)

[Visualization](#)

Executive Summary

We received a sysdiagnose file from an unknown iPhone device. We are tasked with analyzing this file to determine if the device is hacked or not. We delve into the different aspects of this file and extract indicators of compromise that will help us determine if the device is compromised. The report is broken into five different sections. Methodology discusses how the sysdiagnose file was analyzed and the tools used during this process. The key Findings section highlights the conclusion that was made from this report and the probability of a hack. The remaining section dives into the technical analysis of the different files analyzed and the observations made. Finally, we have deployed a visualization tool called timeketch to analyze the events in detail.

Methodology

While analyzing the given sysdiagnose file, we relied on different tools that helped us break down the dump and extract interesting IOCs. Specifically,

1. Manual exploration - Extract the dump & walk through each file
2. Tools - MAC OS VM, VScode, Python scripts & [timeketch](#) for visualization,
3. Prior Research & OSINT - Previous research from Amnesty International, Citizen Lab & European Commission
4. Relevant outside links are highlighted in **Blue**
5. Relevant findings & IOCs are highlighted in **Red**

Key Findings

After thoroughly analyzing the file, we believe we have found suspicious IOCs that indicate the device is probably a lab device used to inspect malicious services & payloads. Therefore the device is not hacked per se. Our best guess is someone is using this device to test/evaluate/observe suspicious IOCs. Here are the top three IOCs

1. **/private/var/db/com.apple.xpc.roleaccountd.staging/com.apple.MobileSoftwareUpdate.UpdateBrainService.16777221.192467.xpc/com.apple.MobileSoftwareUpdate.UpdateBrainService.** Processes/Services under /private/var/db or /private/var/tmp/ are prime locations for suspicious artifacts
 - a. Previous research from Amnesty international shows a device was compromised from similar a IOC
 - b. There is no indication from Apple's documentation & OSINT for the existence of a service named **com.apple.MobileSoftwareUpdate.UpdateBrainService** in iOS

2. **/payload** - Analyzing UUIDToBinaryLocations file shows a suspicious executable at the root of the file system. Such an executable is not observed on uncompromised iOS operating systems making it qualify as suspicious payload.
3. **Cydia** - This is an app that is closely related to jailbreak. Even though the mere existence of this app doesn't indicate jailbreak, It's still a high value signal that couldn't not be discredited. Here are possible scenarios
 - a. The device is indeed jailbroken by malicious actor
 - b. Device is a developer's device (or test device) as we have observed multiple apps & extensions side loaded (Numerous Install & Uninstalls). This is the likely scenario.

There are also other IOCs & observations made throughout this report. The next "Content Analysis" section discusses the rest of the indicators observed. The file dump contains multiple files, however, we selected the most important files & observations

Content Analysis

File Name - sysdiagnose_2023.12.01_16-02-47+0100_iPhone-OS_iPhone_18A8395.tar.gz
 SHA256 Hash - dec46b9194692cbc356cf238bdaf8451e6bfee70d0658ee3c7fe84883507d200
 File Type - gzip compressed data
 Last modified - Fri Dec 1 2023, 15:02:47

Unzipping the compressed tar shows a list of interesting files. Given the directory structure & some of the files listed below, the sample is a [sysdiagnose](#) file taken from an iOS device.

ASPSnapshots	crashes_and_spins	kbdebug.txt	oslog_archive_error.log	swcutil_show.txt
Personalization	disks.txt	launchctl-dumpstate.txt	otctl_status.txt	sysdiagnose.log
Preferences	error_log.txt	launchctl-list-0.txt	pcstatus.txt	system_logs.logarchive
README.txt	errors	launchctl-print-system.txt	ps.txt	tailspin-info.txt
RunningBoard	fileproviderctl.log	libtrace	ps_thread.txt	taskSummary.csv
TimezoneDB	fileproviderctl_check.log	logs	remotectl_dumpstate.txt	taskinfo.txt
WiFi	fileproviderctl_dump.log	lsaw.csstoredump	security-sysdiagnose.txt	uptime.txt
apfs_stats.txt	hidutil.plist	microstackshots	smcDiagnose.txt	vm_stat.txt
brctl	hpmDiagnose.txt	mount.txt	spindump-nosymbols.txt	
ckksctl_status.txt	ioreg	night-shift.log	summaries	

Device

The first attempt is to find what kind of device we are working with and what operating system it's running. This will help us answer few interesting questions

- What vulnerability exists on this specific device?
- What processes are common on this device?

Inspecting **/logs/SystemVersion/SystemVersion.plist**, reveals the OS version, build id & system image id.

Operating System	iPhone OS
Operating System Version	14.1
Build ID	C01DDD74-0524-11EB-9D44-686DB42BF6CD
System Image ID	ACE3B275-26F2-411F-9245-32779C5A8AAD
Build Version	18A8395
Device (Looking up 18A8395 on ipsw site reveals, the device is iPhone 6s)	Hardware - iPhone 6s Identifier - iPhone8,1

Device Language - **de_DE** (GERMANY - DE) - This was inferred from the file **/Preferences/AppleLocale_CurrentUser.txt**

Another important file that indicates the state of the device is **shutdown.log**. It's located here, **/system_logs.logarchive/Extra/shutdown.log**. This file indicates how many times a device has been restarted along with the number of processes that were running during this process. Researchers from [kaspersky](#) found the following

1. Compromised device typically take longer than usual to reboot
2. Processes under **/private/var/tmp**, **/private/var/db** typically hold the operating system from rebooting

```

13 After 1.74s, these clients are still here:
14 remaining client pid: 227 (/System/Library/PrivateFrameworks/AXAssetLoader.framework/Support/axassetsd/F37E6CF0-0357-35F3-AF9B-9BF050D1E234)
15 remaining client pid: 117 (/usr/libexec/findmydevice/712934C6-27A2-380D-B550-2760C0BEE4F6)
16 remaining client pid: 239 (/usr/sbin/BTLEServer/7B26C0AC-41E4-38C2-9274-C30BE38AA5D5)
17 remaining client pid: 226 (/System/Library/CoreServices/osanalytics-helper/7E011BA4-7C3B-3C53-A41A-C62154AA7952)
18 remaining client pid: 200 (/usr/libexec/pipelined/A840B341-512F-3AE3-9BDC-2844E29C376B)
19 remaining client pid: 82 (/System/Library/Frameworks/CoreTelephony.framework/Support/CommCenter/83F83973-329F-31C9-AD66-84F9D0FA28EF)
20 remaining client pid: 87 (/usr/libexec/cloudpaired/C2AFBF83-3256-3522-9139-C732A6CD293A)
21 remaining client pid: 0 (/kernel/40712162-F377-3097-AAF1-9866185617EF)
22 After 2.24s, these clients are still here:
23 remaining client pid: 0 (/kernel/40712162-F377-3097-AAF1-9866185617EF)
24 SIGTERM: [1683886415] All buffers flushed

```

[\(High Res Image Link\)](#)

From the above image, we can see that after 1.74 seconds, the device didn't reboot because 8 processes were still running. Then, at the 2.24s mark, 1 process was still running. This is considered to be two delays which is typical. 3 or more delays is considered as suspicious and the process under it should be inspected.

Observation

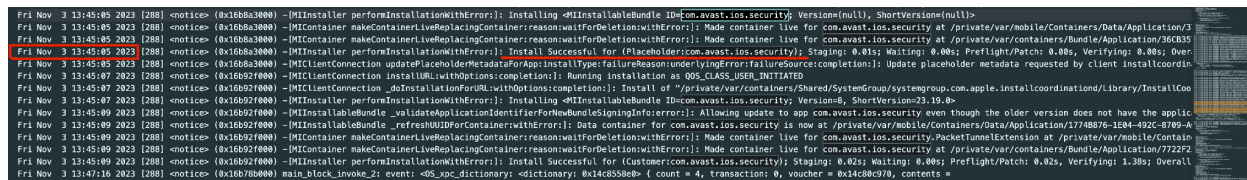
1. Device rebooted 48 times
2. 16 of them are delayed (3 or more waits)
3. No suspicious process holds these delays, hence, might be considered as false positives. Or potentially, suspicious process was not running during these restarts

Apps

There are couple of interesting logs

1. Mobile Installation Logs- This looks like it's generated when apps are installed or uninstalled. Files are located under
/logs/MobileInstallation/mobile_installation.log.x
2. Mobile Activation Logs - It's not clear what this log is about but interesting nonetheless. Files are located under
/logs/MobileActivation/mobileactivationd.log.x

Analyzing mobile installation Logs, we can safely conclude how many applications were installed/uninstalled somewhere between **Fri May 26 17:35:57 2023 & Fri Dec 1 15:32:29 2023 (approximately 6 months)**

A screenshot of a terminal window displaying system logs. The logs show a sequence of events related to the installation of the 'com.avast.ios.security' app. Key entries include 'Installing <UIInstallableBundle ID=com.avast.ios.security; Version=null, ShortVersion=null>', 'Made container live for com.avast.ios.security at /private/var/mobile/Containers/Data/Application/3...', and 'Install Successful for (PlaceholderID=com.avast.ios.security); Staging: 0.01s; Waiting: 0.00s; Preflight/Patch: 0.00s; Verifying: 0.00s; Overall: 0.01s'. The logs are timestamped with 'Fri Nov 3 13:45:05 2023'.

(High Res Image Link)

The above screenshot shows the app, **com.avast.ios.security** being installed. See mobile_installation.log.1 file ; Line 1224 - 1232. By the same token, we can look for “uninstall” string and the associated bundle identifier to come up with a list of applications that were uninstalled. The table below summarizes the observation

App Count Summary (Fri May 26 17:35:57 2023 & Fri Dec 1 15:32:29 2023)

Total Apps	24
Total Installed Apps	17
Total Uninstalled Apps	7

Apps Installed (Fri May 26 17:35:57 2023 & Fri Dec 1 15:32:29 2023)

Bundle ID / App Name	~ Install Date
Reaper - com.mist.reaper.enterprise Extensions - com.mist.reaper.enterprise.home-widget, com.mist.reaper.enterprise.content-blocker, com.mist.reaper.enterprise.dns-over-https, com.mist.reaper.enterprise.ivy-notifications-content, com.mist.reaper.enterprise.remove-metadata-extension	Nov 14, 2023
GTA Car Tracker - com.icraze.gtatracker	Oct 4, 2023
WhatsApp - net.whatsapp.WhatsApp Extensions - net.whatsapp.WhatsApp.BroadcastUploadExtension, net.whatsapp.WhatsApp.NotificationExtension, net.whatsapp.WhatsApp.Intents, net.whatsapp.WhatsApp.ServiceExtension, net.whatsapp.WhatsApp.ShareExtension, net.whatsapp.WhatsApp.TodayExtension	Nov 15 & Oct 5 2023
Avast - com.avast.ios.security Extension - com.avast.ios.security.PacketTunnelExtension	Nov 3, 2023
de.mf.horse-analysis	Nov 23, 2023

Observation

1. Reaper also has multiple extensions. Interesting ones are Content Blocker & Dns-Over-Https. The first is probably a Safari extension that can be used to block phishing URLs while the latter is a full fledged dns over https extension which can allow Reaper to monitor the device's DNS traffic & not just Safari's
2. Given the above point, Reaper might be doing the following
 - a. Anti-Phishing Capability or
 - b. Monitor the device traffic (Via DNS requests) for a prolonged period. Either to block suspicious IOCs (domains) inline or retrospectively analyze the network offline.

Apps Uninstalled (Fri May 26 17:35:57 2023 & Fri Dec 1 15:32:29 2023)

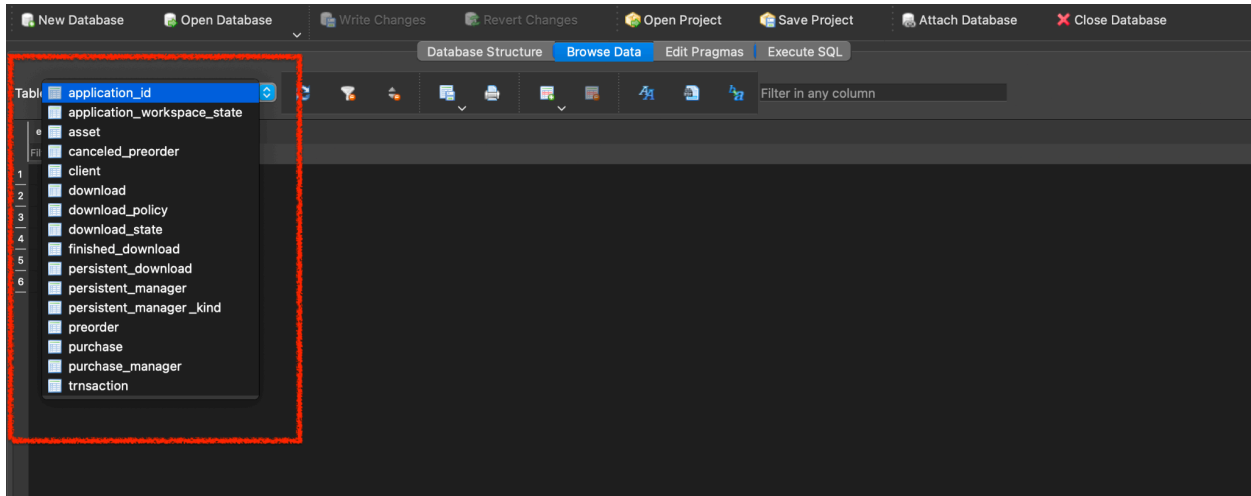
Bundle ID / App Name	~ Uninstall Date
Reaper Test - com.mist.reaper.test Extensions - com.mist.reaper.test.home-widget, com.mist.reaper.test.content-blocker, com.mist.reaper.test.dns-over-https, com.mist.reaper.test.ivy-notifications-content,	June 5, 2023
Reaper - com.mist.reaper.enterprise Extensions - com.mist.reaper.enterprise.home-widget, com.mist.reaper.enterprise.content-blocker, com.mist.reaper.enterprise.dns-over-https, com.mist.reaper.enterprise.ivy-notifications-content, com.mist.reaper.enterprise.remove-metadata-extension	June 18, 2023
de.mf.horse-analysis	Dec 1, 2023

Observation

1. Given the above apps are entirely Reaper's own applications (except de.mf.horse-analysis) and were uninstalled & installed at different points, we can reasonably conclude this might be a test device for one of the developers/testers.
2. These apps are mostly likely side-loaded. Especially **com.mist.reaper.test**, which might be a test/debug version of the enterprise Reaper app. More on this later

Finally, we can inspect the sqlite db file located under

/logs/itunesstored/downloads.28.sqlitedb to understand which applications were installed via itunes. Here is this file opened via [sqlite browser](#)



[\(High Res Image Link\)](#)

Notable Tables - **application_id**, application_workspace_state, asset, client, **download**, **purchase**, **transaction**

Observation

1. The application_id table contains 5 entries. com.apple.AppStore, com.apple.iBooks, com.apple.MobileStore, com.apple.Music & com.apple.videos. Looks like these applications are internal/preloaded.
2. Tables download & purchase are empty suggesting no application was installed from itunes store (Free & Paid). Time frame specified above.
3. Given point #2, the above applications mentioned under "App Installed Section" are indeed sideloaded.

Now that we have listed the applications, we can go ahead and see where they are installed (file system location) and along with their UUID. This will help us to build app metadata. i.e. Bundle ID, Source (Installed From), UUID, Path

The file **/logs/tailspindb/UUIDToBinaryLocations** is of type plist which contains a list of UUIDs and the corresponding paths. This file will also help us enumerate all apps that were installed (Apps under **/private/var/**)

We wrote a simple python script that transforms plist files to csv so that it's more digestible and more importantly sortable. This way we can clearly see which apps are from the system, store or side-loaded.

See output [here](#),

1	4B0E7E5A-F1F5-3893-9A7E-9B28018B9A05	/Applications/Cydia.app/Cydia
2	CE70964E-1D6E-3CB1-8B39-E1C2187690B6	/Applications/Filza.app/Filza
3	FD6E7874-C713-3A26-8503-D48B1554E40C	/Applications/MobileSlideShow.app/Plugins/PhotosReliveWidget.appex/PhotosReliveWidget
4	545BD6AD-E6C8-3D55-A675-3E8323FF1B86	/Applications/Preferences.app/Preferences
5	73AF8E81-3693-3F16-A200-F96B3B682BBC	/Applications/Screen Time.app/Plugins/ScreenTimeWidgetExtension.appex/ScreenTimeWidgetExtension
6	A779C084-8515-3C20-853A-8F445D3C9688	/Applications/Spotlight.app/Spotlight
7	875F8FE6-A510-36E4-A388-A352A5DA1894	/bin/bash
8	2FE141D8-D152-3BCB-B46E-7617E1EAE828	/Developer/Library/PrivateFrameworks/DVTInstrumentsFoundation.framework/DTSerivceHub
9	0DC56BFB-0871-3D7D-95FE-59D1B6A56A7A	/payload
10	1BB5ED99-673B-38A9-89A9-75EABF54376E	/private/var/binpack/Applications/loader.app/loader
11	6E4C54DC-1A96-374B-8AA6-156EDB587726	/private/var/containers/Bundle/Application/1346E4B7-694B-44A8-AF3F-192ACB34A168/Stocks.app/Plugins/StocksWidget.ap
12	FCD75CC3-657A-3483-B807-7A3F00AB46F1	/private/var/containers/Bundle/Application/17B80470-9285-4A6D-959A-0BEF19F155E2/MobileCal.app/MobileCal
13	0C1C1E3E-F7B3-3180-93F8-900C014783BC	/private/var/containers/Bundle/Application/17B80470-9285-4A6D-959A-0BEF19F155E2/MobileCal.app/Plugins/CalendarWidg
14	297B08D7-B499-3526-A16F-1F737764DD1E	/private/var/containers/Bundle/Application/191599D0-BA15-4915-B75A-6E26C30480E5/Weather.app/Plugins/WeatherWidget.
15	46F32BBD-EBE6-395F-A6F1-8E52A7A69834	/private/var/containers/Bundle/Application/8AECBDDF-CA80-4D2F-8B3C-07C6EE0E22EF/IvyForOrgs.app/IvyForOrgs
16	AF25ADF0-87EE-30D6-BD4D-E456B76D0B59	/private/var/containers/Bundle/Application/A866AC64-9386-4674-A5A8-DDA3ED9EC57D/Maps.app/Plugins/GeneralMapsWid
17	B8672574-6152-3B47-8405-828B9A28A48A	/private/var/containers/Bundle/Application/F7357F9C-6A91-41FD-BB1E-C8B2D38A3AC9/Tips.app/Plugins/TipsSwift.appex/Ti
18	E29AD64E-3683-3AA0-BEE9-FEEC6A5AFBDF	/private/var/containers/Bundle/Application/F90E6D12-21A0-43E7-A9BA-B9379BAA8988/SequoiaTranslator.app/Plugins/Cach
19	F1876CF3-535B-3D7F-93D3-EA2A3E45A462	/private/var/db/com.apple.xpc.roleaccounttd.staging/com.apple.MobileSoftwareUpdate.UpdateBrainService.16777221.192467.xpc/com.apple.MobileSoftwareUpdate.UpdateBrainService
20	F68F3133-A0F7-3CE6-AA51-CA3EC61AE0B9	/sbin/launchd
21	DBE88DBB-9CA1-3BD1-A5DD-767A891C6777	/System/Library/CoreServices/CacheDeleteAppContainerCaches
22	02CA8BA-6F0-3C7B-AA2A-E45EBEEDCE	/System/Library/CoreServices/CacheDeleteAppContainerCaches

[\(High Res Image Link\)](#)

Observation (See excel sheet [here](#))

1. Row #1 shows Cydia is installed. Cydia is an application used for jailbreaking iOS devices. There is a chance the device might be jailbroken
2. Row #9 shows a **very suspicious executable named payload**. This is not something that is observed in other uncompromised iOS devices. Therefore, this is most likely a foreign executable ; and potentially malicious. We can also see the process was running by inspecting **taskinfo.txt** & **spindump-nosymbols.txt** files. See screenshot below
3. Row #19 shows a binary that is located under /private/var/db. Directories like **/private/var/db** & **/private/var/tmp** are known locations for most of the commercial spywares. Additionally, Apple does not have an app named **MobileSoftwareUpdate** in iOS which makes this process suspicious. **/private/var/db/com.apple.xpc.roleaccounttd.staging/com.apple.MobileSoftwareUpdate.UpdateBrainService.16777221.192467.xpc/com.apple.MobileSoftwareUpdate.UpdateBrainService**

```
Process:      payload [35]
UUID:        0DC56BFB-0871-3D7D-95FE-59D1B6A56A7A
Path:        /payload
Architecture: arm64
Parent:      launchd [1]
UID:         0
Sudden Term: Tracked (allows idle exit)
Footprint:   864 KB
Time Since Fork: 2615s
Num samples: 9 (1-9)
Note:        1 idle work queue thread omitted

Thread 0x35c  9 samples (1-9)  priority 4 (base 4)
9 <truncated backtrace>
9 ??? (libdispatch.dylib + 85608) [0x1a0659e68]
9 ??? (libsystem_kernel.dylib + 160136) [0x1cb4f7188]
*9 ??? [0xffffffff0075320e4]

Binary Images:
0x102218000 -      ??? payload <0DC56BFB-0871-3D7D-95FE-59D1B6A56A7A> /payload
0x1a0645000 -      0x1a06c3fff libdispatch.dylib <8F32B2C5-489A-364F-B00B-5AA01AE4CF4A> /usr/lib/system/libdispatch.dylib
0x1cb4d0000 -      0x1cb4fffff libsystem_kernel.dylib <5D5E30C1-F5B9-3189-BB5F-6F5538346DFE> /usr/lib/system/libsystem_kernel.dylib
```

[\(High Res Image Link\)](#)

Network

There seems to be three log directories associated with network

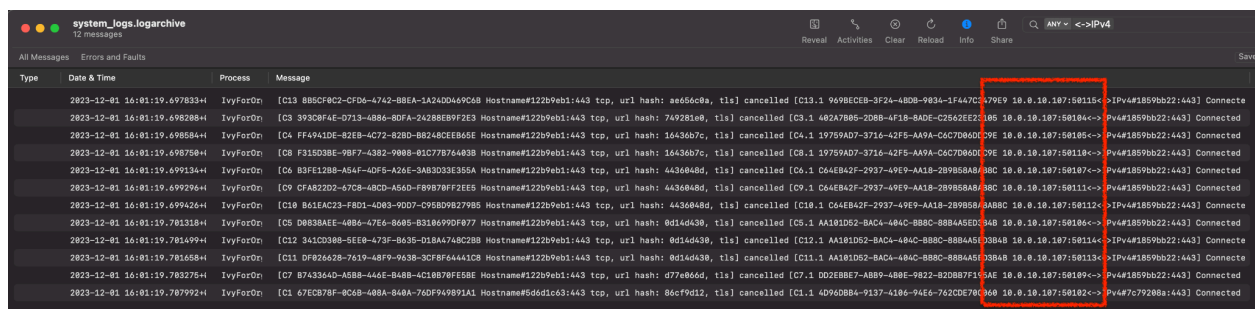
1. **/WiFi/com.apple.wifi.known-networks.plist** - This seems to be binary plist file that contains list of past wifi routers the device connected to
2. **/WiFi/com.apple.wifi.recent-networks.json** - Similar to #1 but for the last recent WiFi router connection
3. Inspect log archive file to filter out network events. **/system_logs.logarchive**
4. **/logs/networking/com.apple.networkextension.plist** & **/logs/networking/com.apple.networkextension.uuidcache.plist** - Binary plist files that contain network activity of applications & system executables

SSID	BSSID (Mac Address)	Joined Date	Location	Manufacturer
SECRT PISO 31	88:89:2f:58:58:c4	2023-06-03 15:33:32	~ Not Determined	Huawei Technologies Co.,Ltd
The Bay	ac:4c:a5:fb:56:e	2023-06-19 03:30:45	~ Not Determined	Vantiva Usa Llc
Tankstelle	dc:39:6f:71:e3:60	2023-05-26 15:36:36	~ Germany	Avm Audiovisuelles Marketing Und Computersysteme Gmbh

5G Gast Netzwerk	de:39:6f:71:e3:61	2023-05-09 10:36:29	~ Germany	
------------------	-------------------	---------------------	-----------	--

Observation

1. It's possible to get the location of the device by reverse searching the mac address on websites like <https://wigle.net>. They have an API but we used their web console since the lookup is only for 4 SSIDs. This is why apps require location permission when they are retrieving sensitive details such as SSID & BSSID
2. Looks like the user lives/works in two different regions. Germany & US. Tankstelle & 5G Gast Netzwerk are located in Germany and the companies that made the routers are based in Germany. "The Bay" is most likely located in the United states (San Francisco) and manufactured by a company called Vantiva Usa Llc
3. All of the connections were captive portal wifis, which indicate, these places might not be public places such as hotels, cafe, etc
4. Open **system_logs.logarchive** with the console app on mac and search for network events. E.g. Search for **<->IPv4**. The IP address **10.0.10.107 is local which indicates there is some kind of local VPN or tunnel (See screenshot below)**
5. There is a good indication that the DNS requests are resolved by Cloudflare. **1.1.1.1**
6. There is also a good indication that the device uses DOH (**DNS Over HTTPS**) which forces the device to encrypt its DNS requests. This will hide important DNS details from Routers, Telco providers. Good for privacy. This conclusion is made by parsing the binary plist file **/logs/networking/com.apple.networkextension.plist**



([High Res Image Link](#))

Visualization

We have spun up a timesketch instance to visualize some of the events from sysdiagnose.

EXPLOREINVESTIGATE

Search

Timelines+timesketch1.6K

Saved Searches0

Data Types0

Tags0

Graphs3

Plugins

Windows logins

Windows services

Chrome downloads

Stories+

Threat Intelligence+

Search Templates0

Sigma Rules+

Analyzer Results+

com.trailofbits.iverify.enterprise

+ ADD TIMELINE+ ADD MANUAL EVENTSELECT ALLUNSELECT ALL

timesketch1.6K

ADD TIMEFILTER

1-40 of 1614 events (0.011s)

Rows per page: 401-40 of 1614

message

2023-11-23T13:52:18.649Zcom.trailofbits.iverify.enterprise:10.10.3.1:Application added

datetime	2023-11-23T14:52:18+00:00
extra_field_1	subsystem: com.apple.appinstallation; processImageUUID: 05B9AF5C-F91D-324C-9C9C-CB1B8A46A12D; processImagePath: /System/Library/CoreServices/SpringBoard.app/SpringBoard
message	com.trailofbits.iverify.enterprise:10.10.3.1:Application added
timestamp	1700747538649598
timestamp_desc	Entry in logarchive: logEvent

2023-11-23T14:21:33.977Z[3537DD17][com.trailofbits.iverify.enterprise] Skipping receipt refresh

2023-11-23T14:22:25.918Z[FF821FB8] - NEW (com.trailofbits.iverify.enterprise evid: 860810977)

2023-11-23T14:40:43.884Z<StoreKitClient: 0x133e611d>: Client initialized with bundle ID com.trailofbits.iverify.enterprise and request bun...

2023-11-23T14:40:43.885ZStarting server purchase queue check for com.trailofbits.iverify.enterprise

2023-11-23T14:40:43.886ZStoreKitServiceConnection(383): Clearing all purchase intents for bundle identifier com.trailofbits.iveripri...

2023-11-23T14:40:44.912ZFound 0 transactions in pending queue for com.trailofbits.iverify.enterprise

[\(High Res Image Link\)](#)