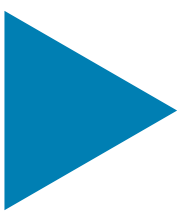
OFFENSIVE SERVICES

AWARE7

# iOS Hacking

Eine Einleitung für den Einstieg ins iOS Pentesting.



# Inhaltsangabe

# Einleitung

Im Jahr 2023 verbringen 30- bis 49-jährigen Bundesbürger täglich ca. zweieinhalb Stunden an ihrem Smartphone – bei den 16- bis 29-Jährigen sind es sogar fast drei Stunden [1]. Dabei werden die unterschiedlichsten Apps verwendet. Die Applikationen haben dabei verschiedenste Anwendungsfelder, zum Beispiel mit Freunden und Kollegen kommunizieren, Gesundheitsdaten analysieren oder Bankgeschäfte durchführen.

Das Handy ist in den letzten Jahren in die meisten Lebensbereiche des Menschen eingedrungen und ein Leben ohne Smartphone ist in weiten Teilen unserer Gesellschaft nicht mehr vorstellbar. Fahrräder, Kinderbetreuung und Smart-Home sind nur einige weitere Anwendungsfälle, für die mittlerweile Applikationen auf unseren Smartphones bereitstehen.

In Deutschland betrug der Anteil an Apple iPhones zwischen Januar und März dieses Jahres circa 34 % [2]. Circa jedes dritte verkaufte Smartphone läuft mittlerweile unter dem Apple Betriebssystem iOS. Auch hier gilt, dass jede Applikation potenziell Schwachstellen beinhalten kann. Diese können schwerwiegende Auswirkungen auf die Benutzenden haben. Das Problem ist, dass das Testen von iOS-Applikationen auf Sicherheitslücken mit einem größeren Aufwand verbunden ist, als es beispielsweise für Web-Applikationen der Fall ist. Um diese Hürde zu verringern, ist dieses Whitepaper entstanden. Es soll einen einfachen und schnellen Einstieg in das sicherheitsbezogene Testing von iOS-Applikationen geben, um darauf basierend tiefgehende Sicherheitsanalysen von mobilen Anwendungen zu ermöglichen.

Es werden die wichtigsten Werkzeuge und deren Einrichtung vorgestellt, sodass Penetrationstester einen niederschwelligen Einstieg in das Thema haben.

[1]: https://de.statista.com/statistik/daten/studie/714974/umfrage/taegliche-nutzungsdauer-von-smartphones-in-deutschland/

[2]: https://de.statista.com/statistik/daten/studie/251737/umfrage/marktanteil-des-apple-iphone-am-smartphone-absatz-in-deutschland/

# Vorbereitung

Bevor die notwendigen Tools zur Durchführung eines Penetrationstests auf einem iOS-Gerät installiert werden können, muss ein sogenannter **Jailbreak** durchgeführt werden. Dies ist das iOS-Äquivalent zum sogenannten „Rooten“ auf Android-Geräten. Durch den **Jailbreak** erlangen Penetrationstester uneingeschränkten Zugang zum System. Unter anderem ermöglicht ein solcher Zugang den Zugriff auf das Dateisystem und das Installieren von unautorisierten Applikationen. Zwischen den folgenden **Jailbreak-Arten** wird unterschieden:

**Vorbereitung des Macs**
Wir gehen in dieser Anleitung davon aus, dass zur Durchführung des Jailbreaks ein Laptop mit dem Betriebssystem macOS verwendet wird. Grundsätzlich kann der Jailbreak auch unter Linux ausgeführt werden.
Auf dem Mac werden im Verlauf dieses Leitfadens der Node.js Paketmanager npm und der macOS Paketmanager Homebrew verwendet.

Die Installationsanleitung für Homebrew findet sich unter https://brew.sh/. Nach der Installation von Homebrew, kann npm mit diesem Befehl installiert werden:

```
$ brew install node
```

**TETHERED**
- ↳ Verbindung zu Computer und Jailbreak-Software ist nach einem Neustart zwingend erforderlich
- ↳ Das Gerät fährt sonst nicht mehr hoch

**SEMI-TETHERED**
- ↳ Das Gerät kann ohne Computer und Jailbreak-Software hochfahren
- ↳ Jaillbreak-Funktionen stehen nach Neustart nicht mehr zur Verfügung
- ↳ Erneutes Ausführen des Jailbreaks über den Computer notwendig

**SEMI-UNTETHERED**
- ↳ Das Gerät kann ohne Computer und Jailbreak-Software hochfahren
- ↳ Jailbreak-Funktionen stehen nach Neustart nicht mehr zur Verfügung
- ↳ Über eine installierte Jailbreak-App (z.B. checkrain) kann der Jailbreak ohne Computer durchgeführt werden

**UNTETHERED**
- ↳ Nach dem Jailbreak ist keine Verbindung zum Computer mehr nötig
- ↳ Der Jailbreak bleibt auch nach Neustart erhalten

# Durchführung des Jailbreaks

Zur Durchführung eines Jailbreaks stehen bereits verschiedene Anleitungen zur Verfügung. Das vorliegende Dokument beschreibt anhand von checkraln das genaue Vorgehen. checkrain ist ein frei verfügbarer Jailbreak für iOs 14. Folgende Schritte müssen dafür durchgeführt werden:

1. Download der neusten Version von checkraln über diesen Link https://checkra.in/ und anschließende Installation der Anwendung (Anmerkung: Alternativ kann checkraln auch via Homebrew installiert werden).

2. Der Start der Anwendung muss über einen Rechtsklick im Programmordner im Finder durchgeführt werden.

3. Jetzt muss das iOS-Gerät, auf dem der Jailbreak durchgeführt werden soll, via USB mit dem Mac verbunden werden. Mit einem Klick auf „Start“ wird der Jailbreak gestartet (Anmerkung: sollte an dieser Stelle eine Inkompatibilitätswarnung erscheinen und der „Start“ Button nicht auswählbar sein, so muss in den Geräteeinstellungen über „Options > Allow untested iOS/iPadOS/tvOS versions“ das Jailbreaking ungetesteter iOS-Versionen zugelassen werden. Die Kompatibilität des jeweiligen iPhones kann auf der folgenden Seite nachgeschlagen werden https://www.theiphonewiki.com/wiki/Jailbreak/14.x).

4. Als Nächstes muss das iOS-Gerät in den Wiederherstellungsmodus (DFU-Modus) versetzt werden, die dazu zu befolgenden Schritte werden vom Installationsprogramm angegeben.

5. checkraln beginnt nun mit der Durchführung des Jailbreaks auf dem iOS-Gerät. Es kann passieren, dass das Programm während der Installation zu bestimmten Aktionen, wie das Kabel Ab- und wieder Anstecken, auffordert.

6. Nach erfolgreicher Durchführung sollte die checkraln Applikation auf dem iOS-Gerät zur Verfügung stehen.

7. Über die checkraln App kann der Cydia App Store installiert werden. Mit diesem lassen sich die verschiedensten Tools zur Durchführung sicherheitsrelevanter Analysen installieren (Anmerkung: sollte eine benötigte Anwendung nicht im Store vorhanden sein, könnte ein manuelles Hinzufügen eines Repositorys notwendig sein. Dies wird im nächsten Kapitel behandelt).

8. Der Jailbreak über checkraln ist semi-tethered. Das bedeutet, dass der Jailbreak erneut angewendet werden muss, wenn das iOS-Gerät neu gestartet wurde, damit Cydia und die über Cydia installierten Programme wieder genutzt werden können.

# Einrichtung des iOS Geräts

Nach Durchführung des Jailbreaks steht auf dem iOS-Gerät der Cydia App Store zur Verfügung. Um einen Penetrationstest durchzuführen, müssen weitere Programme, sogenannte Tweaks, installiert werden. Einige Tweaks sind nicht über die bereits vorhandenen Standard-Repositories verfügbar, weswegen die für diesen Guide benötigten Repositories noch hinzugefügt werden müssen. Ein Repository kann als Sammlung von Applikationen eines Entwicklers verstanden werden.

Folgender Link führt zu einem Video, in welchem das Hinzufügen eines Repositories im Cydia Appstore gezeigt wird: Hinzufügen eines Repositories im Cydia App Store. Abbildung 1 zeigt das Hinzufügen eines Repositories über eine URL.

**Folgende Repositories werden benötigt:**

- https://ryleyangus.com/repo
- https://cydia.radare.org
- https://build.frida.re
- https://julioverne.github.io
- https://repo.co.kr
- https://apt.binger.com
- https://repo.hackyouriphone.org

Anmerkung: Beim Hinzufügen dieses Repo sitories erscheint eine Quellenwarnung, bei der “Trotzdem hinzufügen” ausgewählt wer den muss.

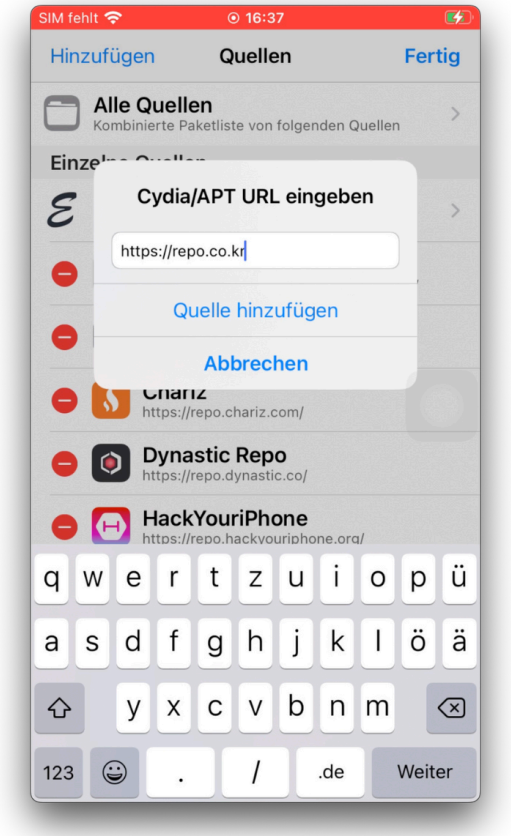



Abb. 1: Hinzufügen eines Repositories im Cydia App Store

Nach dem Hinzufügen der aufgelisteten Repositories sollten die benötigten Tweaks installiert werden. Unter folgendem Link findet sich ein Video, welches das Installieren einer App im Cydia App Store zeigt: Installieren eines Programms über den Cydia Store. Nach der Installation einiger Programme ist es notwendig, die Benutzeroberfläche des iOS-Gerätes (SpringBoard) neu zu starten. Cydia bietet dies über die App mit dem Button “SpringBoard neu starten” automatisch an. Nach einem Neustart des SpringBoards ist kein erneuter Jailbreak notwendig. Alternativ kann die Benutzerüberfläche auch über das Terminal neu gestartet werden. Dies wird im Abschnitt SpringBoard Neustarten beschrieben.

**Die zu installierenden Apps sind:**

- A-Bypass: Jailbreak Detection Bypass
- AppSync Unified: Ermöglicht die Installation von beliebigen IPAs
- BigBoss Recommended Tools: Hilfreiche Tools wie top, whois, curl etc.
- Filza File Manager: File Manager App auf dem iOS-Gerät
- Frida: Frida Server
- MTerminal: Terminal App auf dem iOS-Gerät (v1.4-6), siehe Abbildung 2.
- OpenSSH: Ermöglicht SSH-Zugriff auf das Gerät

**Nutzername:** mobile/root
**Passwort**: alpine

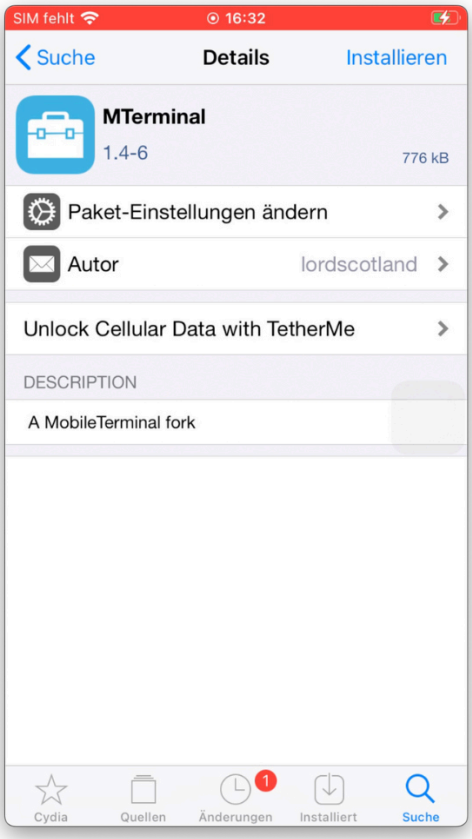



Abb. 2: Installation einer App über Cydia

**Nach der Installation aller Programme sollte das SpringBoard des iOS-Geräts neu gestartet werden, um Problemen mit dem Programm** AppSync Unified **vorzubeugen.**

# Konfiguration von Burp Suite als Proxy

Zum Analysieren des Netzwerkverkehrs des iOS-Geräts kann das HTTP-Intercepting Tool Burp Suite benutzt werden. Unter Burp Suite ist es über den Reiter Proxy möglich, einen neuen Proxyserver zu konfigurieren, der auf allen Netzwerkschnittstellen lauscht. Auf dem iOS-Gerät muss die IP-Adresse und der Port des Proxys in den WLAN-Einstellungen eingetragen werden. Diese Eintragung wird in Abbildung 3 gezeigt. Wenn beide Geräte im selben Netzwerk sind wird nun sämtlicher Netzwerkverkehr über den Proxy umgeleitet. Folgender Link führt zu einem Video, welches die Einrichtung eines Proxies zeigt: Einrichtung eines Proxies unter iOS. Das Tool kann nun jegliche HTTP-Kommunikation mitlesen und verändern.

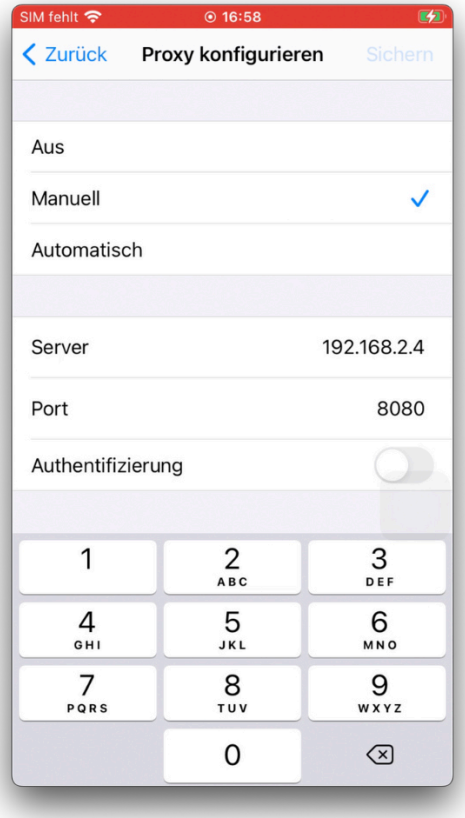



Abb. 3: Kofiguration eines Proxy-Servers in iOS.

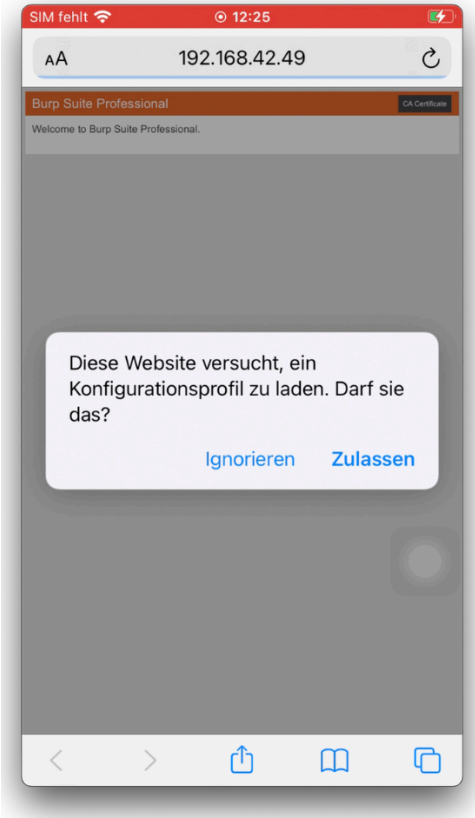



Abb. 4 Burp- Zertifikat über den Browser herunterladen.

Um auch mit TLS abgesicherte Kommunikation über den Proxy mitschneiden zu können, muss das Burp Suite Root Zertifikat auf dem Gerät installiert werden. Dieses kann über den Broswer heruntergeladen werden, wenn das iPhone im selben Netzwerk wie der Proxy ist. Dazu muss die Seite des Burp Suite Proxyservers aufgerufen werden. Die URL setzt sich aus IP-Adresse und Port zusammen und sollte identisch mit dem in den WLAN-Einstellungen eingetragenen Proxyserver sein. Oben rechts im Fenster kann das Burp Suite CA Root Zertifikat heruntergeladen werden. Abbildung 4 zeigt diesen Vorgang. In den Einstellungen des Geräts erscheint daraufhin ein neues Profil. Das Zertifikat wird unter "Allgemein > Info > Zertifikatsvertrauenseinstellungen" aktiviert. Folgendes Video zeigt das Vorgehen: Burp Suite CA Zertifikat installieren.

Damit ist es jetzt möglich, auch mit TLS abgesicherte Kommunikation mitzuschneiden. Einige Apps implementieren jedoch eine Schutzmaßnahme gegen dieses Vorgehen. Im Kapitel Umgehen des SSL Pinnings wird die Maßnahme und auch deren Umgehung erläutert.

# Umgehen der Jailbreak-Detection

**Springboard neu starten**
Bei Neu- und Deinstallation von Programmen, aber auch bei bestimmten Einstellungen, kann es teilweise zu Problemen kommen. Beispielsweise funktioniert eine neu installierte App nicht oder eine Einstellung wurde nicht übernommen. In solchen Fällen lohnt sich ein Neustart des SpringBoards. Dies kann mit folgendem Befehl im Terminal des iOS-Geräts oder per SSH getan werden:

```
$ killall -HUP SpringBoard
```

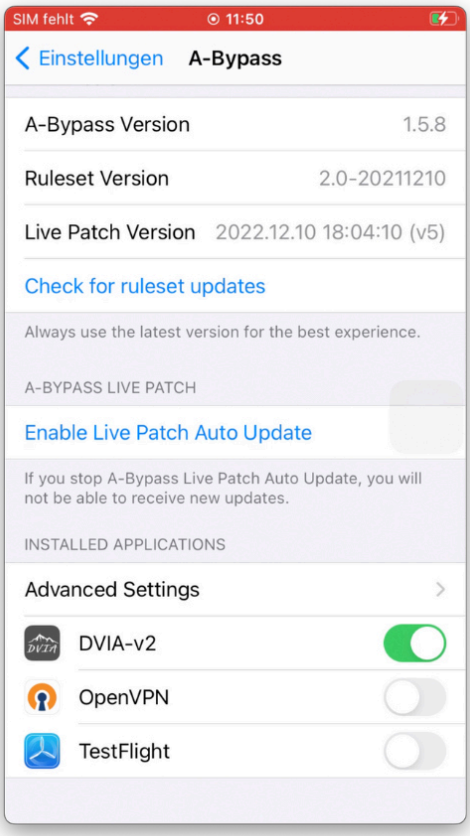



Abb. 5: Der Jailbreak-Detection Bypass kann für jede App separat eingestellt werden

Jailbreak-Detection ist eine Schutzmaßnahme, die dazu führt, dass Apps nicht auf Geräten starten oder funktional eingeschränkt sind, wenn ein Jailbreak erkannt wird. Entwickler implementieren solche Maßnahmen, um Analysen der Applikationen zu verhindern und externe Bedrohungen zu minimieren. Dafür prüft eine mit diesem Schutz ausgestattete App bestimmte Merkmale im Betriebssystem, die auf einen Jailbreak hindeuten. Wird ein Merkmal erkannt, so kann die App sich davor schützen. Es gibt mehrere Programme und Methoden, um diesen Schutz zu umgehen, jedoch gibt es kein Programm, welches für alle iOS-Geräte, Versionen und Apps funktioniert. Im Folgenden wird der Tweak A-Bypass vorgestellt, welcher für eine Vielzahl von Jailbreak-Detection Methoden geeignet ist.

m Kapitel Einrichtung des iOS-Geräts wurde bereits erläutert, dass die App über Cydia installiert werden sollte. A-Bypass sollte daraufhin unten in den Einstellungen erscheinen. Falls nicht, könnte ein Neustart des SpringBoards Abhilfe schaffen. Das A-Bypass Menü in den Einstellungen listet daraufhin die installierten Apps auf. Mit dem angezeigten Schieberegler kann der Jailbreak Detection Bypass für die zu untersuchende App aktiviert werden. Dies wird in Abbildung 5 gezeigt. Beim Starten der App wird daraufhin A-Bypass automatisch versuchen, die Jailbreak Erkennung zu umgehen. Folgendes Video zeigt das beschriebene Vorgehen: Jailbreak-Detection Bypass mit A-Bypass.

A-Bypass sendet die Bundle-ID der gestarteten App an einen südkoreanischen Server. Dies sollte blockiert werden.

# Umgehen des SSL Pinning

Beim sogenannten SSL Pinning handelt es sich um eine Sicherheitsmaßnahme, die die Kommunikation der Anwendung sicher gegen Man-In-The-Middle Angriffe machen soll. Hierzu gibt der Entwickler der Anwendung eine Liste von gültigen Server-Zertifikaten vor, welche dann zur Laufzeit herangezogen wird, um die Identität des Servers zu validieren. Wird mit einem anderen Server (wie zum Beispiel der Burp Suite Proxy) über eine sichere Verbindung kommuniziert und findet sich das Zertifikat nicht in der Liste der erlaubten Zertifikate, so wird die App keine Verbindung zu dem Server aufbauen. Das manuelle Hinzufügen von Root-Zertifikaten hat dabei keine Auswirkungen. Bei der Umgehung dieser Sicherheitsmaßnahme kann der Penetrationstester auf verschiedene Arten vorgehen. Der grundlegendste Ansatz wäre die Editierung des Quellcodes. Da dieser jedoch nur in seltenen Fällen zur Verfügung steht, muss auf alternative Werkzeuge zurückgegriffen werden, um den Datenverkehr der Anwendung zu analysieren. Die einfachste Möglichkeit ist die Anwendung von SSL Kill Switch 2, was die Validierung von TLS-Zertifikaten aushebelt.

TLS ist danach Systemdeaktiviert

Vor der Installation auf iOS 14, ist es notwendig, dass das Programm Substitute installiert wird.

Ohne Substitute kann nach dem Neustarten des SpringBoards Cydia nicht mehr verwendet werden und SSL Kill Switch 2 muss manuell deinstalliert werden. Zusätzlich sollte überprüft werden, dass die Programme Debian Packager, Cydia Substrate und PreferenceLoader installiert sind. Diese werden im Normalfall automatisch von checkra1n installiert. Nach dem Hinzufügen aller Repositories werden zwei SSL Kill Switch 2 Programme in der Cydia Suche angezeigt. Das zu installierende Programm hat ein gelbes Logo und unten in der Übersicht des Programms wird als Repository julioverne.github.io angezeigt. Nach der Installation und einem Neustart des SpringBoards, erscheint in den Einstellungen des Geräts unten ein neuer Eintrag mit dem Namen SSL Kill Switch 2. Hier kann die Überprüfung der Zertifikate wieder aktiviert werden und bestimmte Apps von Kill Switch ausgeschlossen werden. Dafür muss die sogenannte Bundle-ID einer App bekannt sein, welche eine App eindeutig identifiziert. Dieser Wert kann mit dem Tool Grapefruit oder Frida extrahiert werden. Das konkrete Vorgehen wird im nächsten Kapitel erläutert. Die Einrichtung von Grapefruit wird im Kapitel Konfiguration von Grapefruit behandelt.

Um SSL Kill Switch 2 manuell zu entfernen, kann folgender Terminal Befehl in der App Terminal oder per SSH benutzt werden:

```
$ apt-get --purge remove "com.julioverne.sslkillswitch2"
```

Das SpringBoard muss daraufhin neu gestartet werden.

# Konfiguration von Frida

Mithilfe des Frida Toolkits lassen sich verschiedene Aufgaben im Bereich des Pentesting und Reverse-Engineering umsetzen. Die Inbetriebnahme einer Frida Instanz erfordert einen Client, dessen Installation auf dem Mac mit folgendem Befehl durchgeführt werden kann:

```
$ pip3 install frida-tools
```

Für diesen Befehl muss Python auf dem System installiert sein. Die umfassende Installationseinleitung und Dokumentation kann unter https://frida.re/docs/installation gefunden werden. Nach der Durchführung der Schritte unter Einrichtung des iOS-Geräts sollte der Frida Server bereits auf dem iOS-Gerät installiert sein. Folgende Befehle stehen mittels frida-ps bereit:

Auflistung der angeschlossenen Geräte, auf dem der Frida Server läuft:

```
$ frida-ls-devices
```

Alle installierten Apps auflisten:

```
$ frida-ps -Uia
```

Laufende Prozesse auflisten:

```
$ frida-ps -U
```

Laufende Apps auflisten:

```
$ frida-ps -Ua
```

Über diese Befehle ist es möglich, sich die Bundle-ID einer zu untersuchenden App anzeigen zu lassen. Eine vollständige Dokumentation kann hier gefunden werden:
https://frida.re/docs/frida-ps/.

# Konfiguration von Grapefruit

Bei Grapefruit handelt es sich um eine GUI Anwendung für den Mac, welche den Frida Server eines iOS-Gerätes nutzt, um Einsicht in die zu untersuchende App zu bekommen. Die Installation erfolgt auf dem Mac mit folgendem Befehl:

```
$ npm install -g igf
```

Nachdem die Anwendung mit dem Befehl $ igf gestartet wurde, ist sie im Browser über die URL http://localhost:31337 verfügbar. Das Gerät muss nun per Kabel oder per SSH verbunden werden und der Frida-Server muss gestartet sein. Durch Auswählen der zu untersuchenden App in der Übersicht von Grapefruit kann diese App gestartet werden. Unter dem Reiter Basic Information werden unter anderem die Bundle-ID der App und der Inhalt der Info.plist angezeigt. In der Leiste links können die verschiedenen Funktionen von Grapefruit benutzt werden. Mit diesem Tool sind eine Vielzahl von Operationen möglich, einige davon sind:

- Untersuchen der von der App gespeicherten Dateien
- Einsehen der registrierten URL Schemes
- Auslesen der gespeicherten Cookies
- Simulation von GPS Daten

# Unsigned/Untrusted IPA installieren

Es kann vorkommen, dass eine App auf dem iOS-Gerät installiert werden muss, welche nicht über den gewöhnlichen Apple App Store zur Verfügung steht und zusätzlich nicht ausreichend signiert ist. Als Beispiel kann die Damn Vulnerable iOS App(DVIA) installiert werden, welche sich zum Testen der aufgeführten Pentesting Tools eignet. Dafür muss AppSync Unified auf dem iOS-Gerät installiert sein, was nach der Durchführung der Schritte Einrichtung des iOS-Geräts der Fall sein sollte. Außerdem muss die Installationsdatei der App im IPA Format auf dem Mac heruntergeladen werden. Nachfolgend werden zwei Möglichkeiten beschrieben, um die App auf dem iOS-Gerät zu installieren:

1. **via ideviceinstaller:**
   Zuerst muss libimobiledevice und ideviceinstaller via Homebrew auf dem Mac installiert werden. Dafür wird folgender Befehl im Terminal eingegeben:

```
$ brew install libimobiledevice
```

```
$ brew install ideviceinstaller
```

Als Nächstes kann die App mit folgendem Befehl installiert werden:

```
$ ideviceinstaller -i DVIA-v2-swift.ipa
```

DVIA-v2-swift.ipa ist dabei der Pfad zu der heruntergeladenen App Datei.

2. **Alternativ kann die App via Xcode installiert werden:**
   - Xcode starten und beliebiges Projekt anlegen oder öffnen.
   - In der oberen Leiste wird über Window > Device and Simulators ausgewählt. Unter Installed Apps beim entsprechenden Gerät wird die IPA ausgewählt. Daraufhin startet die Installation auf dem iOS-Gerät.

# Weitere Jailbreaks

Wie eingangs erwähnt, war der Pentesting-Leitfaden eine Beschreibung für den iOS 14 Jailbreak checkra1n. Für andere iOS Versionen stehen andere Jailbreaks zur Verfügung. So zum Beispiel palera1n für iOS 15.0 bis iOS 16.3, wobei nicht alle Versionen unterstützt werden. Die genauen Daten finden sich in der offiziellen Dokumentation. Der Code und die Dokumentation von palera1n kann unter https://palera.in gefunden werden. In diesem Kapitel werden kurz die Unterschiede zwischen palera1n und checkra1n beleuchtet.
Die Installation von palera1n wird auf macOS wie folgt durchgeführt:

- Herunterladen des Jailbreak-Pakets palera1n-macos-universal unter https://github.com/palera1n/palera1n/releases
- Öffnen des Terminals und Navigieren zum heruntergeladenen Ordner
- Folgende Befehle ausführen:

```
$ sudo mkdir /usr/local/bin
```

Falls dieser Ordner nicht existiert:

```
$ sudo mv ./palera1n-macos-universal /usr/local/bin/palera1n
```

palera1n-macos-universal sollte dabei mit der heruntergeladenen Version ersetzt werden

```
$ sudo xattr -c /usr/local/bin/palera1n
```

```
$ sudo chmod +x /usr/local/bin/palera1n
```

palera1n ist jetzt auf dem System installiert. Nun wird beschrieben, wie der Jailbreak auf dem Gerät durchgeführt wird. Folgende Schritte sind dafür zu unternehmen:

- Um Fehler zu vermeiden, wird empfohlen (falls möglich) das Gerät mit einem USB-A zu Lightning-Kabel, statt mit einem USB-C zu Lightning-Kabel zu verbinden
- Es sollten mindestens 2-3 GB freier Speicher zur Verfügung stehen
- Folgender Befehl muss ausgeführt werden: $ palera1n <parameter>, wobei für <parameter> folgendes gesetzt sein muss:
  - für Geräte mit 16 GB oder iOS 15 Geräte mit mehr als 2-3 GB freien Speicher aber unter 10-15 GB: -B -f
  - für iOS 16 Geräte oder Geräte mit mindestens 10-15 GB freien Speicher: -c -f

Bei checkra1n wurde der Cydia App Store installiert, unter palera1n wird der Silio App Store installiert. Die Handhabung der beiden App Stores zum Installieren von Apps und Hinzufügen von Repositories unterscheidet sich aber nicht weiter. Der Großteil der in diesem Guide vorgestellten Tweaks funktioniert auch unter diesem Jailbreak. Jedoch kann es auch – zum Beispiel bei Frida – zu Problemen kommen. Dies hängt von den verwendeten iPhone-Modellen und iOS-Versionen ab. Noch moderne Jailbreaks sind XinaA15 und Fugu15, die auch unter iOS 15 funktionieren, jedoch auch aktuellere iPhone-Modelle unterstützen. Beide sind semi-untethered. Eine gute Übersicht dazu kann hier gefunden werden: https://www.theiphonewiki.com/wiki/Jailbreak/15.x .

# Anhang - Tool Liste

**ibimobiledevice und ideviceinstaller**
- Verschiedene CLI Tools die direkt mit dem iOS-Gerät interagieren
- Installation:

```
$ brew install libimobiledevi
```

```
$ brew install ideviceinstaller
```

**objection**
- Nutzt Frida Server und ermöglicht auch SSL Pinning Bypass
- Runtime Mobile Exploration Toolkit
- Erleichtert verschiedene Aspekte der dynamischen Analyse
- Dokumentation

**frida-ios-dump**
- Nutzt Frida Server
- Ermöglicht Entschlüsseln und Auslesen (Dumpen) von IPAs, die auf dem iOS-Gerät installiert sind
- Dokumentation im Github Readme

**grapefruit**
- Nachfolger von passionfruit
- Nutzt Frida Server
- Web Oberfläche für verschiedene Funktionen wie Dumpen von App Dateien, Logs, etc.
- Dokumentation

**Burp Suite**
- Netzwerkverkehr von Apps mitlesen
- Vorher muss SSL Pinning umgangen werden
- Einrichtung am iOS-Gerät
- Zertifikat am iOS-Gerät installieren
- weitere Hinweise

**Terminal App**
- Terminal Emulator für iOS

**AppSync Unified**
- Ermöglicht die Installation beliebiger IPA App Dateien auf iOS-Geräten mit Jailbreak

**A-Bypass**
- Jailbreak Detection Bypass
- Auswahl des Bypass pro App in den Einstellungen möglich
- Einrichtung

# Über die AWARE7

Die AWARE7 GmbH ist ein Cyber Security Unternehmen aus Gelsenkirchen das Technologien und Produkte entwickelt sowie Dienstleistungen anbietet, die zur Förderung, Steigerung und Erhaltung des IT-Sicherheitslevels dienen. Durch die praktische Arbeit und die regelmäßige Veröffentlichung von wissenschaftlichen Artikeln gelingt es uns komplexe Betrugs- und Angriffsmethoden zu erklären und zu entdecken, um Unternehmen und Behörden zu schützen.

IT-Sicherheit funktioniert nur, wenn die Technik sicher und die Menschen sensibilisiert sind. Komplexe Angriffe nutzen menschliche Schwachstellen in Kombination mit technischen Sicher- heitslücken aus. Betrachten Sie das IT- Sicherheitsniveau in ihrem Unternehmen ganzheitlich auf menschlicher und technischer Ebene. Wir sind in Sachen Sicherheit ganzheitlich an Ihrer Seite.

AWARE7

## Kontakt

AWARE7 GmbH
Munscheidstraße 14
45886 Gelsenkirchen
info@aware7.de

Chris Wojzechowski
Geschäftsführer
+49 209 88306761
chris@aware7.de