

HỌC VIỆN KỸ THUẬT MẬT MÃ

**KHOA AN TOÀN THÔNG TIN**



BÁO CÁO BÀI TẬP LỚN

**AN TOÀN MẠNG KHÔNG DÂY**

**Đề tài:**

**TÌM HIỂU HACKING IOS**

***Nhóm sinh viên thực hiện:*** Vũ Tiến Đạt AT170609

Đặng Xuân Đức AT170612

Hoàng Hữu Ánh AT170604

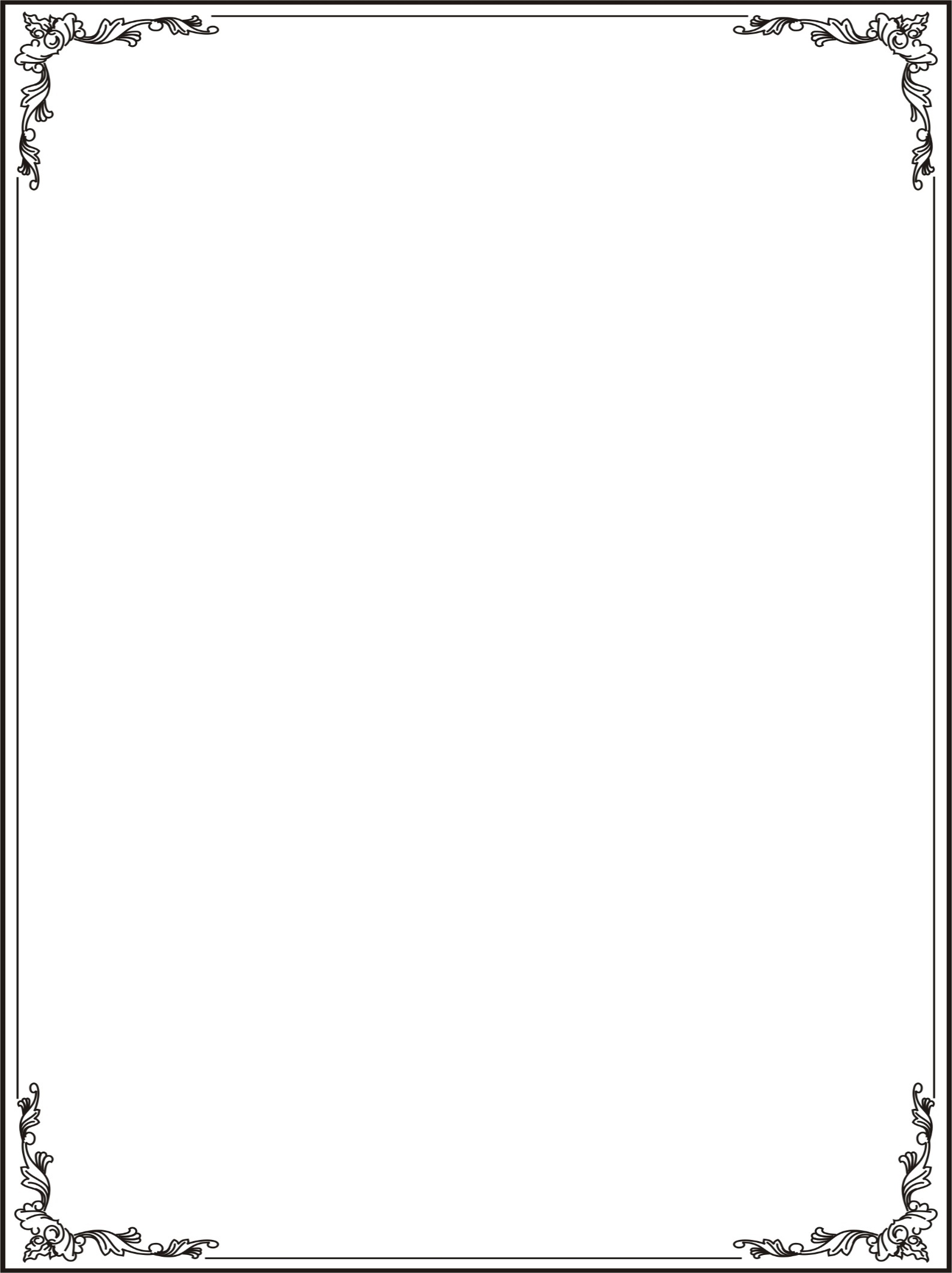
Tạ Quang Chiến AT170607

Phan Văn Hoàng AT170622

Nhóm 12

***Giảng viên hướng dẫn****:* TS. Hoàng Sỹ Tương

Hà Nội, 13-11-2023



HỌC VIỆN KỸ THUẬT MẬT MÃ

**KHOA AN TOÀN THÔNG TIN**



BÁO CÁO BÀI TẬP LỚN

**AN TOÀN MẠNG KHÔNG DÂY**

**Đề tài:**

**TÌM HIỂU HACKING IOS**

***Nhóm sinh viên thực hiện:*** Vũ Tiến Đạt AT170609

Đặng Xuân Đức AT170612

Hoàng Hữu Ánh AT170604

Tạ Quang Chiến AT170607

Phan Văn Hoàng AT170622

Nhóm 12

***Giảng viên hướng dẫn****:* TS. Hoàng Sỹ Tương

Hà Nội, 13-11-2023

# LỜI CẢM ƠN



Nhóm chúng em xin gửi lời cảm ơn chân thành tới các thầy giáo, cô giáo trong Khoa An toàn thông tin đã tạo điều kiện thuận lợi cho nhóm em trong quá trình thực hiện bài tập này.

Chúng em xin gửi lời cảm ơn đặc biệt đến Tiến sĩ thầy: Hoàng Sỹ Tương đã nhiệt tình hướng dẫn và chỉ bảo nhóm trong suốt thời gian thực hiện bài tập, giúp chúng em có thể hoàn thành tốt bài tập lớn lần này.

|  |  |
| --- | --- |
| **GIẢNG VIÊN HƯỚNG DẪN** | **NHÓM SINH VIÊN THỰC HIỆN** |
|  | Vũ Tiến Đạt |
|  | Đặng Xuân Đức |
|  | Hoàng Hữu Ánh |
|  | Phan Văn Hoàng |
| TS. Hoàng Sỹ Tương | Tạ Quang Chiến |

LỜI NÓI ĐẦU

Trong quá trình làm bài tập, cũng như là trong quá trình làm bài báo cáo, khó tránh khỏi sai sót, rất mong các thầy bỏ qua. Đồng thời do trình độ cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, chúng em rất mong nhận được góp ý của thầy để chúng em học thêm được nhiều kinh nghiệm và hoàn thành tốt hơn trong các bài báo cáo sắp tới.

Báo cáo được chia làm 3 chương, với các nội dung sau:

**Chương I: Cơ sở lý thuyết**

**Chương II: Giới thiệu về Kỹ thuật khai thác IOS**

**Chương III: Thực nghiệm**

DANH MỤC HÌNH VẼ

[Hình 1.1 Quá trình attack vector mà tin tặc có thể sử dụng 6](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736857)

[Hình 1.2 Giải phẫu của một cuộc tấn công trên thiết bị di dộng 8](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736858)

[Hình 3.1 Cài đặt checkaln 19](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736859)

[Hình 3.2 Giao diện checkaln 20](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736860)

[Hình 3.3 Thông báo trước khi vào recovery mode 20](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736861)

[Hình 3.4 Recovery mode 21](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736862)

[Hình 3.5 Đưa thiết bị vào chế độ DFU 21](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736863)

[Hình 3.6 Bắt đầu tải jailbreaking 22](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736864)

[Hình 3.7 22](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736865)

[Hình 3.8 Biểu tượng checkaln ở màn hình chính 23](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736866)

[Hình 3.9 Đọc file Plist 24](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736867)

[Hình 3.10 Giao diện DVIA-v2 24](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736868)

[Hình 3.11 Giao diện Local Data Storage 25](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736869)

[Hình 3.12 Giao diện Plist 25](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736870)

[Hình 3.13 Chạy objection 26](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736871)

[Hình 3.14 Kiểm tra file 26](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736872)

[Hình 3.15 Giao diện DVIA-v2 27](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736873)

[Hình 3.16 Giao diện Local Data Storage 27](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736874)

[Hình 3.17 Nhập nội dung 28](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736875)

[Hình 3.18 Chạy objection 28](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736876)

[Hình 3.19 Kiểm tra nsuserdefaults 29](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736877)

[Hình 3.20 Giao diện DVIA-v2 29](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736878)

[Hình 3.21 Giao diện Toucj/Face ID Bypass 30](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736879)

[Hình 3.22 Kích hoạt thông báo 30](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736880)

[Hình 3.23 Chạy objection 31](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736881)

[Hình 3.24 Kích hoạt khai thác 31](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736882)

[Hình 3.25 Thử lại vân tay 31](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736883)

[Hình 3.26 Ghi lại thông tin sau khi khai thác thành công 32](file:///D:\Nam4\Antoanmangkhongday\decuong.docx#_Toc154736884)

**MỤC LỤC**

[LỜI CẢM ƠN iii](#_Toc154736923)

[LỜI NÓI ĐẦU iv](#_Toc154736924)

[DANH MỤC HÌNH VẼ v](#_Toc154736925)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 3](#_Toc154736926)

[1.1 Tổng quan về Apple IOS 3](#_Toc154736927)

[1.1.1 Khái niệm 3](#_Toc154736928)

[1.1.2 Cấu trúc 3](#_Toc154736929)

[1.1.3 Vai trò 4](#_Toc154736930)

[1.2 Tổng quan về Hacking MOBILE 5](#_Toc154736931)

[1.2.1 Mobile Platform attack vector 5](#_Toc154736932)

[1.2.2 Owasp top 10 mobile risk 7](#_Toc154736933)

[1.2.3 Cấu trúc cuộc tấn công thiết bị di động 8](#_Toc154736934)

[CHƯƠNG 2. GIỚI THIỆU VỀ KỸ THUẬT KHAI THÁC IOS 10](#_Toc154736935)

[2.1 IOS Jailbreaking 10](#_Toc154736936)

[2.1.1 Khái niệm 10](#_Toc154736937)

[2.1.2 Phân loại 10](#_Toc154736938)

[2.1.3 Kỹ thuật Jailbreaking 11](#_Toc154736939)

[2.2 Kỹ thuật tấn công thiết bị IOS 11](#_Toc154736940)

[2.2.1 Kỹ thuật SSL pinning bypass 11](#_Toc154736941)

[2.2.2 Insecure Local Data Storage 12](#_Toc154736942)

[2.2.3 Fingerprint Bypass 14](#_Toc154736943)

[2.3 Tool pentesting objection 14](#_Toc154736944)

[2.3.1 Khái Niệm: 14](#_Toc154736945)

[2.3.2 Cấu trúc: 15](#_Toc154736946)

[2.3.3 Cách hoạt động: 15](#_Toc154736947)

[2.3.4 Tác dụng: 16](#_Toc154736948)

[2.3.5 Cách cài đặt. 18](#_Toc154736949)

[2.3.6 Yêu Cầu Tiên Quyết: 18](#_Toc154736950)

[2.3.7 Cài Đặt Objection: 18](#_Toc154736951)

[2.3.8 Khởi Chạy Objection: 18](#_Toc154736952)

[CHƯƠNG 3. THỰC NGHIỆM 19](#_Toc154736953)

[3.1 Chuẩn bị 19](#_Toc154736954)

[3.2 Kịch bản IOS Jailbreaking 20](#_Toc154736955)

[3.3 Kịch bản Hacking IOS 23](#_Toc154736956)

[3.3.1 Insecure Local data Storage 23](#_Toc154736957)

[3.3.2 Fingerprint Bypass 29](#_Toc154736958)

[KẾT LUẬN 33](#_Toc154736959)

[TÀI LIỆU THAM KHẢO 34](#_Toc154736960)

# CƠ SỞ LÝ THUYẾT

## Tổng quan về Apple IOS

### Khái niệm

iOS là hệ điều hành di động của Apple cho các thiết bị như iPhone, iPad và iPod Touch. Với giao diện người dùng đơn giản, App Store đa dạng, và tính bảo mật cao, iOS tập trung vào trải nghiệm người dùng mượt mà. Các dịch vụ như iTunes và iCloud giúp quản lý dữ liệu và đồng bộ thông tin. An toàn và bảo mật được đảm bảo thông qua Touch ID và Face ID. Siri là trợ lý ảo tích hợp, và người phát triển sử dụng Swift hoặc Objective-C để phát triển ứng dụng. iOS là một phần của hệ sinh thái rộng lớn của Apple, mang lại trải nghiệm đồng bộ trên nhiều thiết bị của họ.

cấu trúc

### Cấu trúc

Cấu trúc của iOS bao gồm nhiều thành phần quan trọng, từ phần mềm đến phần cứng. Dưới đây là một cái nhìn tổng quan về cấu trúc của hệ điều hành iOS:

Kernel (Nhân): Đây là phần cốt lõi của hệ điều hành, quản lý tài nguyên hệ thống và cung cấp giao tiếp giữa phần cứng và phần mềm. iOS sử dụng một phiên bản cá nhân hóa của Mach kernel.

Quản lý Tài Nguyên: Các thành phần quản lý tài nguyên bao gồm các thành phần như trình quản lý bộ nhớ, trình quản lý nguồn điện, và trình quản lý thiết bị đầu cuối để quản lý tài nguyên phần cứng hiệu quả.

Core OS (Hệ thống Cơ Bản): Là tập hợp các framework cơ bản cung cấp các dịch vụ hệ thống như quản lý tệp tin, mạng, bảo mật và các dịch vụ cơ bản khác.

Core Services (Dịch Vụ Cơ Bản): Bao gồm các dịch vụ cung cấp các tính năng như đồng bộ hóa dữ liệu, quản lý cấu hình, và các dịch vụ về vị trí.

Media Layer (Lớp Phương Tiện): Bao gồm Core Audio, Core Animation, Core Graphics, và các framework khác liên quan đến xử lý đa phương tiện và đồ họa.

Cocoa Touch: Là một tập hợp các framework được xây dựng đặc biệt cho giao diện người dùng đa chạm. Bao gồm UIKit (đối với giao diện người dùng), MapKit (đối với bản đồ), và các framework khác hỗ trợ phát triển ứng dụng di động.

App Frameworks (Framework Ứng Dụng): Bao gồm các framework cho việc phát triển ứng dụng như SwiftUI, UIKit, và các framework đặc biệt khác như ARKit cho thực tế ảo.

Xcode: Đây là môi trường phát triển tích hợp (IDE) mà các nhà phát triển sử dụng để xây dựng ứng dụng iOS.

App Store Connect: Là nơi nhà phát triển đưa ứng dụng của họ lên App Store để phân phối cho người dùng.

Swift và Objective-C: Là hai ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng iOS. Swift là ngôn ngữ hiện đại được Apple phát triển, trong khi Objective-C là ngôn ngữ truyền thống đã được sử dụng từ trước đến nay.

Hệ Thống Tệp Tin: iOS sử dụng hệ thống tệp tin APFS (Apple File System) để quản lý lưu trữ và các tệp tin trên thiết bị.

Hệ Sinh Thái Apple: iOS là một phần của hệ sinh thái rộng lớn của Apple, liên kết chặt chẽ với macOS, watchOS, tvOS, và các dịch vụ như iCloud, iTunes, và iMessage.

### Vai trò

Hệ điều hành iOS của Apple đóng vai trò quan trọng trong việc quản lý và điều khiển các thiết bị di động của họ, mang lại một loạt các tính năng và trải nghiệm cho người dùng. Dưới đây là một số vai trò chính của iOS:

Giao Diện Người Dùng Mượt Mà: iOS cung cấp một giao diện người dùng đơn giản, trực quan và mượt mà. Thiết kế tối giản và cảm ứng đa chạm tạo ra một trải nghiệm người dùng thân thiện và dễ sử dụng.

Quản Lý Ứng Dụng: iOS cho phép người dùng tải xuống và cài đặt ứng dụng từ App Store, nơi có hàng triệu ứng dụng khác nhau. Hệ điều hành này quản lý vị trí và tương tác giữa các ứng dụng, đồng thời cung cấp khả năng đa nhiệm để chạy nhiều ứng dụng cùng một lúc.

Bảo Mật và Quyền Riêng Tư: iOS được thiết kế với một lớp bảo mật cao để bảo vệ dữ liệu người dùng. Touch ID và Face ID cung cấp các phương tiện xác minh độc đáo, và Apple chú trọng đến quyền riêng tư của người dùng trong việc xử lý dữ liệu.

Cập Nhật Đều Đặn: Apple thường xuyên phát hành các bản cập nhật cho iOS để cải thiện tính năng, sửa lỗi và bảo mật. Người dùng có thể cập nhật hệ điều hành của mình để tận hưởng những cải tiến mới nhất.

Dịch Vụ Đám Mây iCloud: iOS tích hợp sâu với dịch vụ đám mây iCloud, giúp người dùng đồng bộ hóa và lưu trữ dữ liệu của họ trực tuyến, bao gồm hình ảnh, video, tệp tin và thông tin liên lạc.

Trợ Lý Ảo Siri: Siri là một trợ lý ảo tích hợp vào iOS, giúp người dùng thực hiện các tác vụ bằng giọng nói, như tìm kiếm thông tin, đặt lịch, và điều khiển các thiết bị khác.

Hỗ Trợ Đa Phương Tiện và Giải Trí: iOS có các tính năng đa phương tiện mạnh mẽ, bao gồm quản lý âm nhạc, video, và sách điện tử thông qua các ứng dụng như Apple Music, Apple TV và iBooks.

Phát Triển Ứng Dụng: iOS là nền tảng cho phát triển ứng dụng di động. Người phát triển có thể sử dụng Xcode và ngôn ngữ lập trình Swift hoặc Objective-C để tạo ra các ứng dụng cho App Store.

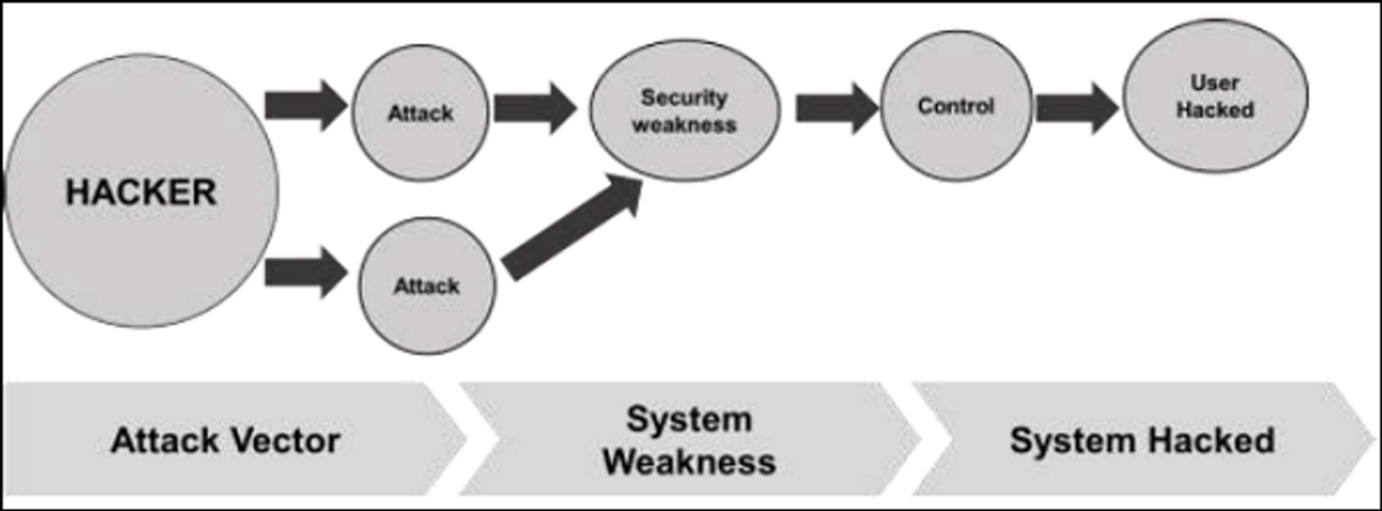
Hệ Sinh Thái Đồng Bộ: iOS là một phần của hệ sinh thái Apple, kết nối chặt chẽ với các hệ điều hành khác như macOS, watchOS và tvOS, tạo ra một trải nghiệm đồng bộ trên nhiều thiết bị Apple.

## Tổng quan về Hacking MOBILE

### Mobile Platform attack vector

a) Định nghĩa

Attack Vector là một phương pháp hoặc kỹ thuật mà hacker sử dụng để giành quyền truy cập vào một thiết bị máy tính hoặc mạng khác nhằm tiêm “bad code” thường được gọi là payload. Vector này giúp tin tặc khai thác các lỗ hổng hệ thống.Nhiều attack vector này lợi dụng yếu tố con người vì đây là điểm yếu nhất của hệ thống này.



Hình . Quá trình attack vector mà tin tặc có thể sử dụng

b) Một số attack vector di dộng

Phần mềm độc hại

* Virus & Rootkit
* Sửa đổi ứng dụng
* Sửa đổi hệ điều hành

Lọc dữ liệu

* Dữ liệu rời khỏi tổ chức
* In màn hình
* Sao chép vào USB và sao lưu mất

Giả mạo dữ liệu

* Sửa đổi bởi ứng dụng khác
* Những nỗ lực giả mạo không bị phát hiện

Mất dữ liệu

* Mất thiết bị
* Truy cập thiết bị trái phép
* Lỗ hổng ứng dụng

c) Hậu quả của các attack vector

Các attack vector là quá trình hack như đã được giải thích và nó đã thành công, sau đây là tác động lên thiết bị di động của bạn:

* Mất dữ liệu của bạn: Nếu thiết bị di động của bạn bị hack hoặc bị nhiễm vi-rút thì tất cả dữ liệu được lưu trữ của bạn sẽ bị kẻ tấn công mất và lấy đi.
* Sử dụng sai tài nguyên di động của bạn: Điều đó có nghĩa là mạng hoặc thiết bị di động của bạn có thể bị quá tải nên bạn không thể truy cập các dịch vụ chính hãng của mình. Trong trường hợp xấu hơn, hacker có thể sử dụng để gắn vào máy hoặc mạng khác.
* Mất danh tiếng: Trong trường hợp tài khoản Facebook hoặc tài khoản email doanh nghiệp của bạn bị hack, hacker có thể gửi tin nhắn giả mạo cho bạn bè, đối tác kinh doanh và những người liên hệ khác của bạn. Điều này có thể làm tổn hại đến danh tiếng của bạn.
* Trộm danh tính: Có thể xảy ra trường hợp trộm danh tính như ảnh, tên, địa chỉ, thẻ tín dụng, v.v. và những thông tin tương tự có thể được sử dụng để phạm tội.

### Owasp top 10 mobile risk

**Malware and Ransomware:** Phần mềm độc hại nhắm mục tiêu vào thiết bị di động có thể dẫn đến mất dữ liệu, đánh cắp tài chính hoặc đòi tiền chuộc.

**Phishing Attacks:** Tấn công kỹ thuật xã hội thông qua tin nhắn hoặc ứng dụng giả mạo lừa người dùng cung cấp thông tin nhạy cảm.

**Insecure Wi-Fi Networks:** Việc sử dụng mạng Wi-Fi không bảo mật có thể khiến thiết bị di động gặp nhiều cuộc tấn công khác nhau, chẳng hạn như các cuộc tấn công trung gian.

**Outdated Software:** Việc không cập nhật hệ điều hành và ứng dụng có thể khiến các lỗ hổng chưa được vá, khiến thiết bị dễ bị khai thác.

**Lost or Stolen Devices:** Việc truy cập trái phép vào thiết bị bị mất hoặc bị đánh cắp có thể dẫn đến vi phạm dữ liệu, đặc biệt nếu thiết bị thiếu các biện pháp bảo mật thích hợp như mật khẩu hoặc sinh trắc học.

**Unsecured App Permissions:** Các ứng dụng yêu cầu quyền quá mức hoặc truy cập dữ liệu nhạy cảm mà không có sự cho phép thích hợp có thể xâm phạm quyền riêng tư của người dùng.

**Jailbreaking/Rooting:** Việc sửa đổi hệ điều hành của thiết bị để bỏ qua các tính năng bảo mật tích hợp có thể khiến thiết bị gặp thêm rủi ro.

**Bluetooth and NFC Vulnerabilities:** Kết nối Bluetooth và NFC không an toàn có thể bị khai thác để truy cập trái phép hoặc chặn dữ liệu.

**Cryptojacking:** Các tập lệnh hoặc ứng dụng độc hại sử dụng tài nguyên của thiết bị để khai thác tiền điện tử mà người dùng không hề hay biết.

**IoT Integration Risks:** Khi ngày càng nhiều thiết bị được kết nối với nhau, các lỗ hổng trong thiết bị Internet of Things (IoT) có thể gây rủi ro cho thiết bị di động nếu chúng là một phần của cùng một mạng.

### A diagram of a step by step installation Description automatically generatedCấu trúc cuộc tấn công thiết bị di động

Hình . Giải phẫu của một cuộc tấn công trên thiết bị di dộng

a) Nhiễm trùng thiết bị

Việc lây nhiễm phần mềm gián điệp di động vào thiết bị được thực hiện khác nhau đối với thiết bị Android và iOS.

Android: Người dùng bị lừa tải xuống ứng dụng từ thị trường hoặc từ ứng dụng của bên thứ ba nói chung bằng cách sử dụng tấn công kỹ thuật xã hội. Việc lây nhiễm từ xa cũng có thể được thực hiện thông qua cuộc tấn công

Man-in-the-Middle (MitM), trong đó kẻ thù đang hoạt động chặn liên lạc di động của người dùng để tiêm phần mềm độc hại.

IOS: Việc lây nhiễm iOS yêu cầu quyền truy cập vật lý vào thiết bị di động. Việc lây nhiễm vào thiết bị cũng có thể thông qua việc khai thác lỗ hổng zero-day chẳng hạn như khai thác JailbreakME.

b) Cài đặt backdoor

Để cài đặt cửa sau cần có đặc quyền của quản trị viên bằng cách root thiết bị Android và bẻ khóa thiết bị Apple. Mặc dù các nhà sản xuất thiết bị đã đặt cơ chế phát hiện root/jailbreak, nhưng phần mềm gián điệp di động vẫn dễ dàng vượt qua chúng.

* Android: Cơ chế phát hiện root không áp dụng cho việc root có chủ ý.
* IOS: “Cộng đồng” bẻ khóa rất ồn ào và có động lực.

c) Bỏ qua các cơ chế mã hóa và lấy thông tin

Phần mềm gián điệp gửi nội dung di động như email và tin nhắn được mã hóa đến máy chủ của kẻ tấn công ở dạng văn bản thuần túy. Phần mềm gián điệp không tấn công trực tiếp vào vùng chứa an toàn. Nó lấy dữ liệu tại thời điểm người dùng lấy dữ liệu từ vùng chứa an toàn để đọc. Ở giai đoạn đó, khi nội dung được giải mã để người dùng sử dụng, phần mềm gián điệp sẽ kiểm soát nội dung và gửi nội dung đó đi.

# GIỚI THIỆU VỀ KỸ THUẬT KHAI THÁC IOS

## IOS Jailbreaking

### Khái niệm

Jailbreaking là quá trình cài đặt một bản vá kernel đã được sửa đổi cho phép người dùng chạy các ứng dụng của bên thứ ba không được nhà cung cấp cho phép. Đó là quá trình vượt qua các giới hạn của người dùng do apple đặt ra, chẳng hạn như sửa đổi hệ điều hành, đạt được đặc quyền quản trị viên và cài đặt các ứng dụng được phê duyệt không chính thức thông qua “side loading”. Có thể thực hiện jailbreaking bẻ khóa bằng cách sửa đổi kernel hệ thống IOS. Một số lý do để bẻ khóa các thiết bị ios như iphone, iPad và iPod Touch là để mở rộng bộ tính năng bị hạn chế bởi Apple và App store của hãng. Jailbreaking cung cấp quyền truy cập root vào HĐH và cho phép tải xuống các ứng dụng , chủ đề và tiện ích mở rộng của bên thứ ba không có sẵn thông qua Apple App Store chính thức. Jailbreaking cũng loại bỏ các hạn chế sandbox, cho phép app độc hại truy cập vào thông tin và tài nguyên di động bị hạn chế. Sử dụng các công cụ như Cydia, hexxa Plus, … để bẻ khóa các thiết bị IOS

### Phân loại

**Khai thác Userland**

Userland Exploit sử dụng lỗ hổng trong ứng dụng hệ thống. Nó cho phép truy cập ở cấp độ người dùng nhưng không cho phép truy cập ở cấp độ iboot. Bạn không thể bảo mật thiết bị iOS trước cách khai thác này vì không có gì có thể gây ra vòng lặp chế độ khôi phục. Chỉ các bản cập nhật chương trình cơ sở mới có thể vá các lỗ hổng như vậy.

**Khai thác iBoot**

Kiểu khai thác này có thể được kết nối một nửa nếu thiết bị có bootrom mới. Jailbreak iboot cho phép truy cập ở cấp độ người dùng và cấp độ iboot. Cách khai thác này lợi dụng lỗ hổng trong iBoot (bộ tải khởi động thứ ba của iDevice) để hủy liên kết công cụ ký mã. Các bản cập nhật chương trình cơ sở có thể vá các lỗi khai thác như vậy.

**Khai thác bootrom**

Bootrom Exploit sử dụng lỗ hổng trong SecureROM (bộ tải khởi động đầu tiên của iDevice) để vô hiệu hóa kiểm tra chữ ký, lỗ hổng này có thể được sử dụng để tải bản vá chương trình cơ sở NOR. Các bản cập nhật chương trình cơ sở không thể vá các lỗi khai thác như vậy. Bẻ khóa bootrom cho phép truy cập ở cấp độ người dùng và cấp độ iboot. Chỉ có bản cập nhật phần cứng bootrom của Apple mới có thể vá lỗi khai thác này.

### Kỹ thuật Jailbreaking

Untethered Jailbreaking (Bẻ khóa không giới hạn): nếu người dùng tắt hoặc bật lại thiết bị, thiết bị sẽ khởi động lại hoàn toàn và bản kernel được vá lỗi mà không cần sự trợ giúp của máy tính nhưng nó vẫn có thể sử dụng các chức năng bình thường.

Semi-tethered jailbreaking (Bẻ khóa 1 nửa): nếu người dùng tắt và bật lại thiết bị, thiết bị sẽ khởi động hoàn toàn. Nó sẽ không còn có kernel được nhưng vẫn có thể sử dụng được cho các chức năng bình thường. Để sử dụng các addon đã bẻ khóa, người dùng cần khởi động thiết bị với sự trợ giúp của tool jailbreaking.

Tethered jailbreaking: nếu thiết bị tự khởi động lại, thiết bị sẽ không còn nhân được vá và có thể bị kẹt trong quá trình khởi động một phần tình trạng; để nó khởi động hoàn toàn với kernel đã được vá, nó phải được "bẻ khóa lại" bằng máy tính (sử dụng tính năng "boot tethered" của công cụ bẻ khóa) mỗi lần nó được bật

Semi-untethered jailbreaking: Bẻ khóa bán không dây cũng tương tự như bẻ khóa bán không dây. Trong kiểu bẻ khóa này, khi thiết bị khởi động lại, kernel không được vá nhưng kernel vẫn có thể được vá mà không cần sử dụng máy tính. Việc này được thực hiện bằng ứng dụng được cài đặt trên thiết bị.

## Kỹ thuật tấn công thiết bị IOS

### Kỹ thuật SSL pinning bypass

SSL pinning là một kỹ thuật được sử dụng để củng cố bảo mật cho các ứng dụng di động và trình duyệt web khi kết nối đến các máy chủ web thông qua HTTPS.

Khi một ứng dụng hoặc trình duyệt web kết nối đến một máy chủ thông qua HTTPS, thông thường nó sẽ kiểm tra chứng chỉ SSL/TLS của máy chủ để đảm bảo rằng nó đang giao tiếp với máy chủ chính xác và không bị tấn công trung gian. Tuy nhiên, một kẻ tấn công có thể cài đặt chứng chỉ SSL giả mạo và đưa ứng dụng hoặc trình duyệt web vào tình trạng tin tưởng vào kết nối không an toàn này.

Với SSL pinning, thay vì chỉ kiểm tra tính hợp lệ của chứng chỉ SSL/TLS, ứng dụng hoặc trình duyệt web sẽ lưu trữ một bản sao của chứng chỉ này trong bộ nhớ của nó. Khi kết nối lại được thực hiện, ứng dụng hoặc trình duyệt web sẽ so sánh chứng chỉ này với chứng chỉ mới từ máy chủ. Nếu chúng không khớp, kết nối sẽ bị từ chối

Các kỹ thuật bypass SSL pinning thường được thực hiện thông qua việc thay đổi hoặc làm giả chứng chỉ SSL được sử dụng để xác minh kết nối giữa ứng dụng và máy chủ. Dưới đây là một số phương pháp cụ thể:

* Dynamic Instrumentation Frameworks: Các công cụ như Frida, objection, Cycript cho phép bạn thực hiện hook vào hàm kiểm tra chứng chỉ SSL pinning trong quá trình runtime. Khi hook vào hàm này, bạn có thể thay đổi hoặc bypass quá trình kiểm tra chứng chỉ, cho phép kết nối tiếp tục mà không cần chứng chỉ được "pinned".
* Custom TrustManager: Trong các ứng dụng Android, một phương pháp là triển khai một TrustManager tùy chỉnh để bypass kiểm tra chứng chỉ. Bằng cách này, ứng dụng sẽ không kiểm tra chứng chỉ được pin và cho phép các chứng chỉ không được tin cậy để thiết lập kết nối.
* Certificate Swapping: Đôi khi, việc thay đổi chứng chỉ trên thiết bị để thay thế chứng chỉ được pin có thể làm giảm tính bảo mật. Tuy nhiên, điều này đòi hỏi quyền truy cập root hoặc quyền truy cập hệ thống cao.
* SSL Pinning Bypass Tools: Có một số công cụ và thư viện mã nguồn mở cung cấp cách tự động bypass SSL pinning. Các công cụ này thường là các mã nguồn đã được phát triển và được cộng đồng sử dụng, nhưng việc sử dụng chúng cần phải cân nhắc kỹ lưỡng vì có thể gây ra các vấn đề an ninh.

Một số ứng dụng sử dụng các kỹ thuật phức tạp và mạnh mẽ để ngăn chặn việc bypass SSL pinning. Việc thành công trong việc bypass phụ thuộc vào cách thức triển khai SSL pinning trong ứng dụng cũng như cách thức ứng dụng xử lý các kỹ thuật bypass.

### Insecure Local Data Storage

Tấn công vào "Insecure Local Data Storage" (Lưu trữ dữ liệu cục bộ không an toàn) là quá trình tận dụng các lỗ hổng hoặc thiếu sót trong cách thức ứng dụng lưu trữ và xử lý dữ liệu trên thiết bị người dùng hoặc trên máy chủ. Điều này có thể dẫn đến việc thông tin nhạy cảm bị đánh cắp hoặc thay đổi bởi những người không được ủy quyền.

Có một số kỹ thuật tấn công mà tin tặc thường sử dụng để tấn công vào dữ liệu cục bộ không an toàn:

a) Tấn công MITM (Man-in-the-Middle)

Tấn công MITM xảy ra khi kẻ tấn công chèn mình vào giữa hai bên trong giao tiếp (ví dụ: giữa ứng dụng và máy chủ) và thu thập, sửa đổi hoặc chặn thông tin truyền qua. Điều này có thể làm cho dữ liệu lưu trữ không an toàn khi được truyền đi hoặc nhận về từ máy chủ.

b) Tấn công đọc Dữ Liệu (Data Leakage)

Tấn công này xảy ra khi thông tin nhạy cảm được lưu trữ mà không được mã hóa hoặc bảo vệ đúng cách, cho phép kẻ tấn công truy cập và đọc dữ liệu một cách dễ dàng từ các tệp tin, cơ sở dữ liệu hoặc cache của ứng dụng.

c) Tấn công Brute Force và Credential Stuffing

Nếu dữ liệu đăng nhập hoặc mật khẩu được lưu trữ một cách không an toàn, tin tặc có thể thực hiện các cuộc tấn công brute force (tấn công mò mật khẩu) hoặc credential stuffing (sử dụng thông tin đăng nhập đã bị đánh cắp từ các vụ việc khác) để đăng nhập trái phép vào tài khoản người dùng.

d) Tấn công Tampering (Can thiệp vào dữ liệu)

Kẻ tấn công có thể can thiệp vào dữ liệu cục bộ để thay đổi hoặc làm giả dữ liệu, làm giảm tính toàn vẹn của thông tin được lưu trữ. Điều này có thể làm ảnh hưởng đến tính tin cậy của ứng dụng hoặc thông tin của người dùng.

e) Tấn công Injection

Các loại tấn công injection như SQL injection hoặc code injection có thể được sử dụng để chèn các đoạn mã độc hại hoặc truy vấn SQL không mong muốn vào dữ liệu cục bộ, từ đó làm mất kiểm soát hoặc tiết lộ thông tin quan trọng.

### Fingerprint Bypass

Fingerprint bypass là quá trình mà kẻ tấn công cố gắng vượt qua hoặc bypass các hệ thống nhận diện dấu vân tay (fingerprint) trên thiết bị di động hoặc máy tính để truy cập trái phép vào thông tin hoặc chức năng bảo mật được bảo vệ bởi dấu vân tay.

Có một số kỹ thuật và phương pháp mà kẻ tấn công có thể sử dụng để bypass dấu vân tay:

a) Sử dụng vật liệu để tạo dấu vân tay giả mạo

Kẻ tấn công có thể sử dụng silicone, cao su, hoặc các chất liệu khác để tạo ra một bản sao chính xác của dấu vân tay từ người dùng thực sự. Bản sao này sau đó được sử dụng để đăng nhập hoặc mở khóa thiết bị.

b) Thu thập dấu vân tay từ hình ảnh

Trong một số trường hợp, dấu vân tay có thể được thu thập từ hình ảnh chất lượng cao hoặc từ các bề mặt mà người dùng đã chạm vào. Kẻ tấn công có thể sử dụng hình ảnh này để tạo ra một bản sao của dấu vân tay và sử dụng nó để bypass.

c) Sử dụng công nghệ 3D hoặc in ấn

Một kỹ thuật phức tạp hơn là sử dụng công nghệ in ấn 3D để tạo ra một bản sao chính xác hơn của dấu vân tay. Điều này có thể đòi hỏi việc sử dụng công nghệ in ấn 3D hoặc mô hình học máy để tạo ra một bản sao vân tay có độ chính xác cao.

d) Tấn công phần cứng

Một phương pháp khác có thể là tấn công trực tiếp vào phần cứng của thiết bị để bypass cảm biến vân tay. Kẻ tấn công có thể can thiệp vào cảm biến hoặc các phần khác của hệ thống vân tay để tạo điều kiện cho việc vượt qua cơ chế bảo mật.

## Tool pentesting objection

### Khái Niệm:

Objection là một công cụ kiểm thử tấn công (pentesting tool) được sử dụng trong lĩnh vực kiểm thử bảo mật, đặc biệt là cho ứng dụng di động chạy trên nền tảng Android và iOS. Nó được xây dựng trên nền tảng Frida và cung cấp khả năng tương tác động với quá trình chạy của ứng dụng.

### Cấu trúc:

1.Giao Diện Dòng Lệnh (CLI):

Giao diện dòng lệnh của Objection là nơi người dùng nhập các lệnh để thực hiện các tác vụ kiểm thử và tấn công.

Cung cấp các tùy chọn và lệnh để tương tác với ứng dụng di động và thực hiện các chức năng kiểm thử bảo mật.

2.Frida Backend:

Frida đóng vai trò quan trọng trong cấu trúc của Objection. Nó là một framework cho phép tương tác động với quá trình chạy của ứng dụng.

Objection sử dụng Frida để chèn mã tùy chỉnh vào quá trình chạy của ứng dụng và theo dõi sự thay đổi, can thiệp, hoặc thu thập thông tin.

3.Plugins:

Objection hỗ trợ các plugin để mở rộng khả năng của công cụ. Các plugin có thể thực hiện các chức năng cụ thể như phân tích bảo mật, can thiệp vào giao tiếp mạng, hoặc thực hiện các tác vụ tự động hóa.

Các plugin giúp mở rộng chức năng và linh hoạt cho người sử dụng.

4.Scripts:

Objection cho phép người dùng thực hiện các tác vụ thông qua các script. Các script này có thể được viết bằng JavaScript, giúp người dùng tùy chỉnh và mở rộng khả năng của Objection.

5.Tùy Chọn Cấu Hình:

Các tùy chọn cấu hình cho phép người dùng điều chỉnh cách Objection tương tác với ứng dụng và thiết lập các thông số khác nhau như địa chỉ IP, cổng kết nối, v.v.

### Cách hoạt động:

1.Kết Nối và Chèn Frida Mã:

Objection sử dụng Frida để kết nối với quá trình chạy của ứng dụng di động. Frida có khả năng chèn mã JavaScript vào quá trình chạy.

Khi bạn khởi động Objection, nó sẽ tạo kết nối với Frida server trên thiết bị di động và chèn mã Frida vào quá trình chạy của ứng dụng.

2.Tương Tác với Quá Trình Chạy:

Mã JavaScript được chèn thông qua Frida sẽ tương tác động với quá trình chạy của ứng dụng. Điều này cho phép theo dõi, can thiệp và thậm chí thay đổi hành vi của ứng dụng trong thời gian chạy.

3.Giao Diện Dòng Lệnh (CLI):

Người dùng tương tác với Objection thông qua giao diện dòng lệnh (CLI). Tại đây, họ có thể nhập các lệnh để thực hiện các chức năng kiểm thử và tấn công.

4.Sử Dụng Plugins và Scripts:

Objection hỗ trợ việc sử dụng các plugin và scripts. Các plugin cung cấp chức năng mở rộng, trong khi scripts cho phép thực hiện các tác vụ tùy chỉnh.

Các lệnh và chức năng của Objection có thể được mở rộng bằng cách sử dụng các plugin có sẵn hoặc viết các script tùy chỉnh.

5.Thực Hiện Các Chức Năng Kiểm Thử:

Objection cung cấp các lệnh để kiểm thử bảo mật ứng dụng, bao gồm theo dõi và can thiệp vào giao tiếp mạng, kiểm tra lỗ hổng bảo mật, và thậm chí tự động hóa một số hành động tấn công.

6.Tương Tác với Các Thành Phần Cụ Thể Của Ứng Dụng:

Người dùng có thể tương tác với các thành phần cụ thể của ứng dụng, như các hàm, biến, hoặc luồng thực thi, để kiểm tra và kiểm thử tính bảo mật.

### Tác dụng:

1.Kiểm Thử Bảo Mật Toàn Diện:

Objection cung cấp một môi trường kiểm thử bảo mật mạnh mẽ cho ứng dụng di động. Nó giúp chuyên gia bảo mật xác định và đánh giá lỗ hổng bảo mật từng chi tiết trong quá trình chạy của ứng dụng.

2.Theo Dõi và Can Thiệp vào Giao Tiếp Mạng:

Objection cho phép người dùng theo dõi và can thiệp vào giao tiếp mạng của ứng dụng di động. Điều này giúp kiểm tra tính bảo mật của các phương thức kết nối và truyền thông.

3.Bypassing SSL Pinning:

Một trong những tác dụng quan trọng của Objection là hỗ trợ việc bypassing SSL pinning. SSL pinning là một biện pháp bảo mật để đảm bảo an toàn trong giao tiếp mạng, và Objection giúp bỏ qua nó để kiểm thử tính bảo mật hiệu quả hơn.

4.Kiểm Thử Động và Tương Tác Thời Gian Chạy:

Objection cho phép chuyên gia bảo mật thực hiện kiểm thử động trực tiếp trong thời gian chạy của ứng dụng. Điều này giúp đào sâu vào quá trình thực thi và định rõ các lỗ hổng có thể xuất hiện.

5.Tự Động Hóa Các Tác Vụ Kiểm Thử:

Các plugin và scripts của Objection cho phép tự động hóa một số tác vụ kiểm thử. Điều này giúp tăng cường hiệu suất và đồng thời giảm khối lượng công việc lặp lại cho người thử nghiệm.

6.Phân Tích Ứng Dụng Đa Nền Tảng:

Objection không chỉ hỗ trợ kiểm thử trên nền tảng Android mà còn trên iOS. Điều này giúp chuyên gia bảo mật có khả năng kiểm thử và đánh giá tính bảo mật trên nhiều hệ điều hành di động.

7.Theo Dõi và Can Thiệp vào Các Thành Phần Cụ Thể:

Objection cho phép người dùng tương tác với các thành phần cụ thể của ứng dụng như hàm, biến, và luồng thực thi, giúp kiểm tra và kiểm thử tính bảo mật của từng phần của ứng dụng.

8.Mở Rộng và Linh Hoạt:

Các plugin và scripts mở rộng khả năng của Objection, giúp người sử dụng có thể thích ứng với các tình huống và yêu cầu kiểm thử cụ thể.

### Cách cài đặt.

Để cài đặt công cụ kiểm thử tấn công Objection, bạn cần thực hiện một số bước cài đặt cơ bản.

### Yêu Cầu Tiên Quyết:

1.Python:

Objection được xây dựng bằng Python, vì vậy bạn cần cài đặt Python trên hệ thống của mình. Bạn có thể tải Python từ trang chính thức: https://www.python.org/.

2.Frida:

Objection sử dụng Frida như một backend, nên bạn cần cài đặt Frida trước khi sử dụng Objection. Cài đặt Frida bằng cách sử dụng npm (Node Package Manager) với lệnh sau:

npm install -g frida

### Cài Đặt Objection:

1.Sử dụng pip:

Mở terminal hoặc command prompt và nhập lệnh sau để cài đặt Objection:

pip install objection

2.Kiểm Tra Việc Cài Đặt:

Để kiểm tra xem Objection đã được cài đặt thành công hay không, bạn có thể nhập lệnh sau:

objection –version

Nếu bạn nhìn thấy phiên bản của Objection, điều này có nghĩa là cài đặt đã thành công.

### Khởi Chạy Objection:

1.Mở Giao Diện Dòng Lệnh:

Mở terminal hoặc command prompt và nhập lệnh sau để bắt đầu sử dụng Objection:

objection

2.Kết Nối Đến Ứng Dụng Di Động:

Sau khi Objection đã chạy, bạn cần kết nối nó với ứng dụng di động bạn muốn kiểm thử. Sử dụng lệnh sau để kết nối:

android connect

3.Kiểm Tra Kết Nối:

Để kiểm tra xem kết nối đã được thiết lập chưa, nhập lệnh sau:

android status

4.Bắt Đầu Sử Dụng:

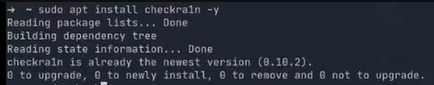
Bây giờ bạn đã kết nối thành công và có thể bắt đầu sử dụng Objection để kiểm thử và tương tác với ứng dụng.

# THỰC NGHIỆM

## Chuẩn bị

Cài đặt xcode

Cài đặt DVIA

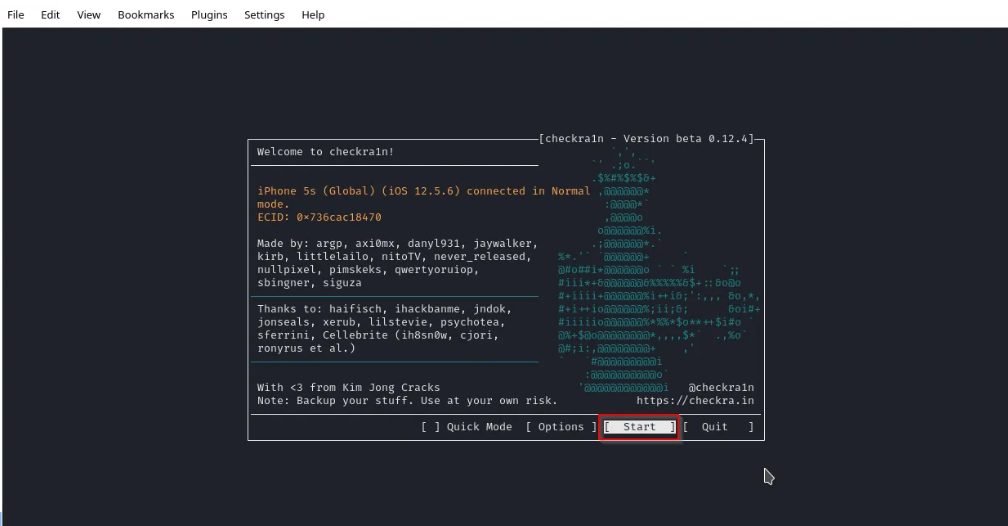
 Máy kali cài đặt checkta1n

Hình . Cài đặt checkaln

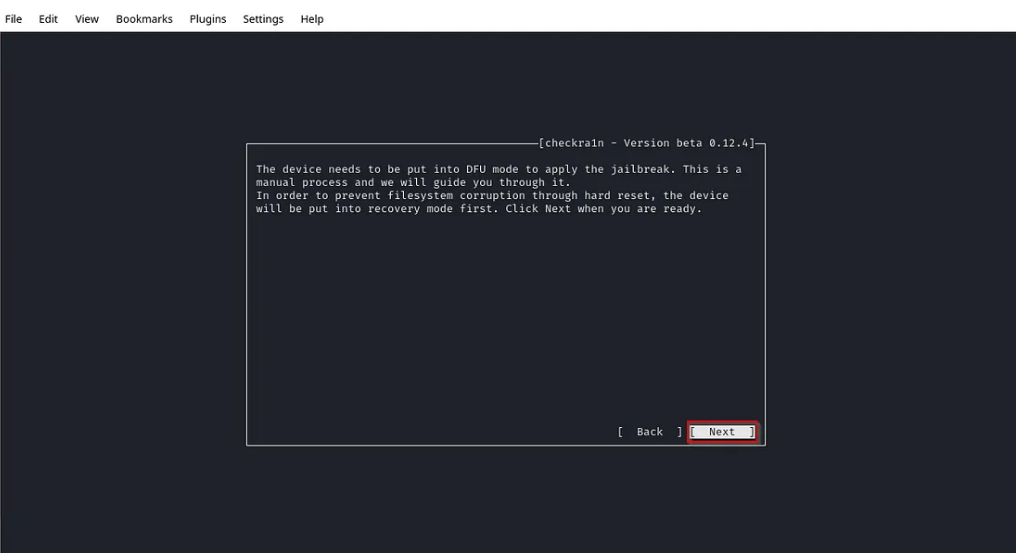
## Kịch bản IOS Jailbreaking

Kết nối thiết bị iOS với máy kali thông qua ssh: ssh root@ mật khẩu là Alpine

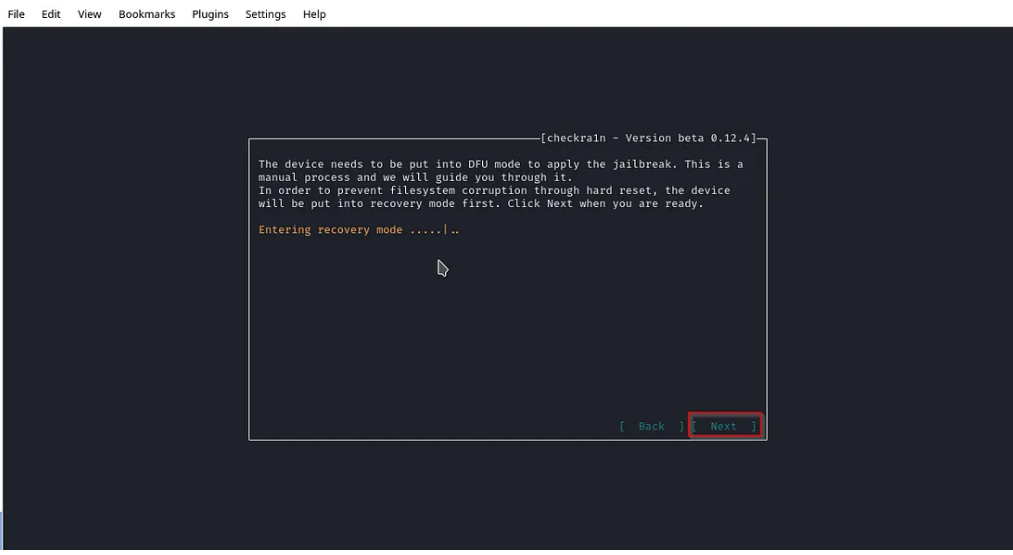
Mở terminal trên máy kali chạy lệnh *sudo checkra1n* để khởi động công cụ checkra1n

 Nhấp vào nút “Bắt đầu” trong giao diện checkra1n. Điều này sẽ đưa thiết bị của bạn vào chế độ phục hồi.

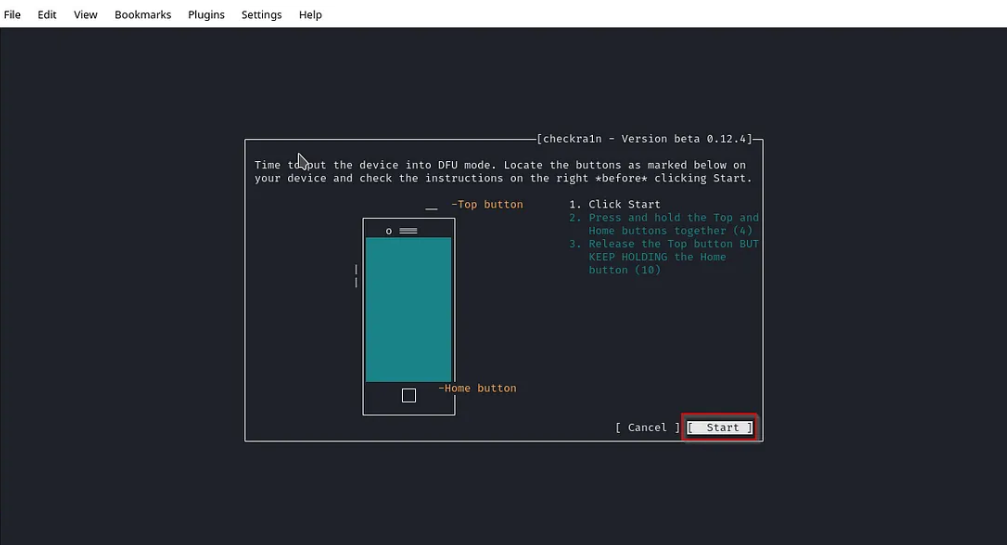
Hình . Giao diện checkaln



Hình . Thông báo trước khi vào recovery mode

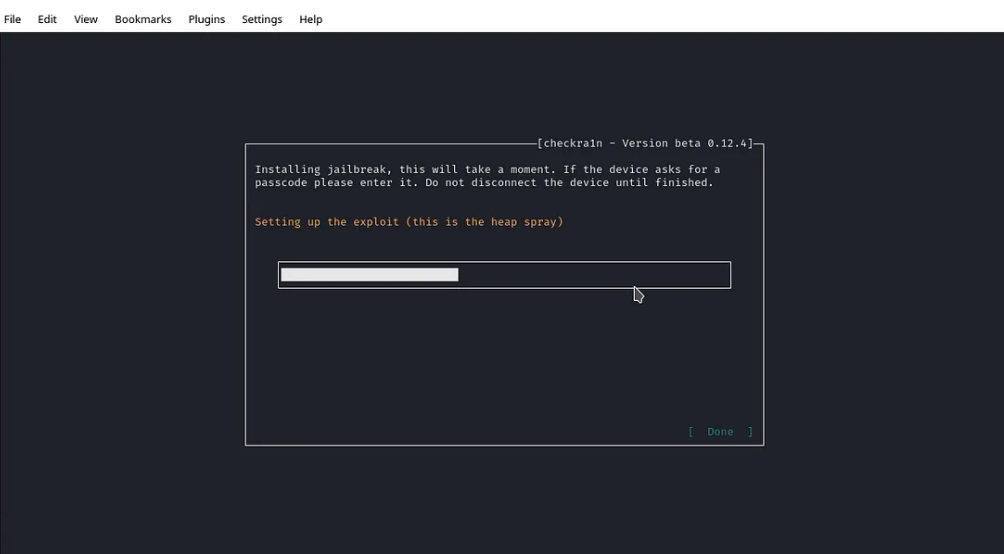


Hình . Recovery mode

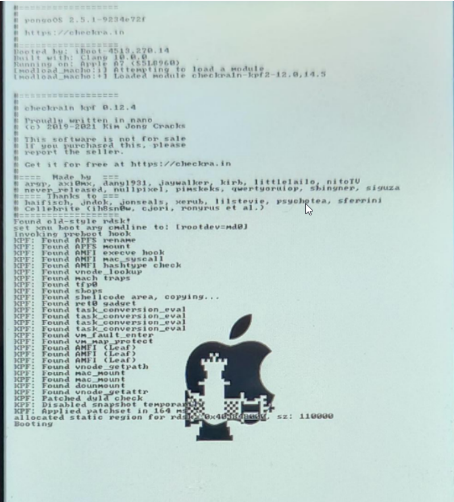
 Làm theo hướng dẫn trên màn hình để đưa thiết bị của bạn vào chế độ DFU (Cập nhật chương trình cơ sở thiết bị).

Hình . Đưa thiết bị vào chế độ DFU

Hình . Bắt đầu tải jailbreaking

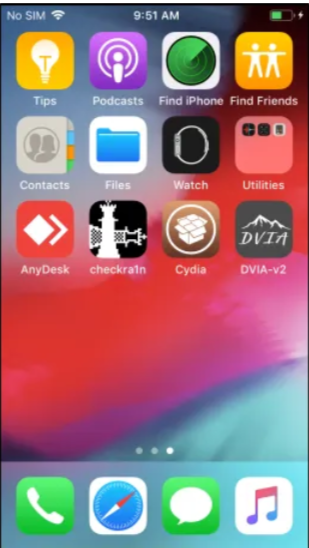


Quá trình bẻ khóa sẽ bắt đầu và thiết bị sẽ được khởi động lại sau khi hoàn tất.



Hình .

Sau khi thiết bị của bạn khởi động lại sau khi bẻ khóa, bạn sẽ thấy biểu tượng ứng dụng checkra1n trên màn hình chính.



Hình . Biểu tượng checkaln ở màn hình chính

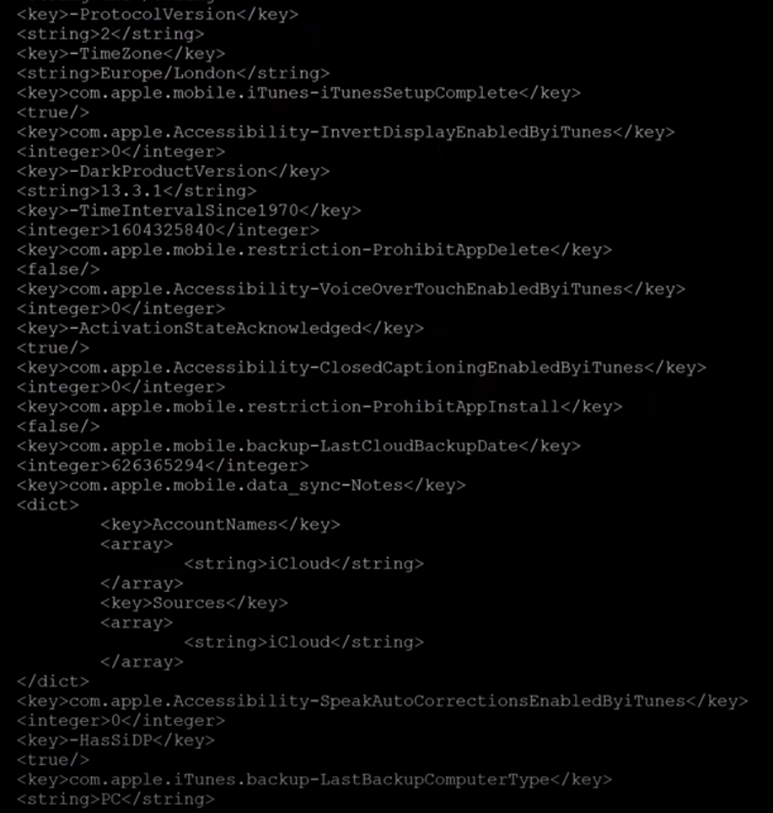
## Kịch bản Hacking IOS

### Insecure Local data Storage

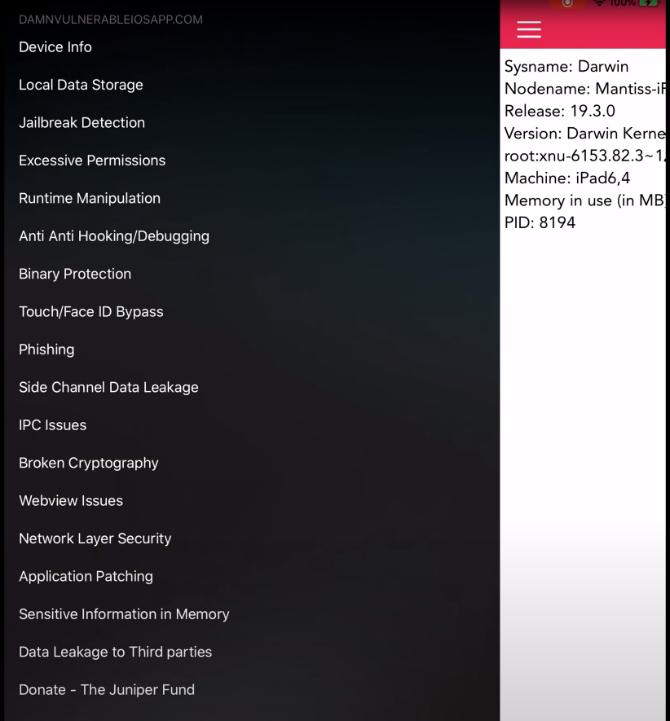
1. Khai thác tệp Plist trong App Directory

Plist (Property List) là định dạng linh hoạt và tiện lợi để lưu trữ dữ liệu cấu hình ứng dụng. Chúng ta có thể gọi nó là bảng kê khai cho một ứng dụng iOS. Đó là một tập tin danh sách tài sản từ Apple. Nó quyết định nên sử dụng biểu tượng nào cho một gói, loại tài liệu nào mà ứng dụng có thể hỗ trợ và nhiều hành vi khác có tác động bên ngoài gói đó. Các tệp này có thể chứa dữ liệu nhạy cảm như khóa API Gmaps, v.v.

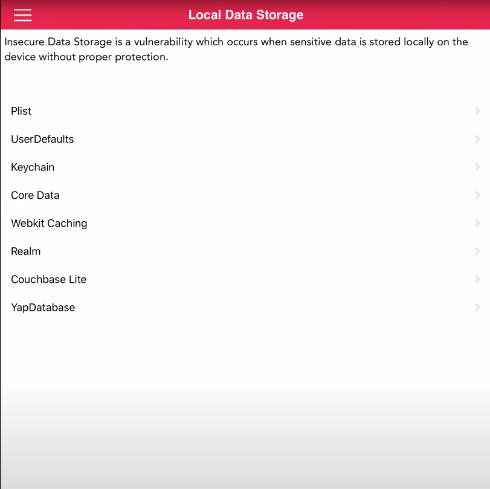
B1: Đọc file Plist trên máy giả lập bằng lệnh *plistutil -i info.plist*



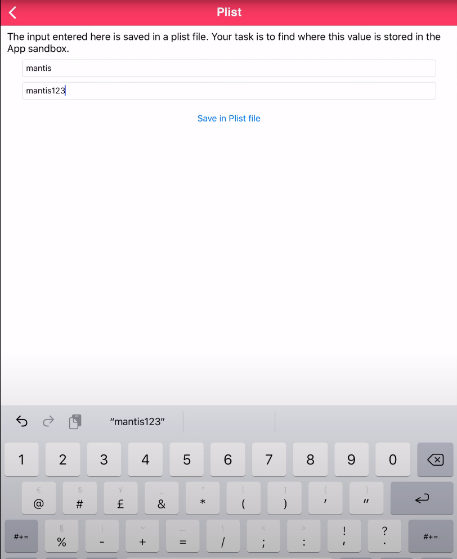
Hình . Đọc file Plist

B2: Vào máy giả lập chọn DVIA-v2 sau đó chọn Local Data Storage

Hình . Giao diện DVIA-v2

B3: Chọn Plist

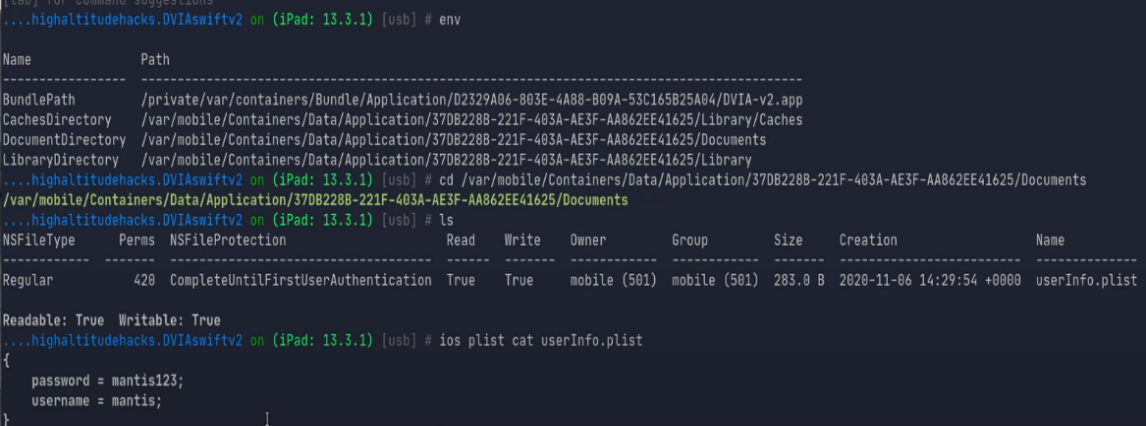
Hình . Giao diện Local Data Storage

B4: Nhập 1 username, password bất kỳ lưu vào file plist

Hình . Giao diện Plist

B5: Chạy objection

Hình . Chạy objection

B6: Sau khi khai thác thành công kiểm tra file

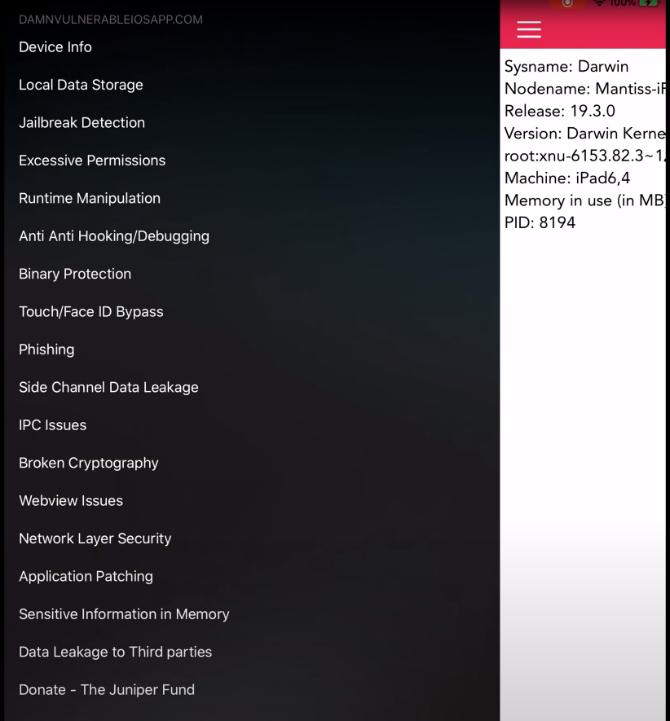
Hình . Kiểm tra file

B7: Thấy được username, password vừa lưu

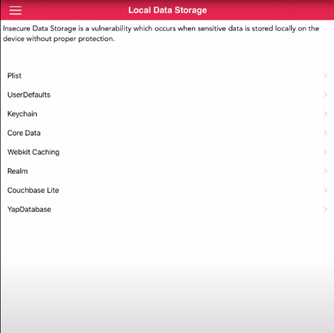
2. Khai thác NSUserDefaults

NSUserDefault là một class cung cấp một giải pháp để lưu trữ dữ liệu một cách đơn giản. Dữ liệu sau khi được lưu thì kể cả reset thiết bị nó cũng vẫn được lưu lại. Có thể sử dụng NSUserDefaults để lưu trữ các kiểu dữ liệu cơ bản như: Bool, Float, Double, Int, String hoặc NSURL, arrays, dictionaries, NSData, NSData

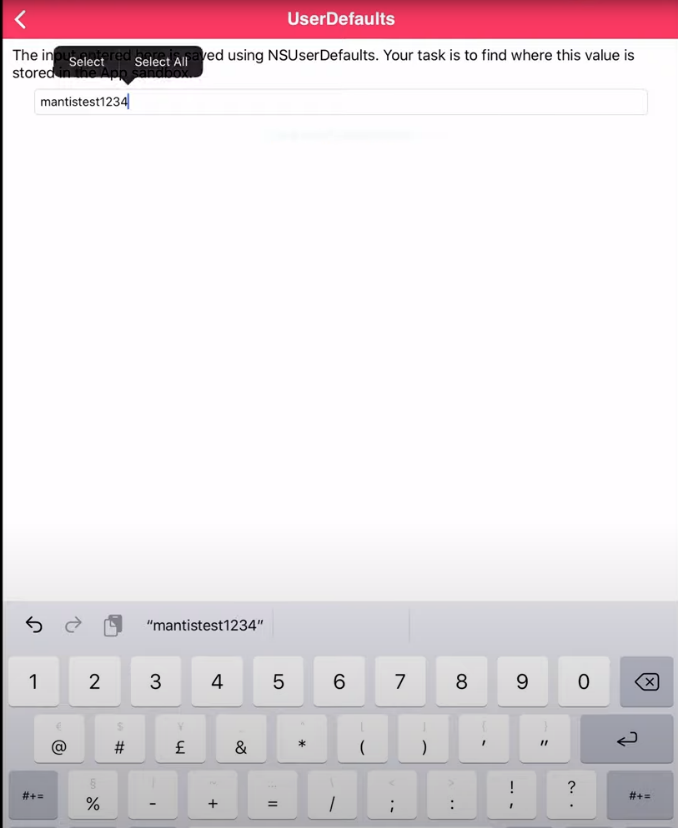
B1: vào thiết bị ios giả lập chọn ứng dụng DVIA-v2



Hình . Giao diện DVIA-v2

B2: Chọn NSUserDefaults

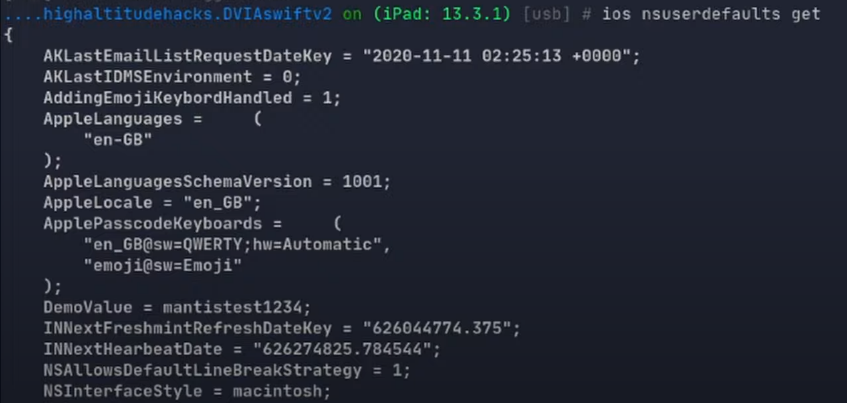
Hình . Giao diện Local Data Storage

B3: Nhập nội dung và lưu vào NSUserDefaults

Hình . Nhập nội dung

B4: chạy Objection

Hình . Chạy objection

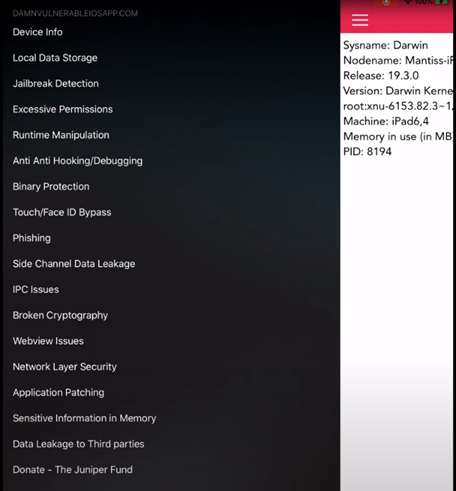
B5: Kiểm tra nsuserdefaults khai thác được thấy được giá trị đã nhập ở *DemoValue*

Hình . Kiểm tra nsuserdefaults

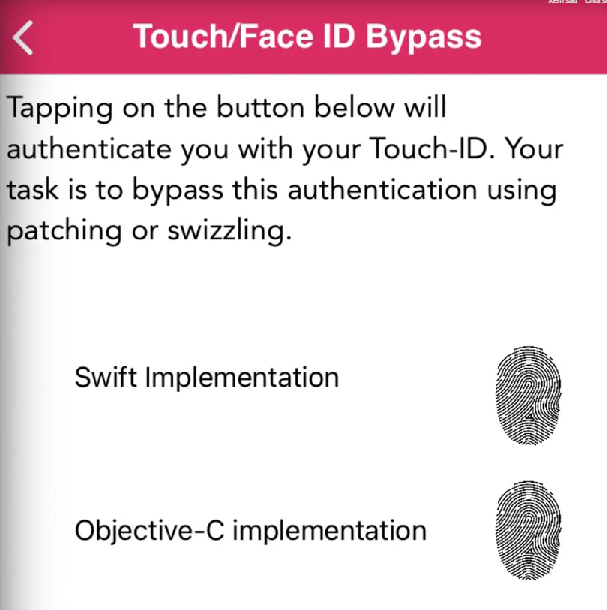
### Fingerprint Bypass

Để vượt qua sinh trắc học, bạn không cần phải có cùng dấu vân tay/khuôn mặt với mục tiêu. Thay vào đó, chúng ta có thể đi thẳng vào cơ chế kiểm tra dấu vân tay. Nếu nó được triển khai không an toàn thì chúng ta có thể bỏ qua việc kiểm tra

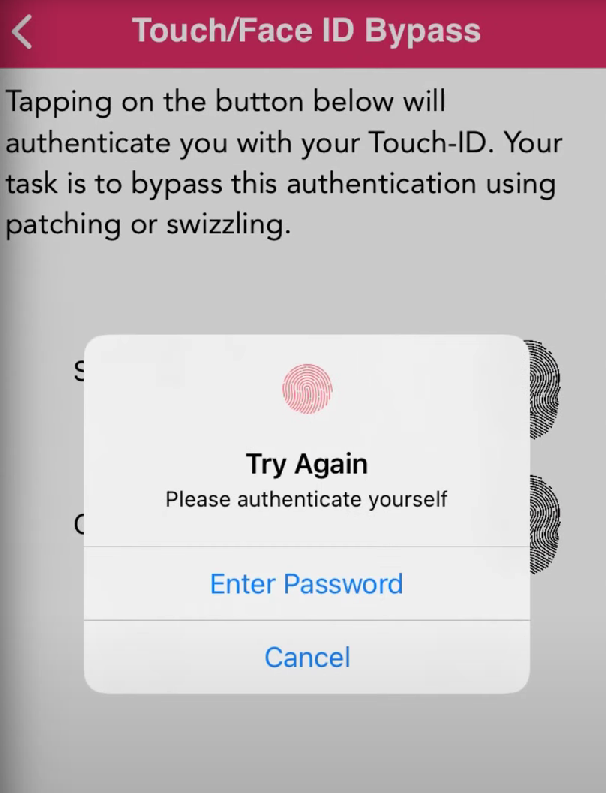
Bạn nên cài đặt dấu vân tay trên điện thoại trước khi thực hiện việc này

B1: vào thiết bị giả lập chọn DVIA-v2 sau đó chọn Touch/Face ID Bybass

Hình . Giao diện DVIA-v2

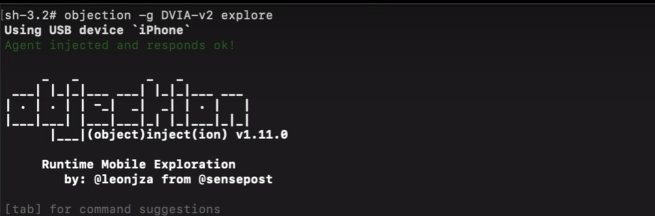
B2: Chọn Objective-C sau đó sử dụng vân tay không chính xác, kích hoạt phản hồi cho biết thông tin nhập của bạn sai

Hình . Giao diện Toucj/Face ID Bypass

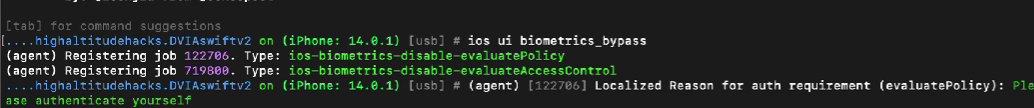


Hình . Kích hoạt thông báo

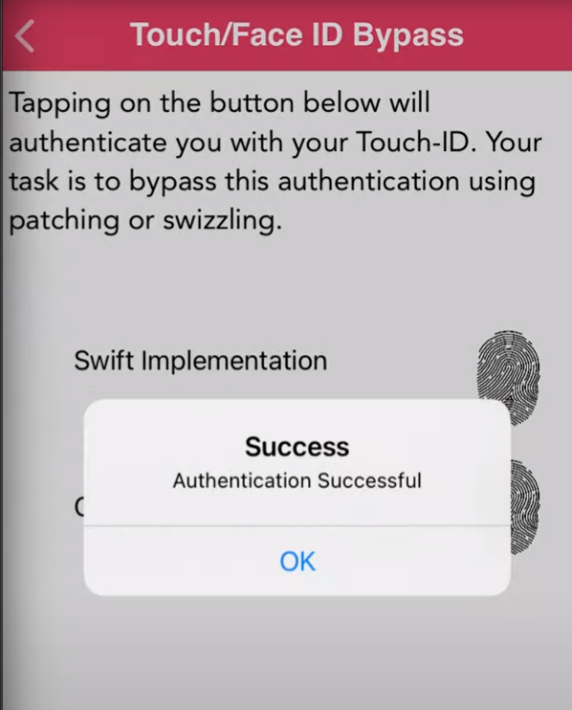
B3: chạy objection

*objection — gadget DVIA-v2 explore*

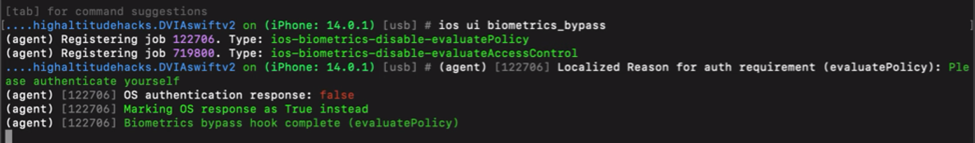
Hình . Chạy objection

**B4: chạy lệnh *ios ui bimetrucs\_bybass* kích hoạt khai thác

Hình . Kích hoạt khai thác

**B5: Thử lại vân tay thì bybass thành công

Hình . Thử lại vân tay



Hình . Ghi lại thông tin sau khi khai thác thành công

KẾT LUẬN

Trong bài báo cáo này, chúng em đã nêu cơ sở lí thuyết cũng như mô tả và cách khai thác một số kỹ thuật. Trong quá trình thực hiện bài tập báo cáo này, do kiến thức cũng như kinh nghiệm của nhóm chúng em còn hạn chế nên một số phần của bài thực tập vẫn chưa được trình bày sâu và kỹ, tuy còn nhiều thiếu sót về mặt kiến thức nhưng chúng em cũng đã thực hiện khai thác thành công.

TÀI LIỆU THAM KHẢO

[1] https://github.com/prateek147/DVIA-v2

[2] https://www.yeswehack.com/learn-bug-bounty/getting-started-ios-penetration-testing-part-2

[3] https://arz101.medium.com/ios-pentesting-bypassing-jailbreak-detection-3502de588901

[4] https://www.cobalt.io/blog/ios-pentesting-101

[5] https://www.greycampus.com/opencampus/ethical-hacking/ios

[6] https://www.youtube.com/watch?v=QHY\_gtCM7y0