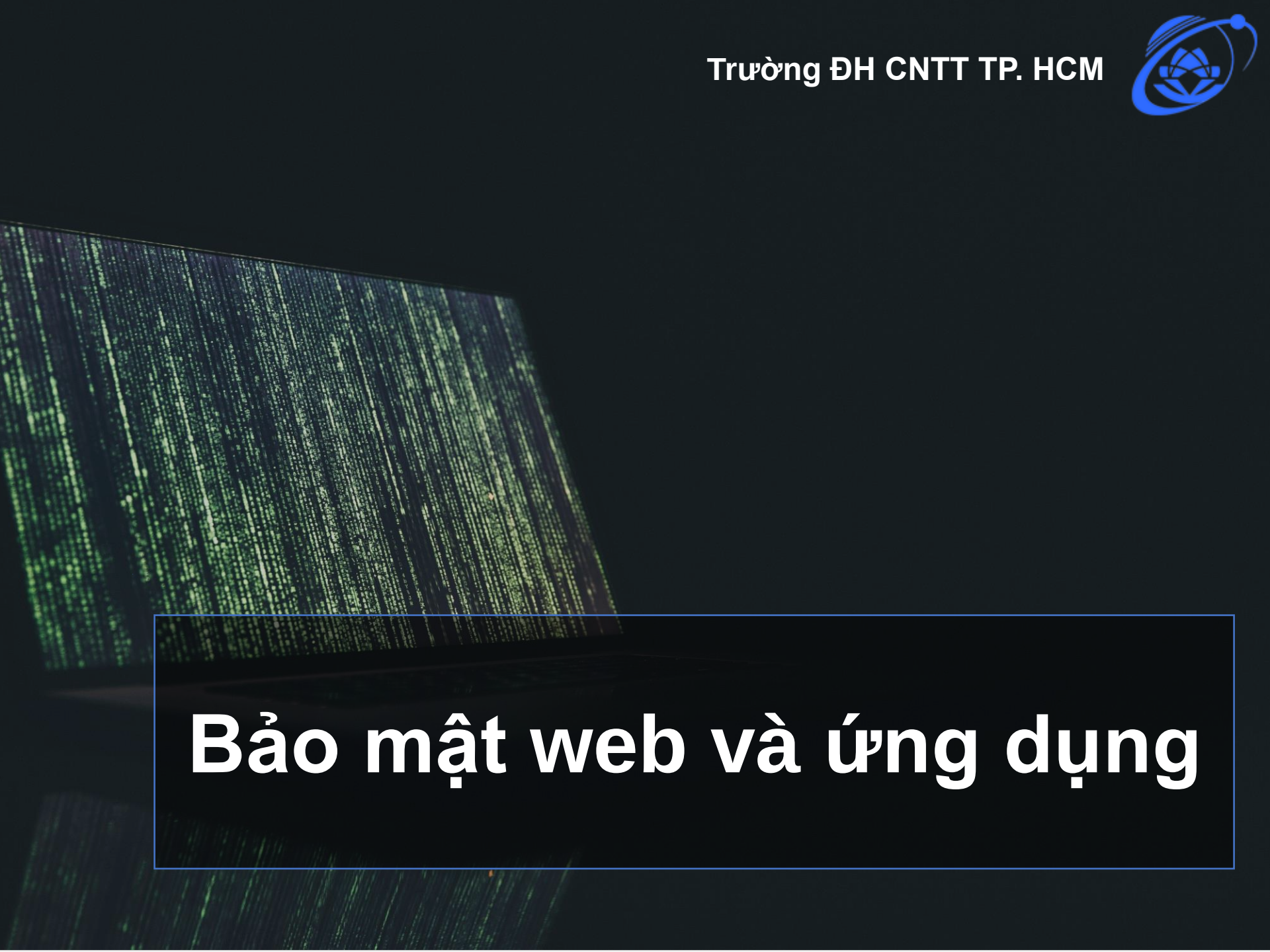
Trường ĐH CNTT TP. HCM



Bảo mật web và ứng dụng

Nội dung

- Chuẩn bảo mật OWASP Mobile Top 10
- Fuzzing testing

OWASP Mobile Top 10

- Tiêu chuẩn đánh giá mức độ bảo mật của ứng dụng dựa trên danh sách 10 loại lỗ hổng khác nhau cho nền tảng ứng dụng di động

https://owasp.org/www-project-mobile-top-10/

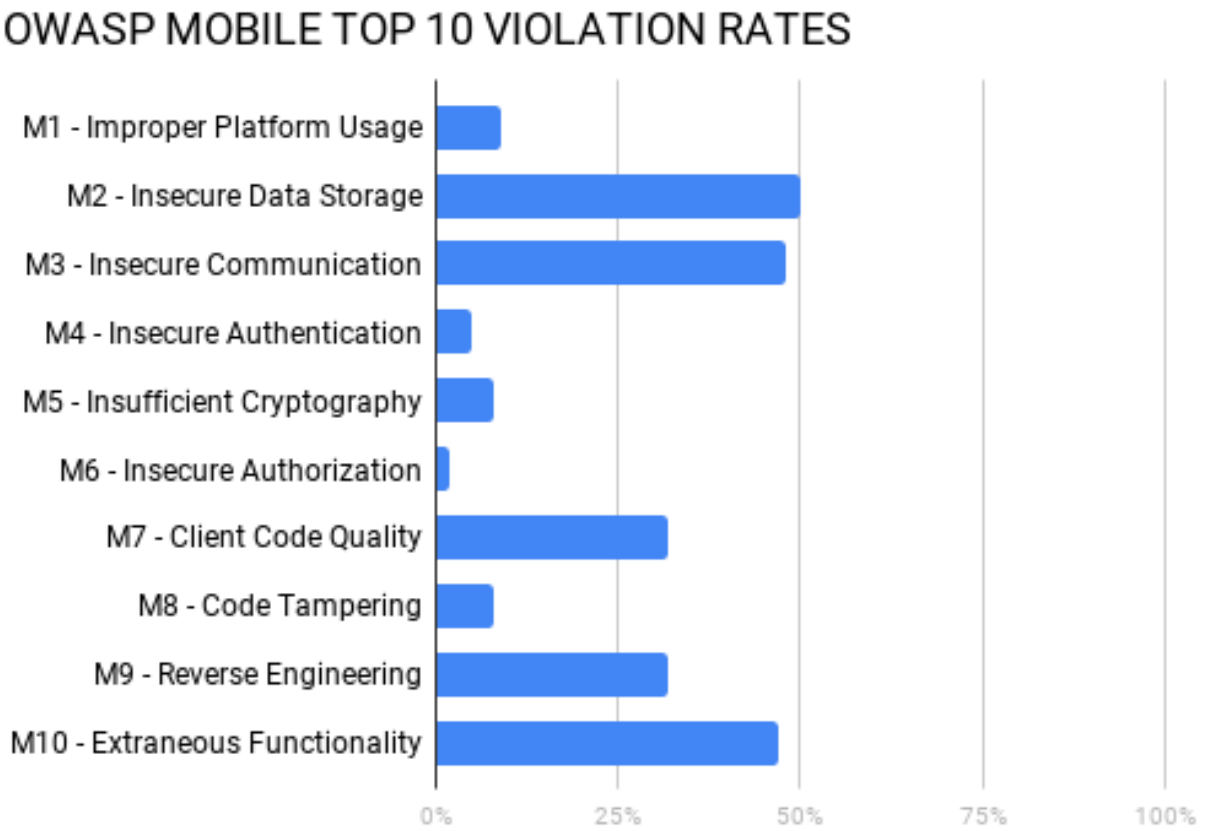
OWASP Mobile Top 10 2016

M1 - Improper Platform Usage	Misuse of features like Touch ID, permissions, Keychain
M2 - Insecure Data Storage	Data Leakage, client-side injection, weak server-side controls
M3 - Insecure Communication	Poor handshake, SSL/TLS/Cert issues, transfer in clear text
M4 - Insecure Authentication	Improper identity mgmt, weak session mgmt
M5 - Insufficient Cryptography	Lack of crypto, improper crypto use
M6 - Insecure Authorization	Improper local auth, forced browsing
M7 - Client Code Quality	Code mistakes eg. Buffer overflows, format string vulns
M8 - Code Tampering	Binary patching, method hooking/swizzling, memory mods
M9 - Reverse Engineering	Exposure to attacker reversing tools
M10 - Extraneous Functionality	Dev/QA inadvertent disabling security, hidden backdoors

OWASP Mobile Top 10 2024

Comparison Between 2016-2024		
OWASP-2016	**OWASP-2024-Release**	**Comparison Between 2016-2024**
M1: Improper Platform Usage	M1: Improper Credential Usage	**New**
M2: Insecure Data Storage	M2: Inadequate Supply Chain Security	**New**
M3: Insecure Communication	M3: Insecure Authentication / Authorization	**Merged M4&M6 to M3**
M4: Insecure Authentication	M4: Insufficient Input/Output Validation	**New**
M5: Insufficient Cryptography	M5: Insecure Communication	**Moved from M3 to M5**
M6: Insecure Authorization	M6: Inadequate Privacy Controls	**New**
M7: Client Code Quality	M7: Insufficient Binary Protections	**Merged M8&M9 to M7**
M8: Code Tampering	M8: Security Misconfiguration	**Rewording [M10]**
M9: Reverse Engineering	M9: Insecure Data Storage	**Moved from M2 to M9**
M10: Extraneous Functionality	M10: Insufficient Cryptography	**Moved from M5 to M10**

OWASP Mobile Top 10



- Ít nhất 85% các ứng dụng mắc phải 1 hay nhiều loại lỗ hổng.
- 50% các ứng dụng liên quan đến lỗ hổng ở việc truyền nhận và lưu trữ dữ liệu
- 1/3 ứng dụng liên quan đến các vấn đề về mã nguồn (code)

(Theo NowSecure, 2018)

OWASP Mobile Top 10

OWASP MOBILE TOP 10

M1 - Improper Platform Usage	Misuse of features like Touch ID, permissions, Keychain	
M2 - Insecure Data Storage	Data Leakage, client-side injection, weak server-side controls	50% Fail
M3 - Insecure Communication	Poor handshake, SSL/TLS/Cert issues, transfer in clear text	48% Fail
M4 - Insecure Authentication	Improper identity mgmt, weak session mgmt	5% Fail
M5 - Insufficient Cryptography	Lack of crypto, mproper crypto use	
M6 - Insecure Authorization	Improper local auth, forced browsing	2% Fail
M7 - Client Code Quality	Code mistakes eg. Buffer overflows, format string vulns	32% Fail
M8 - Code Tampering	Binary patching, method hooking/swizzling, memory mods	
M9 - Reverse Engineering	Exposure to attacker reversing tools	32% Fail
M10 - Extraneous Functionality	Dev/QA inadvertent disabling security, hidden backdoors	47% Fail

NowSecure™

M1: Improper credential usage

- Liên quan đến khía cạnh quản lý và bảo vệ thông tin xác thực như: API key, khoá bí mật, khoá công khai,...
- Thông tin xác thực được mã hoá cứng (hardcoded) và việc sử dụng thời gian để tạo khoá dễ bị khai thác.
- Thông tin khoá được lưu trữ hoặc sử dụng không đúng cách.
- Thông tin xác thực được gửi đi không được mã hoá thích hợp dẫn đến bị đánh cắp.
- Không áp dụng các quy trình xác thực nhiều bước (MFA).
- Không triển khai hoặc quản lý phiên truy cập (session) lỏng lẻo.

M2: Inadequate Supply Chain Security

Kẻ tấn công tận dụng các thư viện, package bên thứ ba sử dụng trong ứng dụng để khai thác mà không cần tấn công vào ứng dụng (Tấn công chuỗi cung ứng).

Tấn công chuỗi cung ứng sẽ lâu dài nên đòi hỏi phải kiểm tra thường xuyên các đoạn mã (review code) trong ứng dụng và kiểm soát các bản cập nhật của ứng dụng.

M3: Insecure authentication/Authorization

- Lỗ hổng này liên quan đến khâu xác thực người dùng và quản lý phiên đăng nhập
- M3 bao gồm các yếu tố sau:
  - Không xác thực, định danh được người dùng
  - Không thể duy trì định danh của người dùng khi có yêu cầu
  - Gặp điểm yếu trong quy trình quản lý phiên

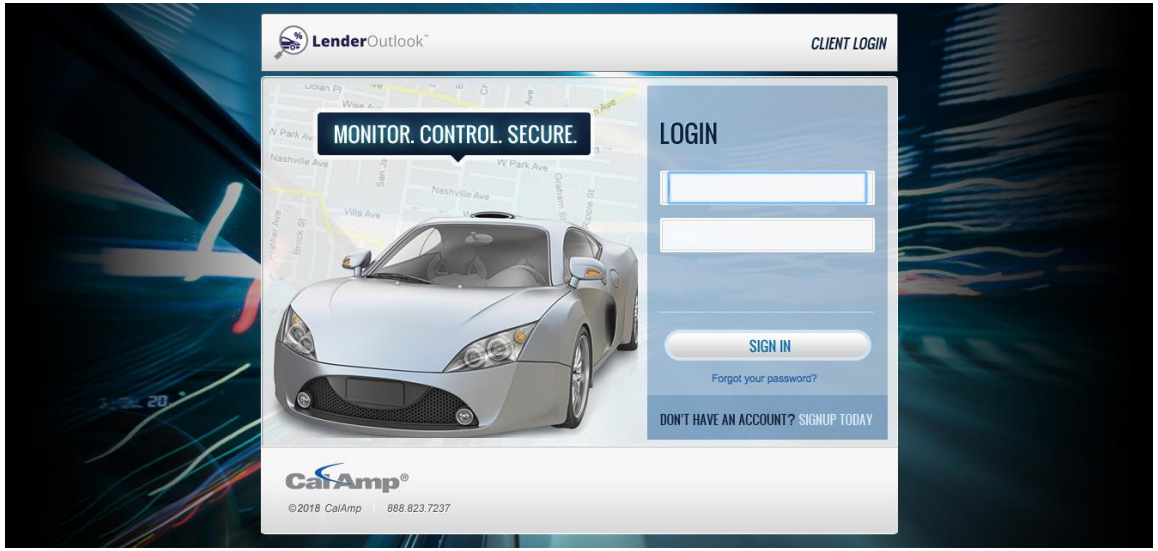
M3: Insecure authentication/Authorization

- Ví dụ: vượt qua bước xác thực 2 yếu tố (2-FA) trên ứng dụng Grab Android.
- Thực hiện bruteforce 4 kí tự để bypass 2FA. Nguyên nhân là do không có sự giới hạn số lần gửi mã 4 kí tự từ ứng dụng lên máy chủ.
- Thời điểm bị lỗi: 2017

https://hackerone.com/reports/202425

M3: Insecure authentication/Authorization

- Phân quyền có sai sót hoặc không phân quyền
- Có khả năng bị khai thác leo thang đặc quyền
- VD: Ứng dụng **Viper smart start** (theo dõi quản lý xe ô tô) bị lỗ hổng phân quyền. Sau khi đăng nhập vào server, có thể thay đổi định danh (ID) của xe, xem thông tin của các xe khác…



https://medium.com/@evstykas/remote-smart-car-hacking-with-just-a-phone-2fe7ca682162

M4: Insufficient Input/Output Validation

- Các lỗ hổng phổ biến trong API do dữ liệu đầu vào bị sửa đổi hoặc không đúng định dạng dẫn tới việc chèn các câu lệnh khai thác, mã thực thi.
- Tấn công phổ biến: SQL Injection, Command Injection, XSS,...

M5: Insecure communication

Kết nối dữ liệu không được mã hóa và xác thực.

- poor handshaking/weak negotiation (f. ex. lack of certificate pinning)
- SSL version không thích hợp
- Giao tiếp với dữ liệu dạng cleartext của các thông tin nhạy cảm
- Sử dụng HTTP thay vì HTTPS

M5: Insecure communication

Ví dụ: Ứng dụng Misafe trên đồng hồ thông minh

Kẻ tấn công có thể:

- Lấy thông tin GPS theo thời gian thực trên đồng hồ trẻ em (smartwatch)
- Gọi tới đứa trẻ thông qua đồng hồ
- Tạo một cuộc gọi audio một chiều, theo dõi trẻ.
- Gửi tin nhắn thoại tới đứa trẻ
- Lấy hình ảnh và thông tin thể trạng của trẻ.

https://nakedsecurity.sophos.com/2018/11/16/hacking-misafes-smartwatches-for-kids-is-childs-play

M6: Inadequate Privacy Controls

Thông tin và dữ liệu riêng tư của người dùng có khả năng không được kiểm soát đầy đủ. Bao gồm các dữ liệu các nhân và việc tuân thủ quyền riêng tư như GDPR, CCPA.

Dữ liệu có thể bị rò rỉ qua nhiều cách thức khác nhau, có thể do ứng dụng lưu trữ nhiều dữ liệu dư thừa của người dùng mà không dùng đến hoặc việc bị kẻ tấn công lợi dụng để lấy thông tin người dùng.

https://gdpr-info.eu/
https://oag.ca.gov/privacy/ccpa

M7: Insufficient Binary Protections

- Buffer overflow
- Format string
- Ví dụ: lỗ hổng 0-Day (CVE-2019-3568) – buffer overflow trên WhatApp VOIP stack được tìm ra trong **WhatsApp**, mục tiêu là cài đặt âm thầm spyware (Pegasus spyware) vào thiết bị.

https://thehackernews.com/2019/05/hack-whatsapp-vulnerability.html

M7: Insufficient Binary Protections

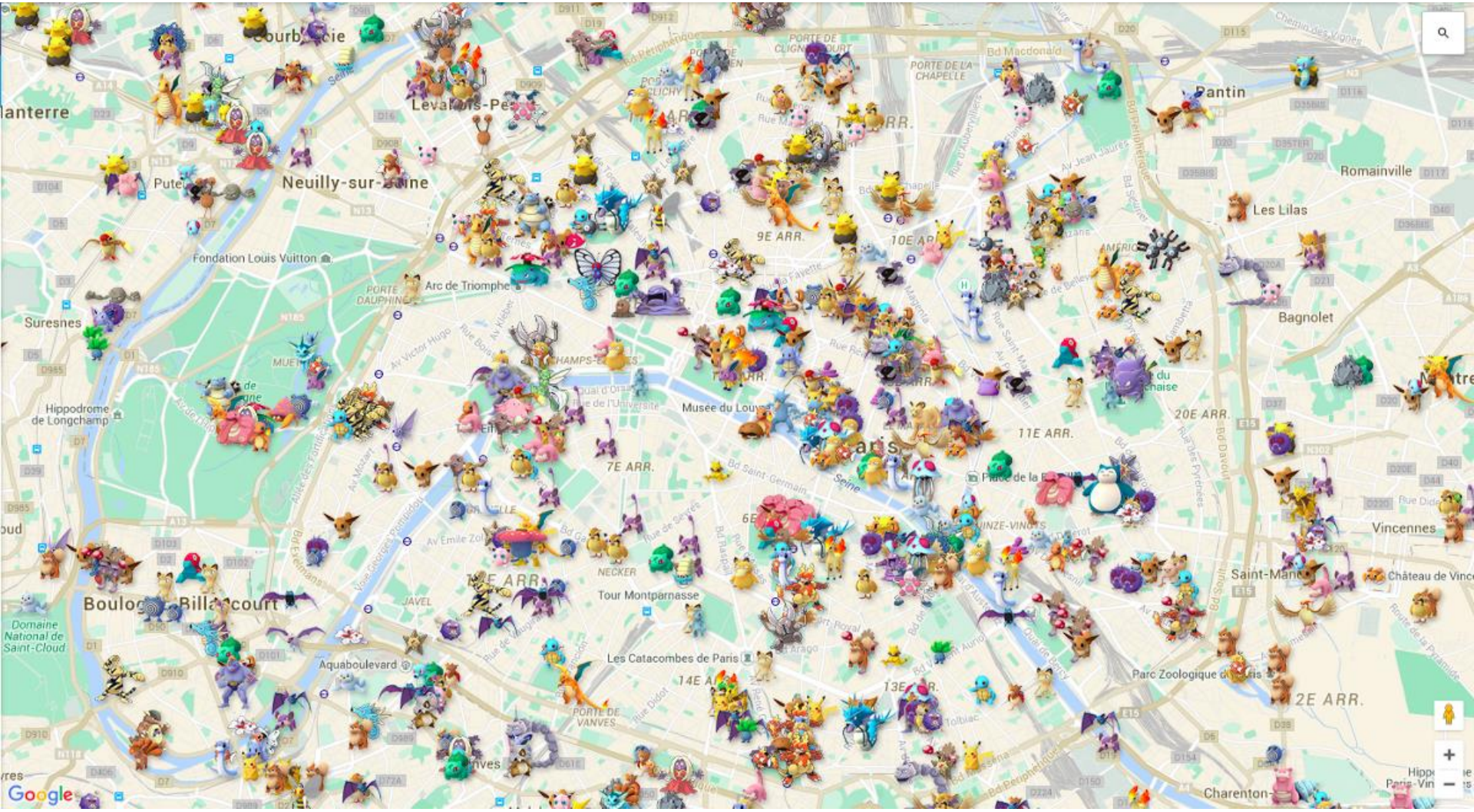
- Binary patching
- Local resource modification
- Method hooking and swizzling
- Dynamic memory modification

M7: Insufficient Binary Protections

- **Ví dụ:** Ứng dụng **Pokemon GO** bị dịch ngược, thêm vào các vị trí giả mạo nhằm tìm kiếm các Pokemon hiếm và làm cho trứng nở nhanh hơn.

https://nordicapis.com/how-pokemon-go-fans-hacked-em-all-and-how-to-prevent-similar-reverse-engineering

M7: Insufficient Binary Protections



M7: Insufficient Binary Protections

- Phân tích binary để xác định:
  - Source code
  - Thư viện (library)
  - Thuật toán và các tham số liên quan

M8: Security Misconfiguration

- Cấu hình sai về bảo mật dễ xảy ra do việc kiểm soát tính năng, phân quyền ở một số phần mềm là quá phức tạp, dẫn tới việc bỏ sót hoặc cấp quyền không cần thiết.
- Áp dụng nguyên tắc “đặc quyền tối thiểu” trong phân quyền dữ liệu.

M9: Insecure data storage

- Lưu trữ dữ liệu không an toàn
- Rò rỉ dữ liệu vô ý

Bao gồm:

- wrong keychain accessibility option
- insufficient file data protection
- access to privacy resources when using this data incorrectly.

M9: Insecure data storage

- ✓ Rò rỉ thông tin GPS trong ứng dụng hẹn hò Tinder:
- ✓Tham khảo: https://www.youtube.com/watch?v=3E2DwdS_PvQ

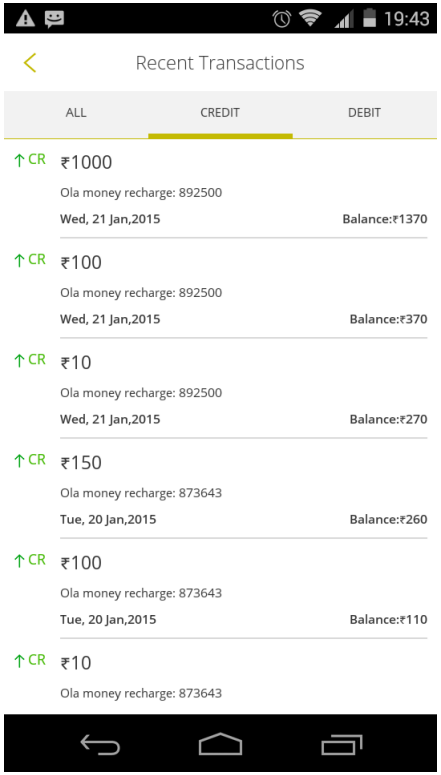
M10: Insufficient cryptography

- Sử dụng một thuật toán mã hóa đã lỗi thời/dễ bẻ khóa; hoặc tự viết một thuật toán mã hóa có lỗ hổng
- VD: Lỗ hổng trên ứng dụng Ola (dịch vụ thuê xe như Grab) tại Ấn Độ do Appknox công bố → ảnh hưởng đến ví tiền trong ứng dụng.
  - Sử dụng thuật toán mã hoá không mạnh
  - Appknox phát hiện ra Key dùng để mã hóa là: PRODKEYPRODKEY12

https://www.appknox.com/blog/major-bug-in-ola-app-can-make-you-either-rich-or-poor

M10: Insufficient cryptography

- Appknox chỉ ra ứng dụng Ola có dữ liệu truyền nhận KHÔNG thông qua giao thức mã hóa SSL
- Thực hiện nạp lại ví Ola mà không thông qua cổng thanh toán.




```
[redacted]$ curl -H 'api-key:@ndro1d' -H 'enable_auto:true'  'http://mapi.olacabs.com/v1_1/ola_money/user_information_for_transaction?amount=1000&user_id=[redacted]&enable_auto=true&enable_marketing=true'
{"status":"SUCCESS","request_type":"USER_INFORMATION_FOR_TRANSACTION","order_id":892500,"email":"[redacted]s@gmail.com","coupon_applied":false,"return_url":"http://mapi.oladev.in/v1_1/ola_money/callback","wp_return_url":"/v1_1/ola_money/callback_wp"}
esponseCode=100&responseDescription=The+transaction+was+completed+successfully.&checksum=[redacted]&amount=1000&paymentMethod=[redacted]&cardhashid=[redacted]' http://mapi.olacabs.com/v1_1/ola_money/callback
<?xml version="1.0" encoding="UTF-8"?>
<paymentResponse>
  <orderid>892500</orderid>
  <amount type="float">10.0</amount>
  <status type="integer">0</status>
  <statusMsg>The transaction was completed successfully.</statusMsg>
</paymentResponse>
```

Fuzzy testing

Fuzzy testing

```
reverse : List a → List a
reverse list =
    List.reverse list
```

- Hàm reverse có chức năng nhận giá trị đầu vào là một danh sách → cho kết quả trả về là một danh sách bị đảo ngược.
- Làm sao để kiểm tra hàm này hoạt động đúng như mong muốn?
  - **??** In ra giá trị
  - **??** Fuzzy testing

Fuzzy testing

```
doubleReverseTest : Test
doubleReverseTest =
    fuzz (Fuzz.list Fuzz.int) "A list reversed twice should be the original list" ◁
        \list →
            list
                ▷ reverse
                ▷ reverse
                ▷ Expect.equal list
```

- Fuzzy testing (tự phát sinh dữ liệu test)
- VD: Viết fuzzy test trong **elm-packages**

“Fuzz testing allows you to focus more on expected behavior and less on coming up with and maintaining test data.”

Fuzzy testing

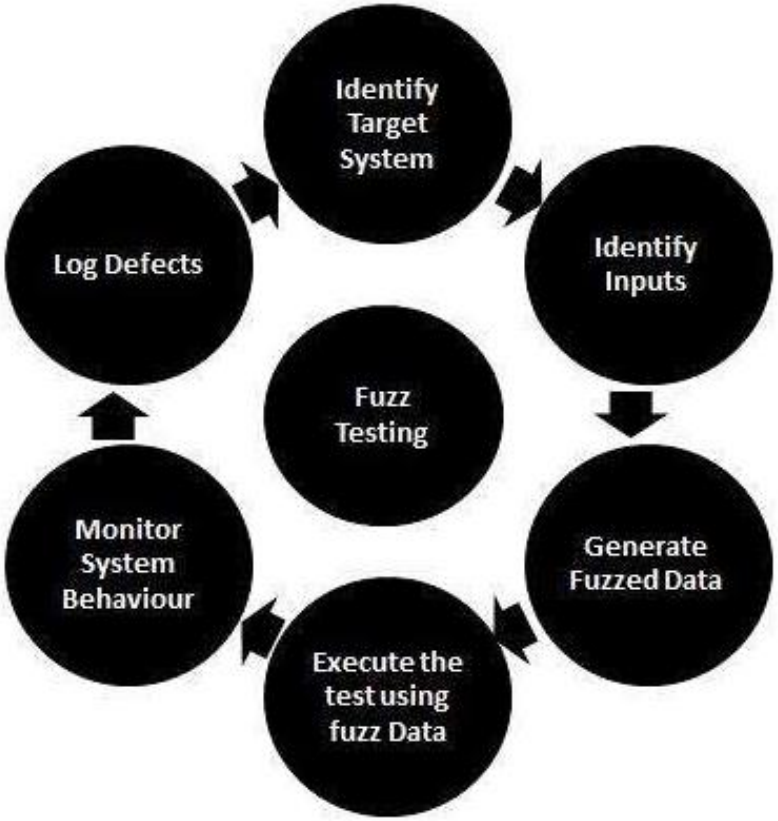
- Fuzz Testing là một loại kiểm thử trong đó các kỹ thuật **kiểm tra tự động** hoặc **bán tự động** được sử dụng để phát hiện lỗi mã hóa và lỗ hổng bảo mật trong phần mềm, hệ điều hành hoặc mạng bằng cách nhập dữ liệu không hợp lệ hoặc ngẫu nhiên (gọi là FUZZ) vào hệ thống
- Là một kỹ thuật kiểm thử phần mềm, và là một loại Security Testing
- VD: kiểm tra Fuzz có thể bao gồm đầu vào của các loại số nguyên, chuỗi ký tự, các biến với các kiểu khác nhau, nếu không được nhập chính xác, có thể khiến ứng dụng phần mềm bị treo hoặc crash

Fuzzy testing

- Fuzz testing là một trong những kỹ thuật thử nghiệm hộp đen
- Fuzzing là một trong những phương pháp phổ biến nhất được hacker sử dụng để tìm lỗ hổng của hệ thống

Chiến lược kiểm tra

- Bước 1: Xác định hệ thống đích
- Bước 2: Xác định đầu vào
- Bước 3: Tạo dữ liệu mờ
- Bước 4: Thực hiện kiểm tra bằng cách sử dụng dữ liệu mờ
- Bước 5: Theo dõi hành vi hệ thống
- Bước 6: Tổng hợp lỗi



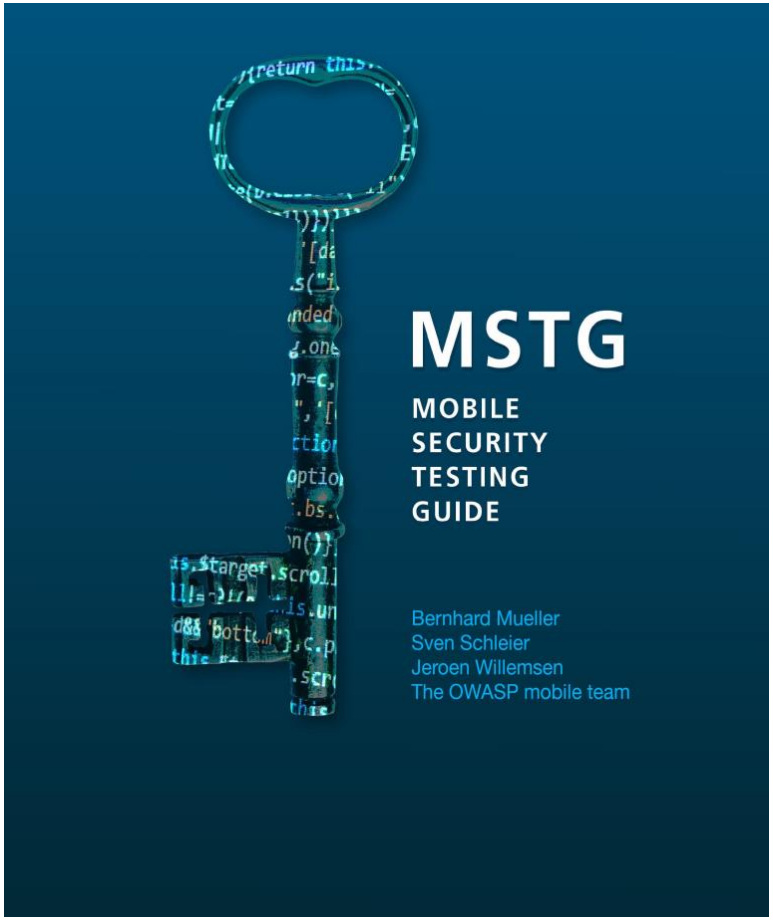
Ưu điểm và nhược điểm

- Ưu điểm
  - Cải thiện kiểm thử bảo mật cho phần mềm/hệ thống
  - Có thể tìm thấy những lỗ hổng nghiêm trọng như crash, rò rỉ bộ nhớ, không xử lý ngoại lệ,…
  - Tìm thấy các bug mà ít nhận được sự chú ý của nhân viên kiểm thử
  - Những lỗi không được tìm thấy khi kiểm thử bị hạn chế về thời gian và nguồn lực thì cũng được kiểm thử fuzzing tìm ra
- Nhược điểm
  - Không thể đưa ra báo cáo tổng thể về bảo mật nếu chỉ áp dụng mỗi phương pháp Fuzz testing
  - Ít hiệu quả trong các trường hợp xử lý các mối đe dọa bảo mật không gây ra sự cố chương trình (như worm, trojan, virus,…)
  - Đòi hỏi nhiều thời gian
  - Thiết lập điều kiện biến đầu vào ngẫu nhiên là một việc khó, đòi hỏi nhiều công sức/ thuật toán

Công cụ kiểm tra Fuzz

- Peach Fuzzera
- Proxy Spike
- Webscarab
- Burp
- OWASP WSFuzzer
- AppScan

Tham khảo



https://github.com/OWASP/owasp-mstg/

Tham khảo

- https://mobile-security.gitbook.io/mobile-security-testing-guide/ (OWASP Mobile Security Testing Guide)
- https://mobile-security.gitbook.io/mobile-security-testing-guide/android-testing-guide/0x05a-platform-overview **(Android Testing Guide)**
- https://mobile-security.gitbook.io/mobile-security-testing-guide/ios-testing-guide/0x06a-platform-overview (**IOS TESTING GUIDE**)
- Android Anti-Reversing Defenses (https://mobile-security.gitbook.io/mobile-security-testing-guide/android-testing-guide/0x05j-testing-resiliency-against-reverse-engineering)
- iOS Anti-Reversing Defenses (https://mobile-security.gitbook.io/mobile-security-testing-guide/ios-testing-guide/0x06j-testing-resiliency-against-reverse-engineering)
- Fuzzy Testing: https://medium.com/@ckoster22/an-introduction-to-fuzz-testing-a760e753191d

Bảo mật web và ứng dụng

Trường ĐH CNTT TP. HCM