TRƯỜNG ĐẠI HỌC CÔNG NGHỆ SÀI GÒN

**KHOA CÔNG NGHỆ THÔNG TIN**

---oOo---

**LUẬN VĂN TỐT NGHIỆP**

*Tên đề tài:*

**NGHIÊN CỨU VIRUS TRÊN HỆ ĐIỀU HÀNH ANDROID VÀ TRIỂN KHAI ỨNG DỤNG**

Người hướng dẫn: **ThS. Phạm Liệu**

Sinh viên thực hiện: **Phan Hoàng Lộc**

TP HỒ CHÍ MINH – NĂM 2021

BỘ GIÁO DỤC VÀ ĐÀO TẠO CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ SÀI GÒN** Độc lập – Tự do – Hạnh phúc

TP. HỒ CHÍ MINH - - - - - - - - - -

----------

KHOA: ***Công Nghệ Thông Tin***

**NHIỆM VỤ BÀI THI TỐT NGHIỆP**

*(Sinh viên phải dán tờ này vào trang thứ nhất của Bản thuyết minh)*

HỌ VÀ TÊN: 1/ PHAN HOÀNG LỘC MSSV: DH51703617 LỚP: D17\_TH09

NGÀNH: **Tin học**

1. ĐẦU ĐỀ BÀI THI:

Nghiên cứu virus trên hệ điều hành Android

1. NHIỆM VỤ YÊU CẦU VỀ NỘI DUNG VÀ SỐ LIỆU BAN ĐẦU:

* Tìm Hiểu Về Kiến Trúc Của Hệ Điều Hành Android.
* Tìm Hiểu Về Máy Học (Các Thuật Toán như: SVM, Cây Quyết Định, Học Có Giám Sát, Không Giám Sát, Học Tăng Cường Và Học Bán Giám Sát).
* Nghiên Cứu Virus Trên Hệ Điều Hành Android.
* Phân Loại Các Loại Mã Độc, Phân Tích Mã Độc.
* Các Phương Pháp Lây Nhiễm Mã Độc Trên Hệ Điều Hành Android.

1. CÁC HỒ SƠ VÀ TÀI LIỆU CUNG CẤP BAN ĐẦU:

Hướng dẫn mỗi tuần

1. NGÀY GIAO NHIỆM VỤ BÀI THI: 10/05/2021
2. NGÀY HOÀN THÀNH NHIỆM VỤ: 16/08/2021
3. HỌ TÊN NGƯỜI HƯỚNG DẪN: PHẦN HƯỚNG DẪN
   1. ThS. Phạm Liệu Toàn phần

**Nội dung và yêu cầu của Bài thi Tốt nghiệp đã được thông qua Khoa.**

*Tp. Hồ Chí Minh, ngày … tháng … năm 2021*

**TRƯỞNG KHOA NGƯỜI HƯỚNG DẪN CHÍNH**

*(Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên)*

ThS. Phạm Liệu

**LỜI CẢM ƠN**

Để hoàn thành đề tài luận văn tốt nghiệp “Nghiên cứu virus trên hệ điều hành Android”, bên cạnh sự nỗ lực của bản thân đã vận dụng những kiến thức tiếp thu được, tìm hiểu cũng như học hỏi cũng như thu thập thông tin số liệu liên quan đến đề tài, tôi luôn nhận được sự giúp đỡ tận tình từ các thầy cô, bạn bè.

Với tình cảm chân thành, em xin bày tỏ lòng biết ơn đến các thầy cô khoa Công Nghệ Thông Tin trường Đại học Công Nghệ Sài Gòn, đặc biệt là ThS. Phạm Liệu đã tận tình giúp đỡ em hoàn thành đề tài luận văn này. Và em cũng cảm ơn các thầy cô giảng viên trong nhà trường đã truyền đạt cho em rất nhiều kiến thức bổ ích để thực hiện khóa luận và cũng như có được hành trang vững chắc cho sự nghiệp trong tương lai.

Do giới hạn kiến thức và khả năng lý luận của bản thân còn nhiều thiếu sót và hạn chế, kính mong sự chỉ dẫn và đóng góp của các thầy cô giáo để đề tài luận văn của em hoàn thiện hơn.

Và lời cuối cùng em kính chúc thầy cô có nhiều sức khỏe, hạnh phúc thành công trên con đường sự nghiệp giảng dạy.

Tp. HCM, ngày 16 tháng 08 năm 2021

**Sinh viên thực hiện:**

Phan Hoàng Lộc

**MỤC LỤC**

[Chương 1. GIỚI THIỆU 8](#_Toc78808373)

[1.1 GIỚI THIỆU VỀ HỆ ĐIỀU HÀNH ANDROID 8](#_Toc78808374)

[1.1.1 Lịch sử hình thành và phát triển 8](#_Toc78808375)

[1.1.2 Kiến trúc hệ điều hành Android 11](#_Toc78808376)

[1.2 BẢO MẬT TRONG THIẾT BỊ ANDROID 13](#_Toc78808377)

[1.3 QUÁ TRÌNH ĐÓNG GÓI ỨNG DỤNG ANDROID 15](#_Toc78808378)

[1.4 CẤU TRÚC FILE APK 16](#_Toc78808379)

[1.5 PHÂN VÙNG CỦA HỆ ĐIỀU HÀNH ANDROID 17](#_Toc78808380)

[1.6 QUÁ TRÌNH KHI ỨNG DỤNG ANDROID KHỞI CHẠY 19](#_Toc78808381)

[Chương 2. TỔNG QUAN VỀ MÃ ĐỘC TRÊN ANDROID 21](#_Toc78808382)

[2.1 GIỚI THIỆU VỀ MÃ ĐỘC 21](#_Toc78808383)

[2.2 PHÂN LOẠI MÃ ĐỘC 24](#_Toc78808384)

[2.2.1 Một số phương pháp lây nhiễm mã độc trên hệ điều hành android 24](#_Toc78808385)

[2.2.2 Các loại mã độc phổ biến trên hệ điều hành Android 24](#_Toc78808386)

[2.2.3 Giới thiệu phương pháp tấn công backdoor sử dụng Firebase 25](#_Toc78808387)

[2.2.4 Giới thiệu mã độc Trojan Triada 27](#_Toc78808388)

[2.2.5 Giới thiệu về mã độc Trojan Alien 28](#_Toc78808389)

[2.3 PHƯƠNG PHÁP PHÂN TÍCH MÃ ĐỘC 29](#_Toc78808390)

[2.3.1 Mục tiêu phân tích mã độc 29](#_Toc78808391)

[2.3.2 Phương pháp phân tích tĩnh 29](#_Toc78808392)

[2.3.3 Phương pháp phân tích động 30](#_Toc78808393)

[Chương 3. TỔNG QUAN VỀ MÁY HỌC 32](#_Toc78808394)

[3.1 GIỚI THIỆU VỀ MÁY HỌC 32](#_Toc78808395)

[3.2 PHÂN LOẠI CÁC THUẬT TOÁN TRONG MÁY HỌC 32](#_Toc78808396)

[3.2.1 Học tăng cường (reinforcement learning) 32](#_Toc78808397)

[3.2.2 Học có giám sát (supervised learning) 33](#_Toc78808398)

[3.2.3 Học không giám sát (unsupervised learning) 33](#_Toc78808399)

[3.2.4 Học bán giám sát (semi-supervised learning) 33](#_Toc78808400)

[3.3 GIỚI THIỆU THUẬT TOÁN HỌC MÁY HIỆU QUẢ 34](#_Toc78808401)

[3.3.1 Thuật toán cây quyết định 34](#_Toc78808402)

[3.3.2 Thuật toán SVM 35](#_Toc78808403)

[3.4 QUÁ TRÌNH XÂY DỰNG MÔ HÌNH MÁY HỌC 39](#_Toc78808404)

[3.4.1 Tổng quát 39](#_Toc78808405)

[3.4.2 Xác định bài toán 40](#_Toc78808406)

[3.4.3 Chuẩn bị dữ liệu 41](#_Toc78808407)

[3.4.4 Xây dựng mô hình 41](#_Toc78808408)

[3.4.5 Triển khai và bảo trì 42](#_Toc78808409)

[Chương 4. TRIỂN KHAI ỨNG DỤNG 44](#_Toc78808410)

[4.1 THU THẬP DỮ LIỆU 44](#_Toc78808411)

[4.2 ĐÁNH GIÁ MÔ HÌNH 44](#_Toc78808412)

[4.3 CÔNG NGHỆ SỬ DỤNG 45](#_Toc78808413)

[4.3.1 Thư viện ML.NET 45](#_Toc78808414)

[4.3.2 WPF 45](#_Toc78808415)

[4.3.3 Bootstrap 46](#_Toc78808416)

[4.3.4 Jquery 47](#_Toc78808417)

[4.3.5 Entity Framework 47](#_Toc78808418)

[4.3.6 ASP.NET Framework 47](#_Toc78808419)

[4.4 HƯỚNG DẪN SỬ DỤNG ỨNG DỤNG 48](#_Toc78808420)

[4.4.1 Triển khai ứng dụng trên nền tảng Web 48](#_Toc78808421)

[4.4.2 Triển khai trên nền tảng Window 52](#_Toc78808422)

[Chương 5. TỔNG KẾT 55](#_Toc78808423)

[5.1 KẾT QUẢ ĐẠT ĐƯỢC 55](#_Toc78808424)

[5.2 HƯỚNG PHÁT TRIỂN 55](#_Toc78808425)

[TÀI LIỆU THAM KHẢO 57](#_Toc78808426)

**MỤC LỤC CÁC HÌNH VẼ**

[Hình 1-1. Kiến trúc Android 11](#_Toc77669988)

[Hình 1-2. Các lớp bảo mật Android 13](#_Toc77669989)

[Hình 1-3. Quá trình JVM biên dịch 14](#_Toc77669990)

[Hình 1-4. Quá trình Dalvik executable biên dịch 15](#_Toc77669991)

[Hình 1-5. Quá trình cuối tạo file apk 15](#_Toc77669992)

[Hình 1-6. Cấu trúc file apk 16](#_Toc77669993)

[Hình 1-7. Phân vùng trên thiết bị Android 17](#_Toc77669994)

[Hình 1-8. Quy trình khởi động ứng dụng Android 18](#_Toc77669995)

[Hình 2-1. Sơ đồ các cuộc tấn công vào người dùng 2019-2020 21](#_Toc77669996)

[Hình 2-2. Sơ đồ các cuộc tấn công quảng cáo vào người dùng 2019-2020 22](#_Toc77669997)

[Hình 2-3. Các gói cài đặt mã độc trên thiết bị di động 2019-2020 22](#_Toc77669998)

[Hình 2-4. Quy trình tấn công sử dụng firebase 25](#_Toc77669999)

[Hình 2-5. Quy trình tấn công Triada 27](#_Toc77670000)

[Hình 2-6. Mã binary file sử dụng công cụ HxD 30](#_Toc77670001)

[Hình 3-1. Cây quyết định cho ví dụ 34](#_Toc77670002)

[Hình 3-2. Công thức tính khoảng cách từ một điểm đến đường thẳng 36](#_Toc77670003)

[Hình 3-3. Công thức tính khoảng cách từ một điểm đến mặt phẳng 36](#_Toc77670004)

[Hình 3-4. Công thức tính khoảng cách từ một điểm đến siêu mặt phẳng 37](#_Toc77670005)

[Hình 3-5. Các mặt phân chia hai lớp 37](#_Toc77670006)

[Hình 3-6. Phân chia các lớp 38](#_Toc77670007)

[Hình 3-7. Phân chia hai lớp 39](#_Toc77670008)

[Hình 3-8. Công thức tính khoảng cách tổng quát 39](#_Toc77670009)

[Hình 3-9. Mặt phẳng phân chia lớp 40](#_Toc77670010)

[Hình 3-10. Quá trình xây dựng mô hình máy học 41](#_Toc77670011)

[Hình 4-1. Trang chủ ứng dụng web 49](#_Toc77670012)

[Hình 4-2. Kết quả sau khi phân tích file apk 50](#_Toc77670013)

[Hình 4-3. Kết quả chi tiết file apk 51](#_Toc77670014)

[Hình 4-4. Kết quả dự đoán 52](#_Toc77670015)

[Hình 4-5. Trang chủ màn hình 53](#_Toc77670016)

[Hình 4-6. Kết quả sau khi phân tích 53](#_Toc77670017)

[Hình 4-7. Kết quả chi tiết 54](#_Toc77670018)

# 

# GIỚI THIỆU

1. **GIỚI THIỆU VỀ HỆ ĐIỀU HÀNH ANDROID**
2. **Lịch sử hình thành và phát triển**

Android là hệ điều hành dựa trên nền tảng Linux được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu hệ điều hành android được phát triển bởi công ty Android.Inc, sau này được được Google mua lại vào ngày 27 tháng 8 năm 2005, biến nó thành một bộ phận trực thuộc Google. Hiện nay, nó đã trở thành hệ điều hành được sử dụng nhiều nhất trên di động so với các đối thủ khác như: Symbian, Blackberry, Palm OS, webOS và WindowPhone.

Lịch sử phát triển hệ điều hành Android bắt đầu với bản Android beta vào tháng 11 năm 2007 được công bố bởi Google và OHA (là một hiệp hội gồm 84 công ty để phát triển những tiêu chuẩn cho các thiết bị di động như: HTC, Sony, Motorola, Samsung, LG, T-Mobile,…).

Phiên bản thương mại đầu tiên là Android 1.0 được phát hành vào ngày 23 tháng 9 năm 2008, thiết bị thương mại đầu tiên có sẵn Android là HTC Dream. Các tính năng của Android 1.0 như: Ứng dụng Android Market (CH Play bây giờ), Web browser hiển thị đầy đủ các trang web HTML và XHTML, hỗ trợ máy ảnh, sms, mms,…

Vào ngày 9 tháng 2 năm 2009 phiên bản Android 1.1 được phát hành, là phiên bản vá lỗi của Android 1.0 và cập nhật thêm một số tính năng như: Màn hình thời gian chờ dài hơn trong cuộc gọi khi sử dụng loa, cùng với khả năng hiển thị và ẩn bàn phím quay số.

Ngày 30 tháng 4 năm 2009 bản cập nhật Android 1.5 (Cupcake) đã được phát hành. Bản cập nhật bao gồm một số tính năng mới và sửa đổi giao diện người dùng: hỗ trợ bàn phím ảo thứ ba với đoán văn bản và từ điển từ các tùy chỉnh, hỗ trợ widget, quay video và phát định dạng MPEG-4, tính năng sao chép và dán trong trình duyệt web, ngày giờ hiện cụ thể cho sự kiện trong nhật kí cuộc gọi, tự động xoay,…

Ngày 15 tháng 9 năm 2009 phiên bản Android 1.6 (Donut) được phát hành dựa trên Linux kernel 2.6.29. Bao gồm trong bản cập nhật và tính năng mới: Tìm kiếm bằng giọng nói, cho phép truy cập nhanh máy ảnh, cho phép người dùng chọn nhiều ảnh để xóa, hỗ trợ cho màn hình độ phân giải WVGA,…

Ngày 26 tháng 10 năm 2009 phiên bản Android 2.0 (Eclair) được phát hành dựa trên Linux kernel 2.6.29 với các tính năng mới như: mở rộng đồng bộ hóa tài khoản, cho phép người dùng có thể thêm nhiều tài khoản trên một thiết bị cho đồng bộ của Gmail và danh bạ, hỗ trợ Bluetooth 2.1, cải thiện Google Maps 3.1.2, lớp MotionEvent tăng cường theo dõi cảm ứng đa điểm, hỗ trợ UI nhấp 2 lần để Zoom và hỗ trợ HTML 5.

Ngày 3 tháng 12 năm 2009 phiên bản 2.0.1 được phát hành mục đích thay đổi nhỏ API, sửa lỗi và framework. Khoảng một tháng sau ngày 12 tháng 1 năm 2010 tiếp tục phát hành bản vá lỗi phiên bản Android 2.1 thay đổi nhỏ API và sửa lỗi.

Ngày 20 tháng 5 năm 2010 phiên bản Android 2.2(Froyo) được phát hành dựa trên Linux kernel 2.6.32 cùng với các tính năng: tối ưu hóa hiệu suất, tích hợp V8 JavaScript Engine của Chrome vào ứng dụng trình duyệt, chức năng USB tethering và Wi-Fi hotspot, tùy chọn vô hiệu hóa truy cập dữ liệu qua mạng di động, hỗ trợ màn hình PPI, hỗ trợ password chữ và số, hỗ trợ cài đặt ứng dụng từ thẻ nhớ mở rộng.

Ngày 6 tháng 12 năm 2010 phát hành phiên bản Android 2.3 (Gingerbread) được phát hành dựa trên Linux kernel 2.6.35 với các tính năng: hỗ trợ màn hình có kích thước lớn và độ phân giải WXGA, hỗ trợ cho giao tiếp gần NFC, hỗ trợ cho phát video WebM/VP8 và mã hóa âm thanh AAC.

Tháng 12 năm 2010 phiên bản Android 2.3.1 và tháng 1 năm 2011 phiên bản Android 2.3.2 cải tiến và sửa lỗi cho Google Nexus S. Ngày 9 tháng 2 năm 2011 phiên bản Android 2.3.3 một số cải tiến và chỉnh sửa API.

Ngày 28 tháng 4 năm 2011 phiên bản Android 2.3.4 được phát hành cùng với tính năng: hỗ trợ trò chuyện giọng nói hoặc sử dụng Google Talk, hỗ trợ thư viện phụ kiện, chuyển mã hóa mặc định cho SSL từ AES256-SHA thành RC4-MD5.

Ngày 25 tháng 7 năm 2011 phiên bản Android 2.3.5 được phát hành cùng với các tính năng như: cải thiện hiệu suất mạng cho Nexus S 4G, sửa lỗi Bluetooth trên Samsung Galaxy S, cải thiện ứng dụng Gmail, cải thiện hiệu suất pin.

Ngày 2 tháng 9 năm 2011 phiên bản Android 2.3.6 được phát hành cùng với các tính năng như: sửa lỗi tìm kiếm bằng giọng nói. Ngày 21 tháng 9 năm 2011 phiên bản Android 2.3.7 phát hành mục đích hỗ trợ cho Google Wallet cho Nexus S 4G.

Ngày 22 tháng 2 năm 2011 phiên bản Android 3.0 (Honeycomb) được giới thiệu đây là phiên bản Google tập trung cho phát triển máy tính bảng và các thiết bị di động có màn hình lớn. Các tính năng được phát triển như: thêm nút điều hướng có sẵn ở dưới màn hình, cho phép sử dụng trình duyệt ẩn danh, tăng tốc phần cứng, hỗ trợ cho đa nhân xử lý, khả năng mã hóa tất cả dữ liệu của người dùng,…

Ngày 10 tháng 5 năm 2011 phiên bản Android 3.1 được phát hành cùng với các tính năng như: cải tiến UI, kết nối cho phụ kiện USB, hỗ trợ cho chơi âm thanh FLAC, hỗ trợ cho HTTP Proxy cho mỗi kết nối đến điểm Wifi.

Ngày 19 tháng 9 năm 2011 phiên bản Android 4.0(Ice Cream Sandwich) được phát triển dựa trên Linux kernel 3.0.1, các tính năng mới: tách các widgets sang một tab riêng biệt, tích hợp chức năng chụp ảnh màn hình, cải thiện chức năng kiểm tra chính tả khi gõ, mở khóa bằng khuôn mặt, hỗ trợ định dạng WebP, quay video 1080p.

Ngày 9 tháng 7 năm 2012 phiên bản Android 4.1 được phát hành tại hội nghị Google I/O dựa trên Linux kernel 3.0.31 đây là phiên bản cải thiện chức năng và hiệu suất của giao diện người dùng.

Vào ngày 3 tháng 9 năm 2013 phiên bản Android 4.4(KitKat) được phát hành với mục đích là tập trung cải thiện hiệu suất cho các thiết bị tầm thấp tức là với những thiết bị có bộ nhớ Ram 512MB và độ phân giải hiển thị 960x540.

Ngày 25 tháng 6 năm 2014 phiên bản Android 5.0 Lollipop được phát hành với các tính năng mới: hiển thị thông báo trên màn hình khóa, áp dụng material design của Google, cải thiện mức tiêu thụ pin thông qua “Project Volta”, theo dõi mức tiêu thụ pin của ứng dụng.

Tháng 10 năm 2015 phiên bản Android 6.0 Marshmallow được phát hành với các tính năng sau: giảm các hoạt động chạy ngầm khi thiết bị không được sử dụng, hỗ trợ nhận dạng vân tay và cổng kết nối USB Type-C, khả năng chuyển dữ liệu và ứng dụng sang thẻ nhớ microSD, khi cài đặt ứng dụng không tự động cấp quyền.

Vào ngày 22 tháng 8 năm 2016 phiên bản Android 7.0 Nougat được phát hành cùng với các tính năng: chế độ hiển thị chia màn hình, cơ chế tiết kiệm năng lượng “Doze”, chế độ “Data Saver” giới hạn sử dụng dữ liệu di động ngầm.

Ngày 21 tháng 8 năm 2017 phiên bản Android 8.0 Oreo được chính thức phát hành sau 2 phiên bản alpha và beta, các tính năng mới: hỗ trợ công nghệ NAN (neighborhood aware networking) cho wifi, hỗ trợ chế độ “picture in picture”,…

Ngày 7 tháng 3 năm 2018 phiên bản Android 9.0 Pie được Google công bố với các tính năng mới: trình tiết kiệm pin, tối ưu chế độ “dark mode”, chế độ “Khóa” mới vô hiệu hóa xác thực sinh trắc học, thuật toán nén thích ứng AptX cho độ trễ tốt hơn và âm thanh chất lượng hơn thông qua Bluetooth,…

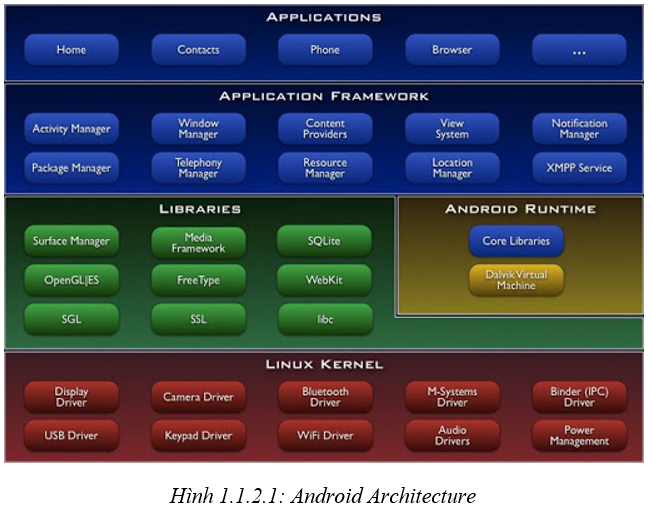
Ngày 3 tháng 9 năm 2019 phiên bản Android 10 được phát hành. Android 10 giới thiệu hệ thống cử chỉ hoàn toàn mới, bổ sung thêm thanh cài đặt, hỗ trợ giao thức mã hóa WPA3.

Ngày 8 tháng 9 năm 2020 phiên bản Android 11 được phát hành với các tính năng cho điện thoại thông minh có khả năng gập, hỗ trợ 5G, hỗ trợ xác thực cuộc gọi STIR/SHAKEN.

Hệ điều hành Android đã có một hành trình phát triển dài và hoàn thiện. Hiện nay hệ điều hành Android đang trở nên phổ biến trên thế giới bằng tính chất lượng và đa dạng và không ngừng cải thiện. Android chắc chắn sẽ còn phát triển hơn và tiếp cận rộng rãi hơn đối với các thiết bị điện tử.

1. **Kiến trúc hệ điều hành Android**

Hệ điều hành Android là một ngăn sắp xếp các thành phần lại với nhau để cấu tạo nên một hệ điều hành, được chia thành 5 phần và 4 lớp chính.



**Hình 1-1. Kiến trúc Android**

Linux kernel là tầng giao tiếp giữa trình điều khiển của phần mềm đối với các tác vụ của phần cứng như: máy ảnh, màn hình hiển thị, loa,... Đồng thời còn xử lý các tất cả những thứ mà Linux có thể làm tốt như mạng kết nối và một chuỗi các trình điều khiển thiết bị, giúp cho việc giao tiếp với các thiết bị ngoại vi dễ dàng hơn.

Libraries đây là các thư viện dựa trên Java phục vụ cho việc phát triển Android một số thư viện chính như: Surface Manager (tạo ra các màn hình khác nhau trên màn hình điện thoại thuộc những ứng dụng khác nhau), OpenGL/ES (dành cho việc đồ họa), Media Framework(được thiết kế sử dụng cho các phương tiện như video, mp4, mp3), Free Type (được sử dụng để điều chỉnh font chữ trên màn hình hiển thị), WebKit (phục vụ cho việc xây dựng trình duyệt web), SQLite (xử lý và tương tác với cơ sở dữ liệu),…

Android Runtime phần này cung cấp một bộ phận quan trọng là Dalvik Virtual Machine là một loại Java Virtual Machine được thiết kế đặc biệt để tối ưu hóa cho Android. Dalvik VM sử dụng một số đặc trưng của Linux như quản lý bộ nhớ và đa luồng, những thứ mà đã có sẵn trong Java. Core libraries cung cấp một số thư viện chính giúp các nhà phát triển ứng dụng Android có thể viết ứng dụng Android bằng Java. Dalvik chỉ là một máy ảo khá đơn giản, song khi các thiết bị di động càng trở nên phức tạp, Google cần phải giải quyết các vấn đề hiệu năng, do dalvik là phần mềm gần như chịu trách nhiệm thực thi gần như tất cả các ứng dụng Android, Google sẽ phải tìm cách giúp máy ảo Android của mình có thể bắt kịp với các tiến độ của ngành công nghiệp phần cứng. Từ đó ART ra đời, ART thiết kế để tương thích hoàn toàn với mô hình biên dịch bytecode của Dalvik, sự thay đổi lớn nhất về thiết kế mà ART mang lại là khả năng biên dịch code theo mô hình AOT thay vì JIT như trước đây, bộ máy runtime mới của Android không còn cần phải dịch từ bytecode sang mã đúng một lần duy nhất.

Application Framework cung cấp các dịch vụ cấp độ cao hơn cho ứng dụng dưới dạng các lớp Java. Các nhà phát triển ứng dụng được phép sử dụng các dịch vụ này để phát triển ứng dụng của họ. Các dịch vụ tại tầng Application Framework: Activity manager quản lý vòng đời của các activity của ứng dụng, Content Providers cho phép ứng dụng chia sẻ dữ liệu với các ứng dụng khác, Resource Manager cung cấp quyền truy cập vào các tài nguyên (chuỗi, màu sắc, các layout giao diện người dùng,..), Notifications Manager cho phép ứng dụng hiển thị thông báo(báo thức, cuộc gọi, tin nhắn, pin, kết nối mạng,..), Package Manager theo dõi những ứng dụng nào đã cài đặt trên thiết bị, View System hiển thị giao diện người dùng(listview, textview, button), Location manager cung cấp thông tin vị trí, Telephony Manager chứa các API để xây dựng ứng dụng điện thoại(Voice, MMS, SMS,…).

Applications đây là tầng tích hợp sẵn các ứng dụng như: danh bạ, sms,… Được tích hợp sẵn trong điện thoại Android.

1. **BẢO MẬT TRONG THIẾT BỊ ANDROID**

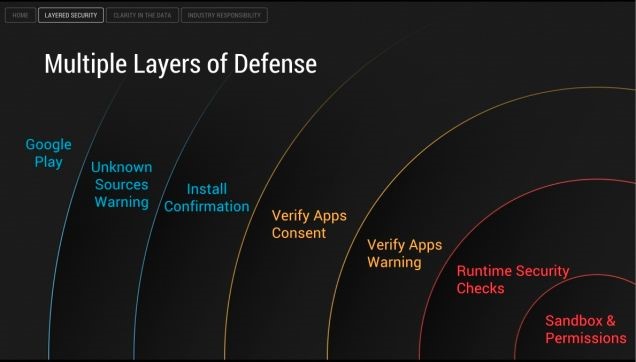
Các mô hình bảo mật được thiết kế để ngăn chặn các cuộc tấn công như Social engineering mà mục đích là cài đặt ứng dụng của bên thứ 3. Một số biện pháp quan trọng mà Google phát triển để bảo vệ người dùng Android khỏi các mối đe dọa khác nhau: Google kết hợp một số điều khiển chẳng hạn như không cho phép các ứng dụng khác nhau có avatar giống nhau hoặc cùng tên, ứng dụng có nội dung bất hợp pháp, có tính chất cá cược, những ứng dụng có nội dung kích động thù địch, ứng dụng có nội dung khiêu dâm, ứng dụng có nội dung bạo lực.

Khi một ứng dụng được đưa lên CH Play nó sẽ thực hiện phân tích động và kiểm tra chữ ký để phát hiện malware, mặc định Android không cho cài đặt các ứng dụng không rõ nguồn gốc muốn cài đặt thì phải điều chỉnh mode cho phép cài đặt ứng dụng không rõ nguồn.

Mỗi ứng dụng khi thực thi sẽ được cấp một máy ảo riêng biệt và ngăn các ứng dụng khác nhau sử dụng tài nguyên của nhau. Kết hợp các phương pháp như ASLR, ProPolice, Security-IOP với mục đích giảm thiểu rủi ro trong việc quản lý bộ nhớ và việc tấn công vào bộ nhớ.

Cập nhật các phiên bản mới nhất của Android mỗi khi Google đưa ra các bản vá hoặc các phiên bản nâng cấp.

Từ phiên bản Android 6.0 Lollipop khi bạn cài đặt một ứng dụng bất kỳ, bạn sẽ phải đồng ý với những “điều khoản” để cài đặt được ứng dụng. Sau đó người dùng có thể cấp thêm quyền cho ứng dụng đó như sử dụng microphone, camera, gps, thực hiện cuộc gọi, gửi tin nhắn,... Đối với các phiên bản từ Android 5.0 trở về trước thì các ứng dụng cài đặt một lần và tự động cấp tất cả các quyền có trong file AndroidManifest.xml.



**Hình 1-2. Các lớp bảo mật Android**

Google play là tầng bảo vệ đầu tiên. Đây là bước khi muốn triển khai một ứng dụng lên Google Store thì Google sẽ tiến hành kiểm tra thông tin nhà phát triển, đánh giá ứng dụng khi đưa lên hệ thống phân phối.

Tiếp theo là tầng cảnh báo nếu người dùng cài đặt một nguồn không chính thức Unknown Sources Warning. Đây là tầng bảo vệ mà người dùng có thể tương tác được. Theo mặc định chỉ có ứng dụng từ nguồn Google Play mới được mới có thể tiến hành cài đặt mà không có thông báo yêu cầu cho phép. Người dùng có thể hoàn toàn can thiệp vào tầng bảo vệ này bằng cách tắt chức năng Unknow source trong Setting khi đó tầng này sẽ bị vô hiệu hóa.

Install confirmation là tầng này hiện thông báo có chọn cài đặt hay không nếu vượt qua tầng thứ hai.

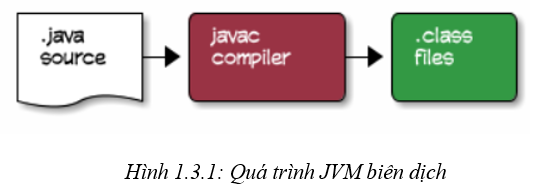
Verify Apps Consent là bước mà kiểm tra ứng dụng cài đặt với cơ sở dữ liệu của Google, nó sẽ tiến hành so sánh phân tích với kho dữ liệu đồ sộ của Google. Chức năng này được ra cảnh báo hoặc ngăn chặn việc cài đặt của các ứng dụng có nguy cơ gây hại đến hệ thống, người dùng có thể tương tác với tầng này và có thể tắt đi.

Verify App Warning khi thực hiện bước trên thì Google sẽ đưa ra cảnh báo hoặc ngăn chặn cài đặt.

Và hai tầng cuối cùng là quá trình cài đặt ứng dụng lên thiết bị và tiến hành thực thi, ứng dụng được thực thi trong chế độ (sand-box) giới hạn về tài nguyên có thể sử dụng, điều này tránh khả năng một ứng dụng vượt quyền hoặc gây ảnh hưởng đến hệ thống. Trong lớp bảo vệ này người dùng không thể tương tác can thiệp.

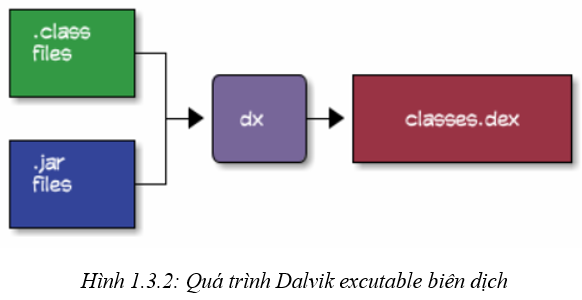
1. **QUÁ TRÌNH ĐÓNG GÓI ỨNG DỤNG ANDROID**

Có rất nhiều công cụ để phát triển Android nhưng đến nay công cụ chính thức và mạnh mẽ nhất chính là Android Studio. Và công cụ này nó giúp chúng ta đóng gói project sang file .apk để có thể cài đặt trên hệ điều hành Android.



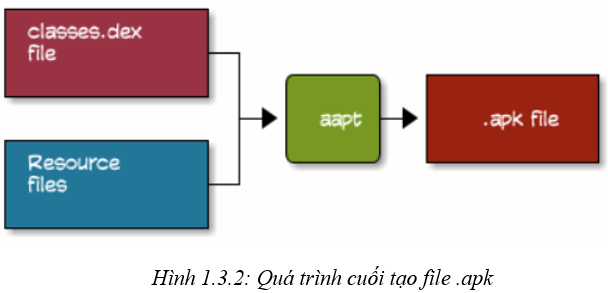
**Hình 1-3. Quá trình JVM biên dịch**

Do Android được viết bằng ngôn ngữ java nên chúng ta cần một trình biên dịch gọi là JVM (Java Virtual Machine) biên dịch các file .java thành các file .class, file .java là những đoạn code mà lập trình viên java có thể đọc được sau đó các file này sẽ được biên dịch thành mã java bytecode.



**Hình 1-4. Quá trình Dalvik executable biên dịch**

File .class chứa mã java bytecode nhưng các thiết bị Android không sử dụng định dạng mã java bytecode. Thay vào đó Android có định dạng mã bytecode riêng biệt được gọi là Dalvik. Do đó chúng ta cần thêm một trình biên dịch nữa để chuyển đổi các file .class và các thư viện file .jar thành một file duy nhất đó chính là file classes.dex, file này chứa mã dalvik bytecode.

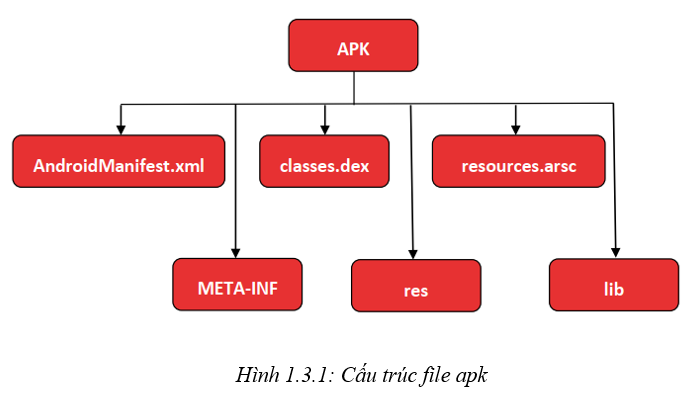


**Hình 1-5. Quá trình cuối tạo file apk**

Sau khi chúng ta có file classes.dex chúng ta cần phải nén tất cả các file còn lại như folder res, file AndroidManifest.xml cùng một số thành phần nữa sẽ được nén lại thành một file duy nhất thông qua công cụ AAPT (công cụ đóng gói tài nguyên Android).

1. **CẤU TRÚC FILE APK**

Tập tin APK(Android application package) tương tự như các gói phần mềm khác như APPX trên [Microsoft Windows](https://vi.wikipedia.org/wiki/Microsoft_Windows) hay gói Deb trên các hệ điều hành như Ubuntu. Để tạo ra tập tin APK, chương trình Android đầu tiên được biên dịch, sau đó tất cả các thành phần của nó sẽ được đóng gói lại vào một tập tin. Một tập tin APK chứa tất cả mã nguồn (ví dụ như các tập tin .dex), tài nguyên, tài sản, chứng nhận, và tập tin manifest. Tương tự như các định dạng tập tin khác, tập tin APK có thể đặt tên bất kì, với tên kết thúc với ".apk".



**Hình 1-6. Cấu trúc file apk**

File AndroidManifest.xml là file AndroidManifest.xml trong project của ứng dụng đã được biên dịch thành mã máy. File này cung cấp thông tin cơ bản của một ứng dụng cho hệ điều hành Android như: Tên java package, activity, service, broadcast, receiver, một application có nhiều activity thì activity nào sẽ được gọi đầu tiên khi run app, chứa các quyền truy cập của ứng dụng đối với hệ thống khi người dùng cho phép, phiên bản Android, các thư viện,…

Folder META-INF lưu trữ thông tin chữ ký để đảm bảo tính toàn vẹn của gói apk và tính bảo mật của hệ thống. Khi đóng gói project thành file .apk công cụ đóng gói đó sẽ tiến hành tính toán tổng kiểm tra các tệp được đóng gói và đưa kết quả tính toán vào thư mục META-INF. Điều này để đảm bảo rằng các tệp trong gói apk không thể được thay thế theo ý muốn.

Folder res lưu trữ toàn bộ các file XML (layout), file hình ảnh (PNG, JPEG) nằm trong các folder, như là -mdpi và -hdpi cho densities, -sw600dp hoặc -large cho kích cỡ màn hình, -en, -de, -pl cho ngôn ngữ. Bất kỳ file XML nào trong res/ đều đã được biên dịch lại. Do đó chúng ta không thể đọc lại được như các công cụ editor.

File classes.dex là file chứa các mã code đã được biên dịch, dưới dạng Dex (DalvikExecutable) bytecode. Có thể nhìn thấy nhiều file DEX trong 1 file APK. Từ Android 5.0 trở lên, với việc giới thiệu ART runtime, chúng được biên dịch dưới dạng OTA file.

File resources.arsc là file chứa một vài resource và file định danh (identifiers) được biên dịch và chứa lại trong file này. Bình thường nó sẽ không được nén lại và chứa trong file apk, giúp cho việc truy xuất dữ liệu nhanh hơn. Việc nén file này bằng cách thủ công có thể là giải pháp đơn giản để giảm dung lượng file.

Folder lib chứa mã biên dịch cho các kiến trúc khác nhau.

Theo thông báo của Google từ tháng 08 năm 2021, các app mới đăng lên Google Play sẽ không dùng định dạng APK quen thuộc như hiện nay nữa mà sẽ dùng định dạng mới là AAB, điểm nổi bật là chúng còn hỗ trợ Play App Signing một tính năng mới giúp app tránh bị chỉnh sửa nhúng mã độc trong quá trình download.

1. **PHÂN VÙNG CỦA HỆ ĐIỀU HÀNH ANDROID**

Hầu hết người dùng Android đang sử dụng điện thoại Android với mục đích để gọi, gửi SMS, duyệt web và các ứng dụng khác. Tuy nhiên đối với một developer để phát triển ứng dụng thì chúng ta cần nên biết về cấu trúc nội bộ Android. Android sử dụng một số phân vùng (boot, system, recovery, data,…) để tổ chức tệp và các thư mục giống trên hệ điều hành Windows.

Trên hệ điều hành Android chủ yếu có 6 phân vùng chính, ngoài ra cũng có một số phân vùng khác nhưng ta thường gặp 6 phân vùng trên bất kỳ thiết bị Android nào dưới đây.

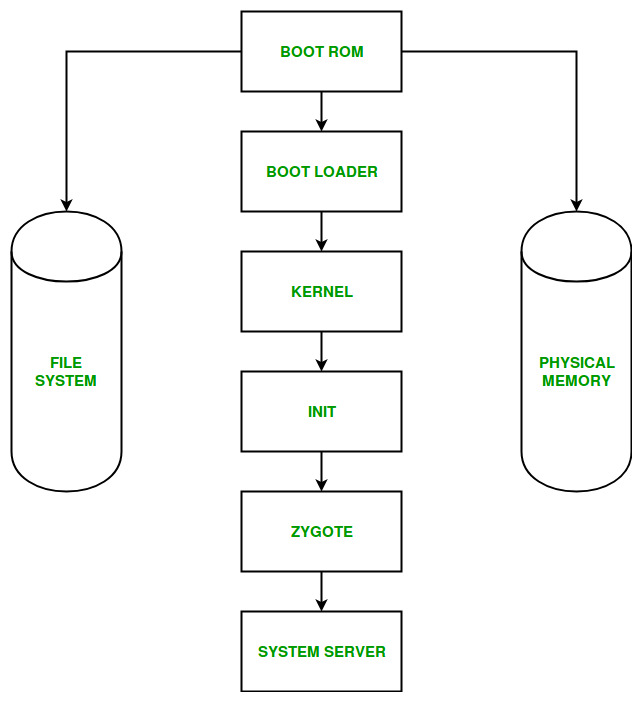


**Hình 1-7. Phân vùng trên thiết bị Android**

Chúng ta có thể thấy các phân vùng như: Boot, System, Recovery, Data, Cache, Misc. Boot là một phân vùng cho phép điện thoại khởi động, đúng như tên gọi của nó và phân vùng này chứa Android Kernel và Ramdisk, thiết bị sẽ không thể khởi động nếu nó không có Partition này. System là một phân vùng chứa toàn bộ hệ điều hành ngoại trừ Android Kernel và Ramdisk, điều này bao gồm các hệ thống được cài đặt sẵn trên thiết bị. Recovery là phân vùng khởi động thay thế cho phép thiết bị truy cập vào bảng điều khiển để tiến hành thực hiện các hoạt động khôi phục. Data đây được gọi là phân vùng userdata nơi mà chứa dữ liệu người dùng như dữ liệu danh bạ, sms, các ứng dụng android đã cài đặt. Cache là phân vùng mà Android thường xuyên truy cập và các ứng dụng, xóa bộ nhớ cache không ảnh hưởng đến dữ liệu hiện có ở đó, nó sẽ được tạo lại khi bạn tiếp tục sử dụng thiết bị. Misc là phân vùng chức các cài đặt hệ thống như Id nhà cung cấp dịch vụ hoặc khu vực, cấu hình USB nếu phân vùng này bị hỏng hoặc bị thiếu thì một số tính năng của thiết bị sẽ không hoạt động bình thường.

Ngoài các phân vùng trên ta còn phân vùng sdcard đây không phải là phân vùng trên bộ nhớ trong của thiết bị mà là thẻ SD.

1. **QUÁ TRÌNH KHI ỨNG DỤNG ANDROID KHỞI CHẠY**



**Hình 1-8. Quy trình khởi động ứng dụng Android**

Boot Rom là nơi chứa những đoạn mã cho việc khởi động ngay sau khi thiết bị được khởi động, nó là một ổ đĩa flash chống ghi. Được nhúng vào trong chip CPU, Boot Rom tải Bootloader vào Ram để thực thi.

Bootloader nó chịu trách nhiệm tải hệ điều hành từ thiết bị, thiết lập môi trường để hệ điều hành có thể chạy và khởi động ứng dụng.

Kernel nó dùng để thiết lập bộ nhớ đệm, tải các driver và file system, khởi động các daemon kernel,… Sau khi thiết lập hệ thống hoàn tất, nó sẽ tìm kiếm tiến trình init trong file hệ thống và khởi chạy tiến trình root đầu tiên.

Init là một tiến trình quan trọng tại đây nó thực thi tập lệnh init.rc. Nếu bạn nhìn vào tập lệnh init.rc nó sẽ bao gồm các lệnh như “start vold” cho file system và “trigger zygote” để bắt đầu tiến trình Zygote trong Android. Một trong những điều quan trọng nhất xảy ra trong quá trình init này là gọi một lệnh khởi động Zygote, lệnh app\_process khởi động ART hoặc Dalvik VM và cũng gọi đến phương thức chính trong Zygote.

Zygote là quy trình dành riêng cho Android khi hệ điều hành Android khởi động, Zygote có nhiệm vụ là launch ứng dụng Android, có thể xem Zygote là cha của các tiến trình ứng dụng, lệnh app\_process sẽ khởi tạo DVM và gọi phương thức main() của Zygote. Sau đó Zygote được start liền load các class Java, các thư viện, API, ngoài ra còn start System Server và mở socket nhằm để lắng nghe các request start ứng dụng Android. Khi nhận được tín hiệu request, Zygote sẽ gọi phương thức fork() để cấp phát vùng nhớ riêng cho ứng dụng yêu cầu cũng như các thư viện. Do đó ta sẽ tiết kiệm được thời gian không cần phải tải các thư viện vì được sử dụng chung do tiến trình Zygote chia sẻ.

System Server được start bởi Zygote, nó tạo ra các service và thêm vào Service Manager. Các service như: Activity Manager (quản lý lifecycle của các activity, service, content provider, broadcast receiver), Package Manager, Alarm Manager,…

Activity Manager sẽ theo dõi các activity được thực thi và cập nhật trạng thái của các activity trong một file. Dựa vào file, Activity Manager gọi Window Manager update những gì hiển thị lên màn hình cho user.

# 

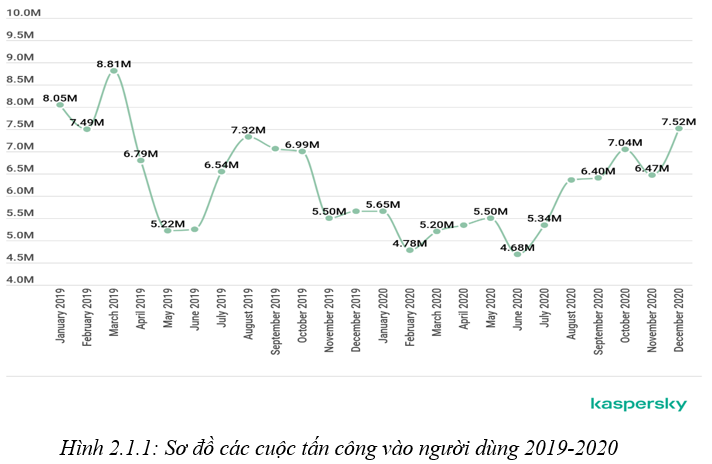
# TỔNG QUAN VỀ MÃ ĐỘC TRÊN ANDROID

1. **GIỚI THIỆU VỀ MÃ ĐỘC**

Mã độc (tên tiếng anh là malware được viết tắt từ malicious software) là các chương trình có chứa đoạn mã độc hại như: chiếm quyền kiểm soát điện thoại, tự động đọc SMS, tự nhắn tin, tự động tải ứng dụng, đánh cắp thông tin người dùng tài khoản mật khẩu, mã hóa dữ liệu,… Đó là những hành vi của một chương trình có chứa mã độc.

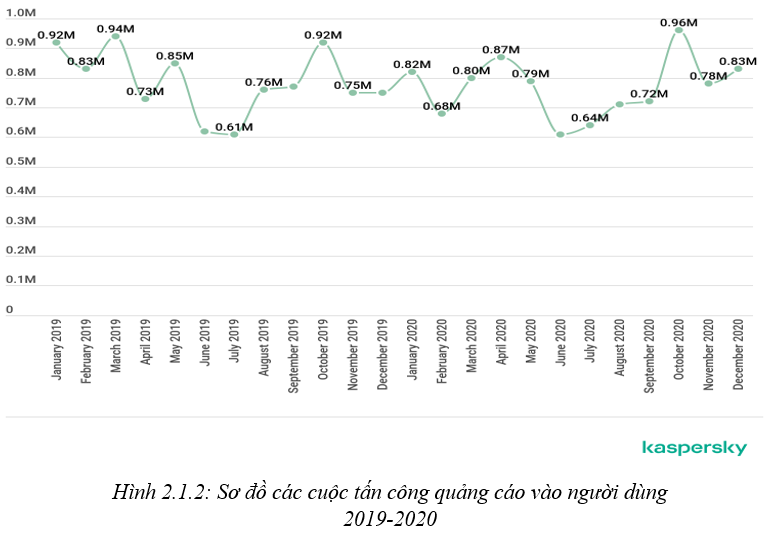
Hiện nay với nhu cầu của cuộc sống thay đổi liên tục smartphone là nhu cầu không thể thiếu đối với con người, nó phục vụ nhu cầu như liên lạc nhắn tin gọi thông qua một ứng dụng có kết nối Internet, thanh toán online sử dụng các ví điện tử liên kết thẻ ngân hàng Momo, ZaloPay, Viettel Pay,... Do đó những thách thức liên quan đến phần mềm độc hại Android cũng đang leo thang với tốc độ nhanh chóng.

Vào năm 2020 theo số liệu công bố từ Kaspersky đã phát hiện 5.683.694 malicious package, 156.710 new mobile banking Trojans, 20.208 new mobile ransomware. Và vào năm này cũng là năm đại dịch “covid” chúng đã lợi dụng bằng cách đặt tên các file apk như: covid.apk, covidMapv8.1.7.apk, tousanticovid.apk, covidMappia\_v1.0.3.apk, coviddetect.apk các ứng dụng này được đặt lên các trang có liên kết thông qua phân phối thư gác.

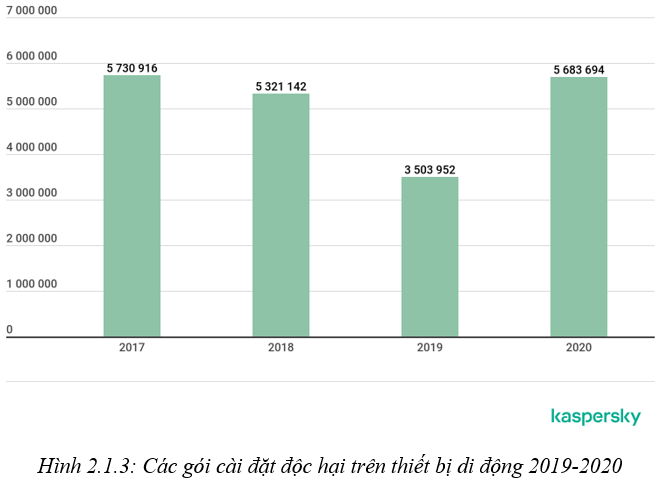


**Hình 2-1. Sơ đồ các cuộc tấn công vào người dùng 2019-2020**

Đáng chú ý ngoài phần mềm độc hại thì phần mềm quảng cáo cả hai rất gần giống nhau về khả năng. Thông thường, mã chạy quảng cáo được nhúng vào ứng dụng của nhà cung cấp dịch vụ ví dụ như trò chơi di động miễn sao là nó phổ biến.



**Hình 2-2. Sơ đồ các cuộc tấn công quảng cáo vào người dùng 2019-2020**



**Hình 2-3. Các gói cài đặt mã độc trên thiết bị di động 2019-2020**

1. **PHÂN LOẠI MÃ ĐỘC**
   1. **Một số phương pháp lây nhiễm mã độc trên hệ điều hành android**

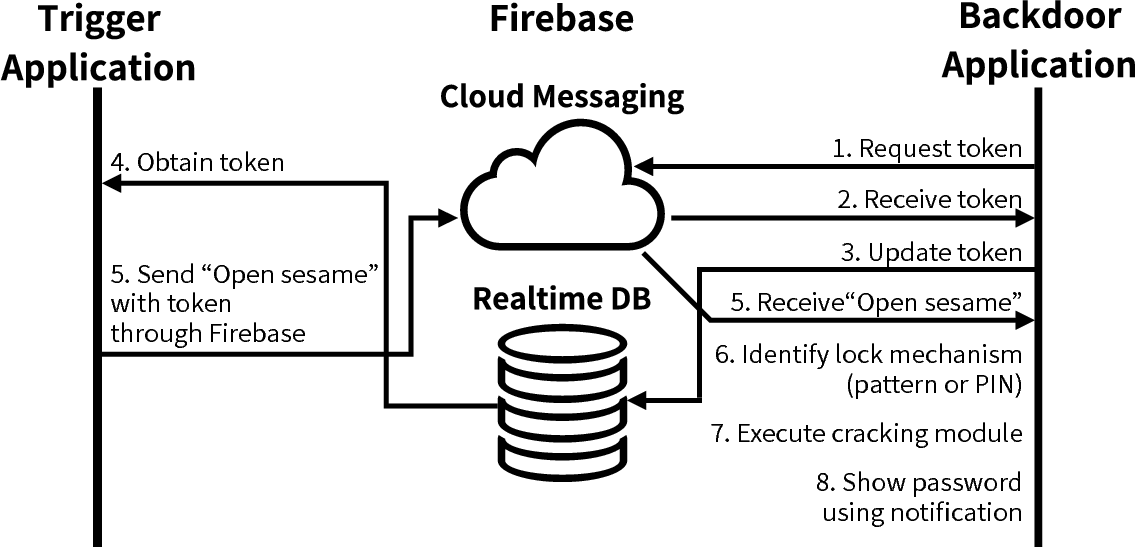
Phần mềm độc hại có hành vi nguy hiểm, có thể bao gồm cài đặt phần mềm mà không cần sự đồng ý của người dùng. Các loại lây nhiễm mã độc phổ biến trên điện thoại Android:

* Adding malicious code: Đây là phương pháp đưa mã độc vào thiết bị Android dựa trên một ứng dụng hợp pháp, khi một ứng dụng đã được đóng gói hợp pháp và được đưa lên Google Store hacker sẽ lợi dụng ứng dụng đó và sử dụng công cụ dịch ngược sau đó chèn thêm các đoạn mã độc và đóng gói lại ứng dụng đó. Khi người dùng cài đặt ứng dụng này thì mã độc đã xâm nhập vào hệ thống tùy theo mục đích của hacker.
* Exploiting application vulnerabilities: Ứng dụng của người dùng được cài đặt trên thiết bị Android bị lỗi. Hacker dựa vào những lỗ hỏng này từ ứng dụng để xâm nhập và có thể cài đặt ứng dụng độc hại cho thiết bị.
* Using technology for evil: Đây là một phương pháp tấn công vào thiết bị phần cứng trên điện thoại. Vào năm 2017 một lỗ hổng đối với Bluetooth có tên là “Blueborne” cho phép người tấn công truy cập vào nhiều thiết bị khác nhau. Thiết bị Android có thể nhận phần mềm độc hại từ những thiết bị lân cận, hoặc khi phần mềm độc hại được gửi qua tin nhắn văn bản hoặc tin nhắn đa phương tiện, người nhận có thể vô tình chấp nhận và cài đặt virus trên thiết bị của họ.
* Fake application: Đây là phương pháp dựa vào icon và tên của ứng dụng hợp pháp, hacker tiến hành xây dựng ứng dụng tương tự nhưng khi cài đặt có những đoạn mã độc đang chạy ngầm.
* Installation from afar: Đây là phương pháp tấn công hacker sẽ cài đặt ứng dụng độc hại từ xa lên điện thoại người dùng, thông qua họ tấn công vào trình duyệt web của thiết bị di động.
  1. **Các loại mã độc phổ biến trên hệ điều hành Android**

Có nhiều cách để phân loại mã độc thành một số loại nhất định với các đặc điểm cụ thể như: tính nhân bản, sự nhiễm độc, tàng hình, lệnh và điều khiển, tập hợp các hành vi được thể hiện trong quá trình chạy trên hệ điều hành Android. Mặc dù không có sự thống nhất chung về việc phân loại phần mềm độc hại tuy nhiên ta có thể phân loại phần mềm độc hại dựa trên mục đích và hành vi của chúng: Worms, trojans, spyware, ransomware, adware.

* Worms là một chương trình có thể sao chép và gửi file từ thiết bị này sang thiết bị khác, khi đã gửi qua một thiết bị mới có khả năng sẽ tiếp tục nhân bản. Worms trên thiết bị android thường được truyền qua tin nhắn văn bản SMS/MMS và thường không yêu cầu thực thi bởi người dùng, Worms không được xem là virus vì nó không nhúng vào chương trình khác.
* Trojans là một loại phần mềm giả mạo, chúng thường ẩn nấu dưới danh nghĩa là một phần mềm hữu ích nhưng chúng thực hiện một số chức năng độc hại như thu thập thông tin người dùng, đánh cắp tài khoản.
* Spyware là loại phần mềm đe dọa cho thiết bị của bạn, bằng cách thu thập và phát tán thông tin cá nhân, phần mềm loại này sẽ giám sát hoạt động của bạn, ghi lại vị trí của bạn và tải các thông tin quan trọng, tài khoản Gmail, tài khoản website. Trong nhiều trường hợp phần mềm spyware được đóng gói cùng với phần mềm vô hại nhưng nó đang thu thập dữ liệu dưới chế độ nền.
* Ransomware là một loại mã độc tống tiền, chúng tấn công người dùng bằng cách mã hóa các tài liệu, hình ảnh, video để buộc người dùng phải trả một số tiền. Một số loại còn khóa thiết bị và yêu cầu tiền chuộc để khôi phục quyền kiểm soát của người dùng.
* Adware là loại mã độc được thiết kế để hiển thị quảng cáo lên màn hình thiết bị của người dùng, thường ngụy trang dưới một chương trình hợp pháp hoặc ẩn mình trong một phần mềm khác để lừa người dùng cài đặt nó. Adware kiếm tiền bằng cách hiển thị quảng cáo cho người dùng.
* Backdoor là một loại phần mềm độc hại cung cấp cho kẻ tấn công quyền truy cập từ xa vào thiết bị. Backdoor có thể là một phần mềm hoàn toàn mới hoặc là một phần của phần mềm vô hại mà mã độc Backdoor được nhúng vào. Phần mềm độc hại Backdoor thu thập thông tin cá nhân từ điện thoại, gửi và nhận tin nhắn, gọi điện và thu thập lịch sử cuộc gọi, trong một số trường hợp nghiêm trọng Backdoor sẽ root (là cách bạn có thể có được quyền quản trị trên thiết bị Android từ đó có thể can thiệp sâu vào file hệ thống) thiết bị mà nó đã được cài đặt.
  1. **Giới thiệu phương pháp tấn công backdoor sử dụng Firebase**

Như đã giới thiệu bên trên Backdoor là một phần mềm tạo ra một kết nối từ thiết bị tấn công đến một thiết bị điều khiển. Sau đây là một phương pháp tấn công backdoor sử dụng Firebase.



**Hình 2-4. Quy trình tấn công sử dụng firebase**

Firebase là một nền tảng mà Google cung cấp để phát triển các ứng dụng di động và trang web, hỗ trợ các developer trong việc đơn giản hóa các thao tác dữ liệu, với Firebase Cloud Messaging ta có thể gửi nhận tin nhắn còn được gọi là push notification.

Hình trên là quá trình hoạt động của một backdoor. Quá trình triển khai bao gồm ba thành phần: Trigger application, Firebase, Backdoor application. Dùng Firebase để hỗ trợ giao tiếp giữa kẻ tấn công và ứng dụng backdoor, Việc sử dụng như vậy gây ra khó khăn cho các phần mềm kiểm tra mã độc, vì nhiều ứng dụng bình thường cũng sử dụng Firebase cho các dịch vụ thông báo.

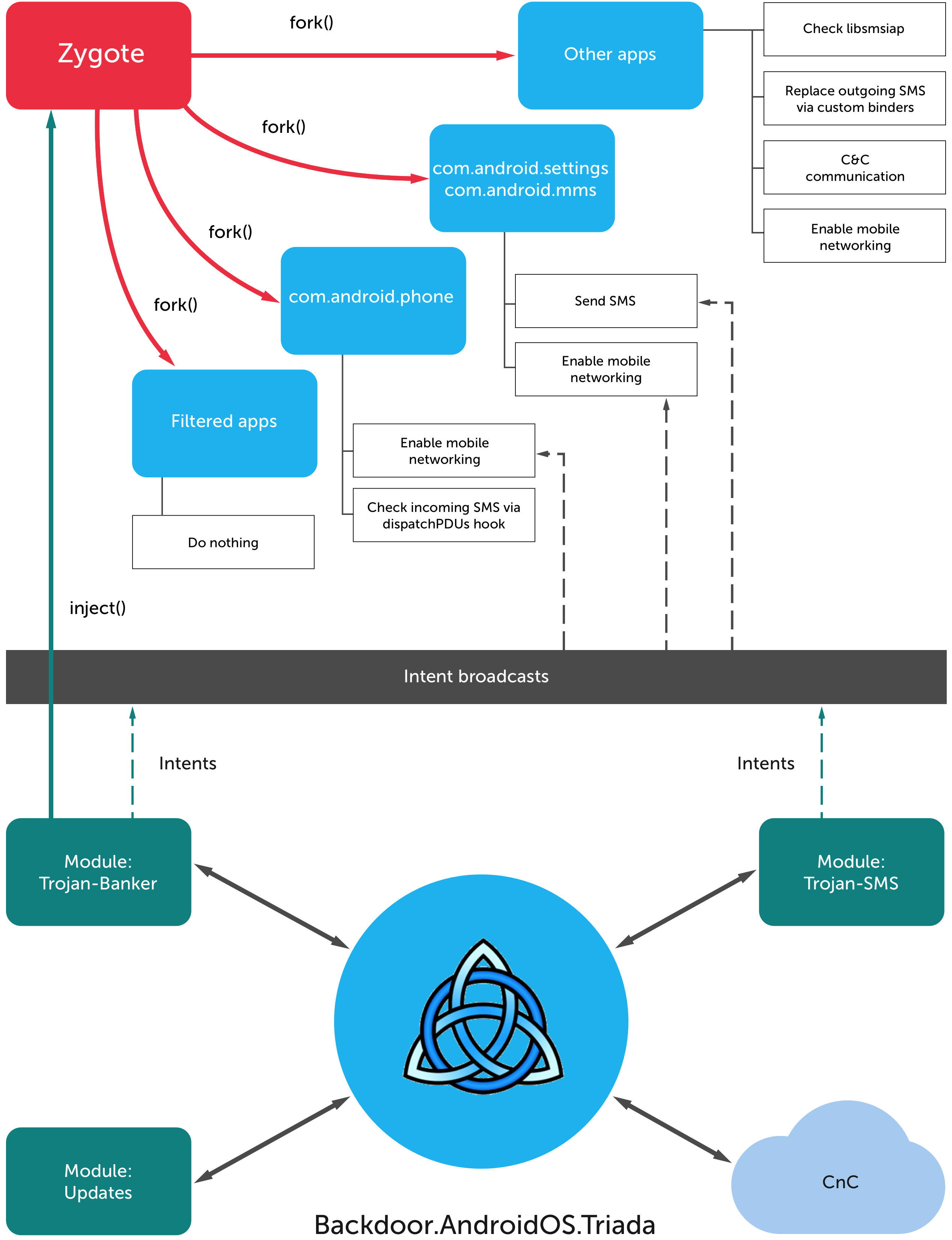
Đầu tiên khi một ứng dụng có tồn tại mã độc Backdoor đã cài đặt xong, chúng sẽ gửi một request đến Firebase, sau đó Firebase sẽ nhận và response lại một mã token cho ứng dụng, tiếp tục ứng dụng sẽ update thời gian thực từ database của Firebase và gửi một mã token cho Firebase, từ Firebase nó sẽ tiến hành gửi token nhận được từ ứng dụng Backdoor gửi đến một thiết bị remote, và khi muốn tấn công thiết bị remote sẽ gửi đến một messaging cùng với token cho Firebase và Firebase sẽ chuyển thông báo này đến thiết bị đã bị tấn công. Tiếp tục khi nhận được thông báo từ Firebase, ứng dụng backdoor đó sẽ thực thi module bẻ khóa mã Pin.

Về việc thiết kế mã độc Backdoor này, thứ nhất thiết kế kích hoạt từ xa do đó một Backdoor muốn thực thi thì phải có yêu cầu từ kẻ tấn công chứ không được tự động kích hoạt. Thứ hai ứng dụng Backdoor này giao tiếp trực tiếp với Firebase không cần giao tiếp trực tiếp với kẻ tấn công. Do đó các hệ thống phân tích không thể phát hiện ứng dụng độc hại này với các ứng dụng lành tính khác do một số ứng dụng cũng sử dụng Firebase.

Tiến trình bẻ khóa đối với mã Pin 4 chữ số, do chọn 5000 mẫu dữ liệu đối với mã Pin 4 chữ số có khả năng 98,72% trong số các mẫu Pin đã thử nghiệm thì được bẻ khóa trong vòng 5 giây.

* 1. **Giới thiệu mã độc Trojan Triada**

Theo Kaspersky thì Triada là Trojan nguy hiểm trên nền tảng Android hiện nay vì chúng lây nhiễm vào một quy trình cốt lõi của hệ điều hành này là Zygote. Triada có thể thay thế các file hệ thống và lây nhiễm vào quá trình cốt lõi của hệ điều hành Android như Zygote, nó thâm nhập sâu và kiểm soát các file hệ thống và điều khiển tiến trình Zygote.



**Hình 2-5. Quy trình tấn công Triada**

Triada còn có khả năng hoạt động ẩn, ngay sau khi được cài đặt nó ngay lập tức nhiễm vào quá trình Zygote bằng cách thay thế các file hệ thống và sau đó di chuyển hoạt động của nó đến bộ nhớ RAM của thiết bị, nơi mà các sản phẩm bảo mật không thể quét đến. Triada thu thập dữ liệu gửi nó đến một máy chủ C&C (Command & Control Server) để tạo hồ sơ cho nạn nhân và gửi ngược lại cho Triada một cái gọi là tập tin cấu hình, chúng có tổng cộng 17 máy chủ C&C với trên 4 tên miền khác nhau. Tập tin này chứa thông tin về các thiết lập của Triada, các hoạt động độc hại mà nó cần phải thực hiện và các module yêu cầu tải xuống để thực hiện các lệnh của nó.

Do Triada ẩn mình trong RAM nên rất khó phát hiện. Tất cả các module của nó cũng được triển khai trong RAM của thiết bị và nếu người dùng khởi động thiết bị của họ, Triada cũng sẽ khởi động cùng quá trình Zygote, nó yêu cầu một tập tin cấu hình mới và cài đặt lại module của nó sau mỗi lần khởi động lại

* 1. **Giới thiệu về mã độc Trojan Alien**

Từ tháng 1 năm 2020 một mã độc Trojan Banking mới phát hiện được gọi là Alien đang xâm nhập vào các thiết bị Android trên toàn thế giới. Alien sử dụng các tính năng nâng cao để vượt qua các biện pháp bảo mật xác thực hai yếu tố nhầm đánh cắp thông tin đăng nhập của nạn nhân.

Alien cũng như các loại malware khác thu thập thông tin danh bạ, SMS, vị trí. Tuy nhiên Alien có một tiến trình đặc biệt kiểm tra, cho phép nó truy cập vào tất cả các bản cập nhật mới trên thiết bị. Điều này cũng bao gồm mã 2FA code cho phép mã độc vượt qua các biện pháp bảo mật 2FA.

2FA (Two-factor Authentication) có nghĩa là xác thực hai yếu tố hoặc còn gọi là phương thức xác minh bảo mật hai yếu tố.

Alien chiếm được 2FA code cũng như phiên bản cập nhật mới bằng cách lợi dụng quyền nhận nội dung của thông báo trên thanh trạng thái thiết bị nhiễm mã độc. Quyền này là “android.permission.BIND\_NOTIFICATION\_LISTENER\_SERVICE”.

Để chiếm được quyền này Alien sử dụng tính năng truy cập từ xa trên ứng dụng TeamViewer, cho phép kẻ tấn công đằng sau mã độc điều khiển từ xa thiết bị nạn nhân. Khi TeamViewer được kích hoạt thành công, kẻ tấn công có toàn quyền điều khiển từ xa đối với giao diện người dùng thiết bị từ đó có thể truy cập và thay đổi cài đặt thiết bị, cài đặt hoặc gỡ ứng dụng, đồng thời sử dụng những ứng dụng ngân hàng trên thiết bị.

1. **PHƯƠNG PHÁP PHÂN TÍCH MÃ ĐỘC**
2. **Mục tiêu phân tích mã độc**

Phân tích mã độc là một cách để đề phòng hoặc ngăn ngừa các loại mã độc có thể tấn công vào thiết bị xâm chiếm thông tin cũng như có thể tấn công hệ thống mạng, cũng giúp ích cho việc khôi phục lại dữ liệu hoặc thông tin hiện trạng ban đầu và truy tìm ra được nguồn gốc tấn công của mã độc.

Khi bị tấn công điều đầu tiên chúng ta cần phải hiểu mã độc này đã tấn công vào phần nào của hệ thống, kiểm tra lại toàn bộ thiết bị xem thiết bị nào đã bị tấn công cũng như tìm kiếm tất cả các tệp tin xem tệp tin nào đã bị tấn công lây nhiễm mã độc, ứng dụng lây nhiễm mã độc.

Do mã độc được phân thành nhiều loại tùy theo chức năng và cách thức lây nhiễm nên việc phân tích truy tìm ra loại mã độc đó thuộc loại nào là cần thiết. Điều này giúp ta có thể hiểu được cách tấn công, giảm thiệt hại. Qua việc phân tích ta có thể biết mã độc đó là một mã độc biến thể hay một mã độc mới chưa được biết đến.

1. **Phương pháp phân tích tĩnh**

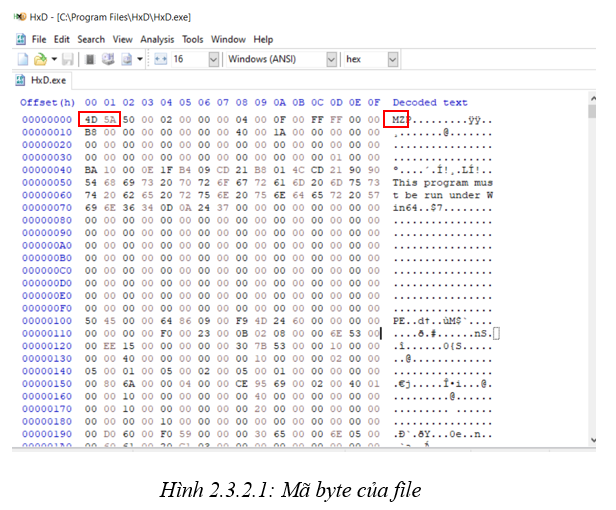
Phân tích tĩnh là loại phương pháp dùng để phân tích mã độc mà không cần thực thi mã độc. Ở loại phân tích này ta chỉ phân tích nội dung của các file xem file đó có chứa mã độc không, hiểu về những đoạn mã thực thi trong file làm nhiệm vụ gì có ảnh hưởng gì đến hệ thống hoặc thiết bị hay không, hiểu thêm về các lời gọi hàm, mã nguồn chương trình dưới dạng Assembly.

Đây là phương pháp phân tích có thể thực hiện một cách nhanh chóng, tuy nhiên đối với mã độc thì việc phân tích tĩnh thường gặp khó khăn do phần mềm độc hại có thể được mã hóa để tránh các loại phân tích, đóng gói, ngụy trang.

Một số kỹ thuật được sử dụng cho việc phân tích như: Sử dụng mã hash để xác định malware đó, kiểm tra định dạng file,… Mỗi kỹ thuật sẽ cung cấp thông tin khác nhau để cho mục đích phục vụ nghiên cứu.

Hash là phương pháp phổ biến để nhận diện và xác định phần mềm độc hại, khi một file được hash thì sẽ tạo ra một chuỗi ký tự, chuỗi ký tự này được xem là chữ ký của file, mỗi một file khác nhau sẽ cho ra một loại chữ ký khác nhau. Người ta dựa vào mã hash để xem file đó có phải là malware hay không.

Mỗi file có các byte định dạng riêng, việc chúng ta phân tích file bằng công cụ Hex Editor chúng ta sẽ thu được nội dung của file được thể hiện ở mã ASCII hoặc Unicode. Từ đó chúng ta kiểm tra các byte định dạng để xác định file đó là loại file gì. Những file thực thi luôn có các byte định dạng MZ (thể hiện dưới dạng hex 4D 5A), các file nén có định dạng là PK (thể hiện dưới dạng hex 50 4B). Việc phân tích như vậy chuỗi có thể giúp chúng ta phát hiện một số loại malware được lưu dưới dạng khác để đánh lừa người dùng ví dụ: Malware netcat là một file thực thi, nhưng chúng được lưu dưới dạng file nén để đánh lừa người dùng.



**Hình 2-6. Mã binary file sử dụng công cụ HxD**

1. **Phương pháp phân tích động**

Phân tích động là phương pháp theo dõi và phân tích hành vi của phần mềm ứng dụng thông qua môi trường thực thi các phần mềm. Phương pháp này thì hiệu quả trong việc xác định các chức năng của malware, ví dụ nếu mã độc của bạn là một keylogger thì phân tích động sẽ cho bạn biết file log của keylogger trên hệ thống, giúp bạn khám phá ra những bản ghi mà keylogger nắm giữ, giải mã ra thông tin mà keylogger gửi đi.

Phân tích động là một kỹ thuật hữu ích trong việc phân tích malware, tuy vậy phương pháp này làm hệ thống chúng ta đặt trong tình trạng nguy hiểm do có nhiều mối nguy hại từ malware, do đó chúng ta phải thiết lập một môi trường Sandbox an toàn để bảo vệ hệ thống.

Một số cách thức khai thác triệt đối malware như là chúng ta phân tích mẫu malware đó trong một thời gian nhất định, sau đó chúng ta tiếp tục phân tích lại nhưng lần này chúng ta thay đổi cấu hình hệ thống và tính toán thời gian phân tích có giống với cấu hình lần trước ta đã phân tích hay không.

# TỔNG QUAN VỀ MÁY HỌC

1. **GIỚI THIỆU VỀ MÁY HỌC**

Những năm gần đây, AI (Artificial Intelligence) và cụ thể hơn Machine Learning nổi lên như một bằng chứng cho cuộc cách mạng công nghiệp lần thứ tư. Trí tuệ nhân tạo đang len lỏi vào mọi lĩnh vực trong đời sống mà có thể chúng ta không nhận ra như: Xe tự hành Tesla, trợ lý ảo Siri của Apple, hệ thống gợi ý sản phẩm của Amazon, hệ thống gợi ý phim của Netflix, máy chơi cờ vây AlphaGo của Google.

Machine learning là một tập con của AI. Nó có khả năng tự học hỏi dựa trên dữ liệu được đưa vào và chúng ta cần phải làm sạch dữ liệu và tìm mô hình đúng đắn. Ứng dụng của học máy những năm gần đây xuất hiện ở khắp mọi nơi từ việc nhận dạng khuôn mặt, vân tay, cử chỉ, giọng nói, xử lý các mô hình ảnh y khoa,… Học máy có thể ứng dụng hầu hết các loại dữ liệu từ cấu trúc (bảng số liệu) và cả dữ liệu phi cấu trúc (hình ảnh, văn bản, tính hiệu âm thanh).

Ý tưởng của học máy là dựa trên một số thuật toán xây dựng mô hình để giải quyết vấn đề cụ thể như: phân lớp, phân loại, hồi quy,…

1. **PHÂN LOẠI CÁC THUẬT TOÁN TRONG MÁY HỌC**

Có hai cách để chúng ta phân loại các thuật toán trong máy học một là chúng ta dựa vào phương thức học và hai là chúng ta dựa vào chức năng của thuật toán. Khi ta dựa vào phương thức học thì thuật toán học máy của chúng ta được chia lại thành bốn loại như: học tăng cường (reinforcement learning), học có giám sát (supervised learning), học không giám sát (unsupervised learning), học bán giám sát (semi-supervised learning).

* 1. **Học tăng cường (reinforcement learning)**

Học tăng cường (reinforcement learning) là học củng cố hay tăng cường, là lĩnh vực liên quan đến dạy cho máy (agent) thực hiện tốt một nhiệm vụ (task) bằng cách tương tác với môi trường (environment) thông qua hành động (action) và nhận được kết quả (reward). Cách học như thế này tương tự như con người học từ môi trường xung quanh bằng cách thử sai, ví dụ khi mùa đông đến con người thường tiếp cận bếp lửa để sưởi ấm lúc đó ta cảm nhận được khi càng tiến lại gần thì càng ấm ngược lại càng xa thì sẽ ít ấm hơn, và khi chúng ta tiếp cận đến lúc chạm vào bếp lửa thì chúng ta sẽ bị bỏng. Trong ví dụ trên, reward là kết quả của việc ta thực hiện hành động (action) tiếp cận càng gần bếp lửa.

* 2. **Học có giám sát (supervised learning)**

Học có giám sát (supervised learning) là thuật toán dự đoán đầu ra (outcome) của một dữ liệu (input) dựa trên các cặp dữ liệu đã có sẵn (data, label). Ví dụ thuật toán dò các khuôn mặt trong facebook, facebook sử dụng thuật toán này để chỉ ra các khuôn mặt trong một bức ảnh yêu cầu người dùng có thể gắn tag friend. Việc này ta có cặp dữ liệu có sẵn data và label tương đương với khuôn mặt và tên người đầu ra outcome là một danh tính xác thực, dữ liệu data và label càng lớn thì độ chính xác càng cao.

Học có giám sát được chia thành 2 loại chính như sau: phân loại (classification), hồi quy (regression). Bài toán phân loại (classification) là bài toán khi kết quả đầu ra outcome của các dữ liệu đầu vào input data được phân thành các nhóm, ví dụ: phân loại hoa, phân loại màu sắc, phân loại email spam,… Bài toán hồi quy (regression) là bài toán dựa vào các kết quả đầu vào input data và kết quả đầu ra outcome sẽ là một giá trị cụ thể, ví dụ có một căn nhà rộng x m2, có bao nhiêu phòng ngủ và cách thành phố z km thì sẽ có giá là bao nhiêu.

* 1. **Học không giám sát (unsupervised learning)**

Học không giám sát (unsupervised learning) đây là bài toán mà chúng ta có dữ liệu đầu vào input data mà không biết kết quả đầu ra outcome. Chúng ta chỉ có thể dựa vào những đặc tính của cấu trúc dữ liệu đầu vào mà chúng ta sẽ có thể phân nhóm (clustering) hoặc giảm số chiều (dimension reduction) để dễ tính toán.

Bài toán học không giám sát được chia thành hai loại: Bài toán phân nhóm (clustering), bài toán kết hợp (association). Bài toán phân nhóm (clustering) là bài toán mà ta dựa vào sự liên quan về những đặc tính của dữ liệu đầu vào input data để ta có thể phân chúng thành những mỗi nhóm khác nhau ví dụ: phân nhóm khách hàng dựa trên hành vi mua hàng, giống như ta cho những đứa trẻ những mảnh giấy có màu sắc khác nhau và rồi tập hợp chúng lại theo từng màu sắc. Bài toán kết hợp (association) là bài toán khi chúng ta muốn tìm ra một quy luật chung cho những dữ liệu cho trước ví dụ những người mua mì gói thường sẽ mua thêm xúc xích, hay khi mua giày sẽ mua thêm vớ.

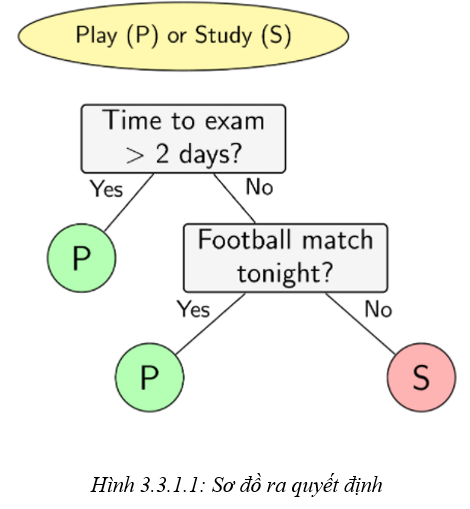
* 1. **Học bán giám sát (semi-supervised learning)**

Học bán giám sát (semi-supervised learning) là bài toán mà ta có tập dữ liệu X mà chúng ta chỉ xác định được một số label thôi không xác định được toàn bộ label của tập dữ liệu X. Ví dụ điển hình của nhóm này là chỉ có một phần ảnh của con người và động vật được gán nhãn những ảnh còn lại thì không biết được nhãn, chúng ta có thể sử dụng unsupervised learning để khám phá ra các cấu trúc của dữ liệu đầu vào.

Phân loại bài toán dựa trên chức năng của các thuật toán, các thuật toán hồi quy (linear regression, logistic regression, stepwise regression), thuật toán phân loại (support vector machine, kernel svm), thuật toán phân cụm (k-mean clustering, k-Medians), artificial neural network algorithms (perceptron, softmax regression).

1. **GIỚI THIỆU THUẬT TOÁN HỌC MÁY HIỆU QUẢ**
   1. **Thuật toán cây quyết định**

Sắp đến kì thi học kì II một bạn học sinh suy nghĩ về bản thân nên chơi hay nên học để ôn bài thi. Cậu học sinh suy nghĩ nếu còn nhiều hơn 1 tuần trước khi thi thì mình sẽ đi chơi bóng đá và xem phim, còn khi hai ngày nữa thi thì sẽ ôn bài sẽ không đi chơi. Việc ra quyết định này của cậu học sinh cũng tương tự như ta trả lời các câu hỏi.



**Hình 3-1. Cây quyết định cho ví dụ**

Sơ đồ hình 3-1 được xem là một mô hình ra quyết định. Việc quan sát suy nghĩ và ra quyết định của con người thường bắt đầu từ các câu hỏi. Machine learning cũng có một mô hình ra quyết định dựa trên các câu hỏi, mô hình này có tên là cây quyết định (decision tree).

Cây quyết định là một mô hình supervised learning (học có giám sát), có thể áp dụng vào cả hai bài toán classification và regression. Cây quyết định gồm 3 phần chính: 1 node gốc (root node), những node là (leaf node), và các nhánh của nó (branches). Node gốc là node đầu tiên của cây quyết định, việc tìm ra được thuộc tính nào làm node gốc mới là vấn đề của cây quyết định.

Với các bài toán có nhiều thuộc tính và mỗi thuộc tính có nhiều giá trị khác nhau, việc tìm được nghiệm tối ưu thường là không khả thi. Thay vào đó, một phương pháp đơn giản được sử dụng là tại mỗi bước, một thuộc tính tốt nhất sẽ được chọn ra dựa trên một tiêu chuẩn nào đó. Với mỗi thuộc tính được chọn, ta chia dữ liệu vào các child node tương ứng với các giá trị của thuộc tính đó rồi tiếp tục áp dụng phương pháp này làm cho mỗi child node.

Sau mỗi câu hỏi dữ liệu được phân chia vào từng child node tương ứng với các câu trả lời cho câu hỏi đó. Câu hỏi ở đây chính là một thuộc tính, câu trả lời chính là giá trị của thuộc tính đó. Một phép phân chia tốt nhất là dữ liệu tại mỗi child node không thể phân chia được nữa.

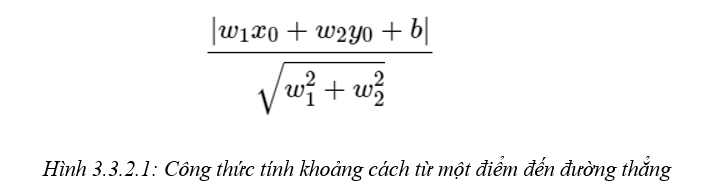
Trọng số của entropy tại các leaf node sau khi xây dựng cây quyết định được coi là hàm mất mát. Các trọng số ở đây tỉ lệ với số điểm dữ liệu được phân vào mỗi node. Công việc của chúng ta là tìm các cách phân chia hợp lý sao cho hàm mất mát cuối cùng đạt giá trị càng nhỏ càng tốt.

* 1. **Thuật toán SVM**

Support vector machine (SVM) dùng để xây dựng một siêu mặt phẳng (hyperplane) để phân lớp (classify) tập dữ liệu thành hai lớp riêng biệt. Một siêu mặt phẳng là một hàm tương tự như phương trình đường thẳng y = ax + b. Trong thực tế, nếu ta chỉ cần phân lớp tập dữ liệu gồm hai feature siêu phẳng lúc này chính là một đường thẳng.

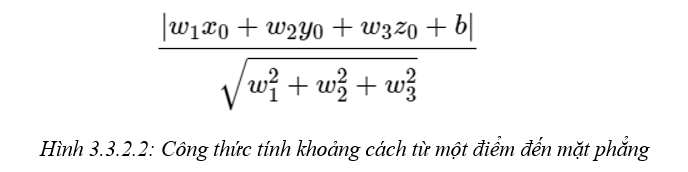
Về ý tưởng ban đầu thì SVM sử dụng thuật toán để ánh xạ dữ liệu ban đầu vào không gian nhiều chiều hơn. Khi ánh xạ tập dữ liệu sang nhiều chiều không gian từ đó SVM sẽ xem xét và chọn ra một siêu mặt phẳng phù hợp nhất để phân chia dữ liệu đó.

Trong không gian hai chiều, ta biết rằng khoảng cách từ một điểm có tọa độ (x0, y0) tới một đường thẳng có phương trình w1x + w2y + b = 0 được xác định bởi công thức.



**Hình 3-2. Công thức tính khoảng cách từ một điểm đến đường thẳng**

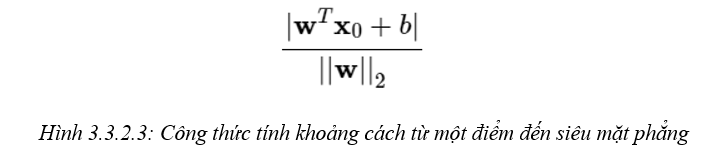
Trong không gian ba chiều, khoảng cách từ một điểm có tọa độ (x0, y0, z0) tới một mặt phẳng có phương trình w1x + w2y + w3z + b = 0 được xác định bởi công thức



**Hình 3-3. Công thức tính khoảng cách từ một điểm đến mặt phẳng**

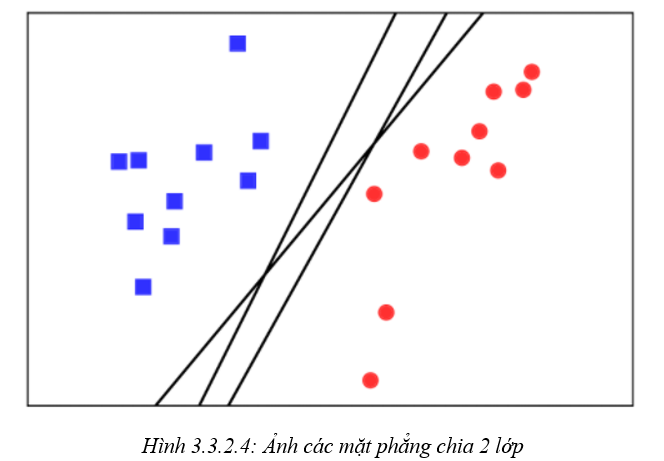
Nếu ta bỏ dấu giá trị tuyệt đối ở tử số, chúng ta có thể xác định điểm đó nằm về phía nào của đường thẳng hay mặt phẳng đang xét. Những điểm làm cho biểu thức trong dấu giá trị tuyệt đối mang dấu dương nằm về cùng một phía (gọi tạm đây là phía dương), những điểm làm cho biểu thức trong dấu giá trị tuyệt đối mang giá trị âm nằm về phía còn lại (gọi là phía âm). Những điểm nằm trên đường thẳng hoặc mặt phẳng sẽ làm cho tử số có giá trị bằng 0, tức khoảng cách từ điểm đang xét đến đường thẳng hoặc mặt phẳng bằng 0.

Từ đây chúng ta có thể tổng quát lên không gian nhiều chiều. Khoảng cách từ một điểm (vector) có tọa độ x0 đến một siêu mặt phẳng có phương trình wTx + b = 0 được xác định bởi công thức.



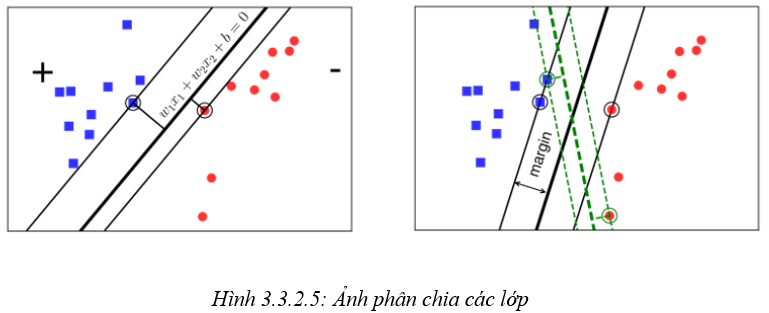
**Hình 3-4. Công thức tính khoảng cách từ một điểm đến siêu mặt phẳng**

Với với d là số chiều không gian. Giả sử chúng ta có hai lớp khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, bây giờ chúng ta cần tìm một mặt phẳng phân chia chính xác hai classes đó. Tức là tất cả các điểm thuộc một lớp sẽ nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại. Ta hãy xem xét hình dưới đây



**Hình 3-5. Các mặt phân chia hai lớp**

Trong các mặt phẳng đó đâu là mặt phẳng được phân chia tốt nhất, trong hình 3-5 ta thấy có hai mặt phẳng lệch về phía lớp có các điểm màu đỏ. Điều này sẽ gây ra việc lớp có các điểm màu đỏ sẽ cảm thấy bị hạn chế các điểm dữ liệu trên không gian so với lớp có các điểm màu xanh.



**Hình 3-6. Phân chia các lớp**

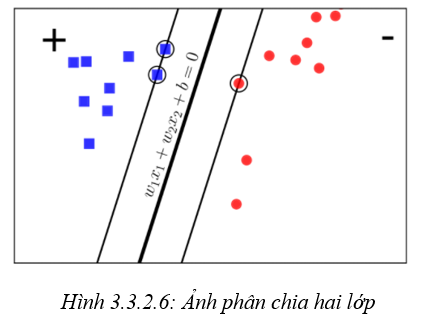
Nếu ta định nghĩa mức độ hạnh phúc của một lớp tỉ lệ thuận với khoảng cách gần nhất từ một điểm của lớp đó đến đường thẳng hoặc mặt phẳng thì lớp màu đỏ ở phía bên trái hình 3-6 sẽ không được hạnh phúc cho lắm vì khoảng cách từ điểm gần nhất của mỗi lớp đến đường hoặc mặt phẳng phân chia thì không đồng đều.

Vì vậy chúng ta cần tìm một đường hoặc mặt phẳng phân chia làm sao cho các điểm (vector) của hai lớp đến mặt phẳng là đồng đều. Tức là khoảng các điểm (vector) của lớp màu đỏ và lớp màu xanh gần nhất với đường thẳng hoặc mặt phẳng và cách đều các điểm (vector). Khoảng cách này được gọi mà margin.

Chúng ta nhìn khung bên phải hình 3-6 xét hai cách phân chia bởi đường nét liền màu đen và đường nét đứt màu xanh, rõ ràng ta thấy đường nét liền phân chia tốt hơn vì nó tạo ra margin rộng hơn đường nét màu xanh.

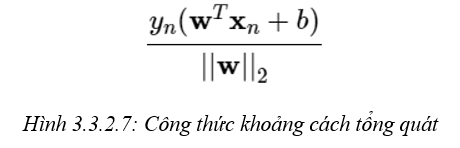
Việc margin rộng hơn sẽ mang lại hiệu ứng phân lớp tốt hơn vì việc phân chia giữa các lớp là rạch ròi hơn. Bài toán tối ưu trong SVM chính là bài toán đi tìm đường phân chia sao cho margin lớn nhất.

Giả sử ta có các cặp dữ liệu training set là (x1, y1), (x2, y2),…(xN, yN), với , và yi là nhãn của dữ liệu đó, d là số chiều của dữ liệu, N là số điểm dữ liệu, mỗi điểm dữ liệu được xác định bởi yi = 1 (lớp 1) và yi = -1 (lớp 2). Để dễ hình dung ta nhìn hình phía dưới đây.



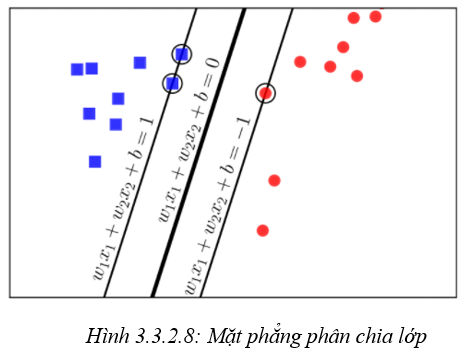
**Hình 3-7. Phân chia hai lớp**

Các điểm (vector) ô vuông xanh thuộc lớp 1, các điểm tròn đỏ thuộc lớp 2 và mặt phân chia hai lớp ta có là là mặt phẳng phân chia giữa hai lớp, lớp 1 nằm về phía dương và lớp 2 nằm về phía âm của mặt phân chia. Ta quan sát thấy một điểm quan trọng như sau với cặp dữ liệu (xn, yn) bất kỳ, khoảng cách từ điểm đó tới mặt phân chia là



**Hình 3-8. Công thức tính khoảng cách tổng quát**

Ta có thể thấy nó gần giống công thức tính khoảng cách ở hình 3.8, thực chất chúng như nhau. Khi ta định nghĩa lớp 1 là phía dương thì ta có ngược lại ta có lớp 2 định nghĩa là phía âm nên có do đó suy ra công thức chung đối với tất cả các điểm (vector) là . Từ đó ta suy ra công thức đối với hình 3-9, ta nhìn hình dưới đây sẽ dễ hiểu hơn.



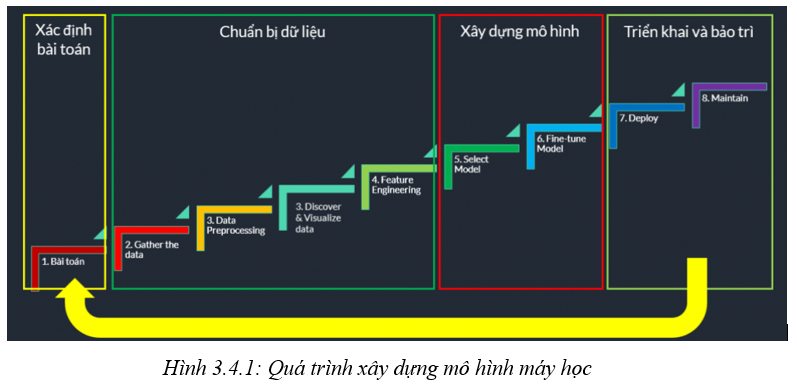
**Hình 3-9. Mặt phẳng phân chia lớp**

Đối với mỗi điểm dữ liệu ta có từ đó ta có thể tính được margin bằng cách tính khoảng cách gần nhất từ một điểm đến mặt phẳng phân chia . Bài toán SVM chính là bài toán tìm w và b sao cho margin này đạt giá trị lớn nhất.

1. **QUÁ TRÌNH XÂY DỰNG MÔ HÌNH MÁY HỌC**
   1. **Tổng quát**

Mô hình máy học là một quy tắc tổng quát hóa từ dữ liệu được đưa vào huấn luyện thông qua một thuật toán huấn luyện. Các thuật toán huấn luyện thực sự không nhiều, tuy nhiên hiệu quả của mô hình thì khác nhau, đó là vì cách thức sử dụng dữ liệu và chất lượng là chất lượng dữ liệu là khác nhau. Do đó, khi dữ liệu đưa vào có chất lượng rất tệ thì mô hình khi sử dụng thực tế cũng sẽ tệ theo, tức là dữ liệu không tổng quát hóa được thống kê của dữ liệu thực tế làm cho kết quả sử dụng thực tế khác xa so với khi huấn luyện làm cho kết quả dự đoán hoàn toàn bị sai lệch. Điều này cũng như ta sử dụng nguyên liệu ôi thiu để nấu ăn thì mặc định món ăn đó sẽ không ngon.

Để xây dựng mô hình máy học hiệu quả, thông thường sẽ phụ thuộc vào 3 yếu tố sau: điều đầu tiên chúng ta cần phải hiểu chúng ta cần giải quyết bài toán gì thì mới có thể xây dựng mô hình cần thiết, tiếp theo là ta cần có dữ liệu chất lượng sự phù hợp của dữ liệu với bài toán đặt ra, dữ liệu đưa vào huấn luyện đòi hỏi phải chính xác, cuối cùng chúng ta cần tìm một thuật toán để cho ra kết quả tốt nhất, chẳng hạn với số lượng dữ liệu nhỏ thì áp dụng các mô hình phức tạp sẽ không hiệu quả và ngược lại do đó việc lựa chọn thuật toán sử dụng cũng cực kỳ quan trọng. Về cơ bản quy trình xây dựng và triển khai ứng dụng sử dụng machine learning tổng quát gồm bốn bước sau.



**Hình 3-10. Quá trình xây dựng mô hình máy học**

* 1. **Xác định bài toán**

Đầu tiên là xác định bài toán là một trong những vấn đề cốt lỗi nhất của việc xây dựng ứng dụng machine learning. Bước này giúp chúng ta hiểu được vấn đề và nghiệp vụ xử lý cần phải thực hiện của bài toán, ở đây chúng ta cần thực hiện xác định rõ vấn đề cần giải quyết, phân loại vấn đề là học có giám sát hay không giám sát hay bài toán phân loại hoặc hồi quy, tiếp tục là chúng ta cần xác định tính hiệu quả của mô hình cần xây dựng, xác định tính cần thiết và tính khả thi của việc xây dựng liệu có nên sử dụng mô hình máy học hay không.

* 1. **Chuẩn bị dữ liệu**

Chuẩn bị dữ liệu sau khi đã xác định được bài toán chúng ta tiến hành đến quá trình chuẩn bị dữ liệu, quá trình này bao gồm: thu thập dữ liệu (garther the data), tiền xử lý dữ liệu (data preprocessing), khám phá và biểu diễn dữ liệu (discover và visualize data), chọn tính đặc trưng (feature engineering).

Thu thập dữ liệu (garther the data) chất lượng của các mô hình machine learning phụ thuộc rất lớn vào số lượng và chất lượng của dữ liệu thu thập do vậy quá trình thu thập dữ liệu có thể coi là một trong số những bước quan trọng nhất và chúng ta cần phải xác định được thu thập dữ liệu từ đâu, thu thập những dữ liệu gì và đặc tính, cấu trúc của dữ liệu là gì, làm sao để thu thập và tập hợp dữ liệu từ các nguồn khác thành bộ dữ liệu chung.

Tiền xử lý dữ liệu (data preprocessing) trên thực tế khi thu thập dữ liệu, dữ liệu thu thập có nhiều dạng biểu diễn khác nhau và thông thường không thống nhất chẳng hạn: dạng các bảng có cấu trúc, dữ liệu hình ảnh, âm thanh, video,…Máy không thể hiểu được các dạng dữ liệu thô như thế này, do đó chúng ta cần biến đổi dữ liệu để biến chúng thành những dữ liệu mà máy có thể hiểu được nói cách khác chuyển chúng thành các thuộc tính (feature) để sử dụng được trong các thuật toán machine learning. Dữ liệu “bẩn” xảy ra khi chúng ta thu thập dữ liệu thiếu “missing data” chúng ta cần phải xử lý những dữ liệu này.

Khám phá và biểu diễn dữ liệu (discover và visualize data) sau khi dữ liệu được tiền xử lý, chúng ta cần thực hiện một số bước để phân tích và khám phá dữ liệu nhằm hiểu được phân bố của dữ liệu cũng như các quan hệ giữa các thành phần trong dữ liệu, bước này là một tiền đề để chúng ta lựa chọn và xây dựng mô hình hiệu quả. Nếu ta thấy rằng mô hình không có quan hệ tuyến tính thì việc áp dụng các mô hình tuyến tính là không thực sự hiệu quả. Một số việc ta cần làm ở giai đoạn này là hiển thị dữ liệu để quan sát các xu hướng và quan hệ của dữ liệu, tìm ra các tính tương quan giữa các thuộc tính trong dữ liệu, loại bỏ những dữ liệu bẩn.

Chọn tính đặc trưng (feature engineering) đây là một trong những nhiệm vụ bắt buộc cần phải thực hiện khi xây dựng mô hình machine learning bởi việc chọn những thuộc tính đặc trưng phù hợp không những giúp xây dựng mô hình có độ chính xác cao hơn mà còn giúp xây dựng các mô hình đơn giản hơn, tốc độ thực thi cao hơn.

* 1. **Xây dựng mô hình**

Sau khi đã xây dựng và chuẩn hóa được toàn bộ dữ liệu, chúng ta bước vào giai đoạn xây dựng mô hình. Việc xây dựng mô hình sẽ bao gồm việc lựa chọn mô hình phù hợp và hiệu chỉnh các tham số của mô hình, để làm được việc này chúng ta thường chia tập dữ liệu được xây dựng từ các bước trên thành ba thành phần riêng biệt: Tập dữ liệu huấn luyện (training data), tập dữ liệu kiểm chứng (validation data) và tập dữ liệu kiểm thử (testing data).

Trong đó tập dữ liệu huấn luyện được sử dụng để huấn luyện mô hình, tập dữ liệu kiểm chứng được sử dụng để hiệu chỉnh các tham số của mô hình và tập kiểm thử được dùng để đánh giá hiệu quả của mô hình trên dữ liệu chưa được quan sát. Khi dữ liệu được chia thành ba phần chúng ta có thể bắt đầu quá trình xây dựng mô hình, quá trình này diễn ra theo ba bước: Lựa chọn mô hình (model selection), hiệu chỉnh mô hình (fine-tune model), đánh giá mô hình (model evaluation).

Lựa chọn mô hình (model selection) nếu là bài toán phân lớp ta có thể chọn các mô hình sau: K-Nearest Neighbors, Artificial neural networks (ANN), Support vector machine (SVM), Random forests, Decision trees, Logistic regression, Naive Bayes,… Hay với bài toán hồi quy chúng ta có thể chọn các mô hình như: Kernel regression, Gaussian process regression, Regression trees, Linear regression, Support vector regression,… Sau khi đã lựa chọn mô hình, chúng ta có thể thực hiện chạy thử nó trên bộ dữ liệu huấn luyện nhằm tìm ra lựa chọn tối ưu nhất (đáp ứng các ràng buộc về độ đo hiệu quả).

Hiệu chỉnh mô hình (fine-tune model): là quá trình cho phép tìm ra các tham số tối ưu nhất cho mô hình, và lựa chọn đánh giá mô hình hiệu quả nhất trên bộ dữ liệu kiểm chứng (validation data). Quá trình này được thực hiện sau khi chúng ta đã lựa chọn được một lớp mô hình phù hợp cho bài toán và mục tiêu của quá trình này là nâng cao hiệu quả của mô hình huấn luyện thông qua việc tối ưu hóa các tham số, hay siêu tham số của mô hình.

Đánh giá mô hình (Model Evaluation) là một phần không thể thiếu của quá trình xây dựng mô hình. Nó giúp tìm ra mô hình tốt nhất đại diện cho dữ liệu của chúng ta và dự kiến được khả năng hoạt động của mô hình được chọn khi triển khai thực tế trong tương lai.

* 1. **Triển khai và bảo trì**

Sau khi có mô hình chúng ta có thể lưu mô hình này dưới dạng phù hợp chẳng hạn các luật if-else với cây quyết định, các đường phân tách lớp với SVM, các tham số đối với mạng nơ-ron, dưới dạng các tệp tin hoặc cơ sở dữ liệu có định dạng phù hợp. Các mô hình này sau đó sẽ được sử dụng như các thành phần trong chương trình ứng dụng nhằm đưa ra các dự đoán.

Sau khi mô hình đã được huấn luyện và triển khai trong thực tế, chúng cần phải được theo dõi để xác định tính hiệu quả. Việc theo dõi các mô hình cho phép xử lý các giá trị thực và giá trị dự đoán từ đó đo lường lại hiệu quả của mô hình dựa trên các độ đo phù hợp, nhằm đánh giá xem mô hình được triển khai có thực sự phù hợp trong thực tế hay không để có sự điều chỉnh.

# TRIỂN KHAI ỨNG DỤNG

* 1. **THU THẬP DỮ LIỆU**

Quá trình thu thập dữ liệu là quá trình khó khăn, để có đủ bộ dữ liệu cho việc nghiên cứu tôi đã thu thập từ nhiều nguồn khác nhau trên internet, tìm kiếm dữ liệu từ các diễn đàn công nghệ. Thu thập thông qua các link từ Github.

Bộ dữ liệu thu thập được chia làm hai loại, loại đầu tiên là những file apk có mã độc tôi đã thu thập được 669 mẫu, loại thứ hai là những file apk không có chứa mã độc tôi đã thu thập được 552 mẫu. Dựa vào các dữ liệu thu thập được ta tiến hành chia bộ dữ liệu thành hai phần, dữ liệu để training và dữ liệu dùng để kiểm tra. Đối với mỗi loại dữ liệu malware và bengin, ta tiến hành lấy mỗi loại 100 mẫu dùng để kiểm tra. Những mẫu còn lại ta dùng để training. Cuối cùng đối với mẫu malware dùng để training ta có 569 mẫu và bengin có 452 mẫu, và 200 mẫu để kiểm tra.

* 1. **ĐÁNH GIÁ MÔ HÌNH**

Bài toán mà chúng ta giải quyết ở đây thuộc loại bài toán classification, vì kết quả của chúng ta là phân nhóm. Nhóm nào thuộc loại malware và nhóm nào thuộc loại bengin, quá trình training ta sử dụng các permission có trong file apk của android để tiến hành đưa vào model để tiến hành dự đoán. Với phương pháp lấy permission trong file apk thì là phương pháp phân tích tĩnh.

Qua quá trình tìm hiểu ta có được 175 permission trên hệ điều hành android. Những permission này tượng trưng cho một quyền của hệ thống để sử dụng những chức năng của ứng dụng, ta tiến hành đọc những permission này vào file csv để làm thuộc tính do đó ta có 175 cột để làm thuộc tính, ta còn thêm cột tên file, mã hash md5, và cột kết quả xem file đó thuộc loại file malware hoặc bengin. Cột kết quả này có giá trị là 1 và 0, và những cột có thuộc tính permission thì mang giá trị là 1 và 0 nếu những file nào có permission nào thì tại thuộc tính đó sẽ mang giá trị là 1 ngược lại mang giá trị là 0.

Ta chọn được ba thuật toán máy học: Cây quyết định, SVM (support vector machine), LBFGS. Kết quả ta thu được với thuật toán cây quyết định xác suất đoán đúng 87% đối với tập dữ liệu kiểm tra là malware và đoán sai là 13%, với tập dữ liệu kiểm tra bengin thì đoán đúng 81% và đoán sai là 19%. Thuật toán SVM xác suất đoán đúng 87% đối với tập dữ liệu malware và đoán sai là 13%, với tập dữ liệu bengin thì đoán đúng 82% và đoán sai 18%.

Thuật toán LBFGS xác suất đoán đúng 83% đối với tập dữ liệu malware và đoán sai là 17%, với tập dữ liệu bengin đoán đúng 84% và đoán sai là 16%.

* 2. **CÔNG NGHỆ SỬ DỤNG**
  3. **Thư viện ML.NET**

Khi nói đến học máy và khoa học dữ liệu, Python được coi là một ngôn ngữ truyền thống, do tính đơn giản và khả năng truy cập vào các thư viện và khuôn khổ tuyệt vời. Các nhà khoa học dữ liệu thích đào tạo các mô hình của họ bằng Python bằng cách sử dụng các thư viện khác nhau như TensorFlow, Scikit Learn,… Trong khi đó, các nhà phát triển ứng dụng sẽ thường sử dụng các ngôn ngữ lập trình khác để tích hợp và sử dụng các mô hình đó như C #, Java,…Do đó, việc đưa Mô hình ML vào sản xuất một cách liền mạch đã trở thành một điểm khó khăn đối với nhiều tổ chức (do sự khác biệt giữa công nghệ và khuôn khổ).

ML.NET là câu trả lời của Microsoft cho vấn đề này. Nó là một khuôn khổ học máy đa nền tảng và mã nguồn mở cho phép các nhà phát triển .NET xây dựng, đào tạo và sử dụng các mô hình học máy tùy chỉnh của họ trong khi vẫn ở trong hệ sinh thái .NET.

ML.NET được đóng gói với một cơ chế công cụ mạnh mẽ được gọi là Model Builder. Nó đơn giản hóa quá trình xây dựng mô hình thông qua Automated ML (hoặc AutoML), giúp các nhà phát triển dễ dàng thiết lập và chạy mô hình của họ chỉ bằng cách tải dữ liệu. AutoML đảm nhận phần còn lại của quá trình xây dựng mô hình bao gồm cả việc tạo mã để sử dụng các mô hình đó.

* 1. **WPF**

WPF là viết tắt của Window Presentation Foundation, là một hệ thống API mới hỗ trợ việc xây dựng giao diện đồ họa trên nền tảng Window. Được xem như thế hệ kế tiếp của WindowForm, WPF tăng cường khả năng lập trình viên bằng cách cung cấp API cho phép tận dụng những lợi thế về đa phương tiện hiện đại. Là một bộ phận của .NET Framework 3.0, WPF có sẵn trong Windows Vista và Windows Server 2008. Đồng thời, WPF cũng có thể hoạt động trên nền Window XP Service Pack 2 và cả Window Server 2003.

WPF cung cấp nền tảng thống nhất để xây dựng giao diện người dùng, trước khi WPF ra đời, việc tạo giao diện người dùng theo những yêu cầu mô tả ở ví dụ trên đòi hỏi sử dụng rất nhiều công nghệ khác nhau. Để tạo form, các control và các tính năng kinh điển khác của một giao diện đồ họa Windows, thông thường lập trình viên sẽ chọn Windows Forms, một phần của .NET Framework. Nếu cần hiển thị văn bản, Windows Forms có một số tính năng hỗ trợ văn bản trực tiếp hoặc có thể sử dụng Adobe’s PDF để hiển thị văn bản có khuôn dạng cố định.

Đối với hình ảnh và đồ họa 2 chiều, lập trình viên sẽ dùng GDI+, một mô hình lập trình riêng biệt có thể truy nhập qua Windows Forms. Để hiển thị video hay phát âm thanh, lập trình viên lại phải sử dụng Windows Media Player, và với đồ họa 3 chiều, anh ta lại phải dùng Direct3D, một thành phần chuẩn khác của Windows. Tóm lại, quá trình phát triển giao diện người dùng theo yêu cầu trở nên phức tạp, đòi hỏi lập trình viên quá nhiều kỹ năng công nghệ.

WPF là giải pháp hợp nhất nhằm giải quyết tất cả những vấn đề công nghệ nêu trên, hay nói cách khác, WPF cung cấp nhiều tính năng lập trình giao diện trong cùng một công nghệ đơn nhất. Điều này giúp cho quá trình tạo giao diện người dùng trở nên dễ dàng hơn đáng kể.

Trong thời đại bùng nổ của Internet, các ứng dụng Web ngày một phát triển. Việc trang bị giao diện người dùng với đầy đủ tính năng như một ứng dụng desktop sẽ thu hút nhiều người sử dụng, và do đó góp phần làm tăng giá trị doanh nghiệp. Tuy nhiên, như đã nêu trong phần đầu, với những công nghệ truyền thống, để phát triển một giao diện đồ họa vừa hoạt động trên desktop vừa trên trình duyệt Web, đòi hỏi phải sử dụng những công nghệ hoàn toàn khác nhau, giống như việc xây dựng hai giao diện hoàn toàn độc lập. Điều này tạo ra chi phí không cần thiết để phát triển giao diện.

WPF là một giải pháp cho vấn đề này. Lập trình viên có thể tạo ra một ứng dụng trình duyệt XAML (XBAP) sử dụng WPF chạy trên Internet Explorer. Trên thực tế, cùng đoạn code này có thể được dùng để sinh ứng dụng WPF chạy độc lập trên Windows.

* 1. **Bootstrap**

Bootstrap là 1 framework HTML, CSS, và JavaScript cho phép người dùng dễ dàng thiết kế website theo 1 chuẩn nhất định, tạo các website thân thiện với các thiết bị cầm tay như mobile, ipad, tablet,...

Bootstrap bao gồm những cái cơ bản có sẵn như: typography, forms, buttons, tables, navigation, modals, image carousels và nhiều thứ khác. Trong bootstrap có thêm nhiều Component, Javascript hỗ trợ cho việc thiết kế responsive của bạn dễ dàng, thuận tiện và nhanh chóng hơn.

Bootstrap là một trong những framework được sử dụng nhiều nhất trên thế giới để tạo ra các Responsive Website. Bootstrap đã tạo ra một tiêu chuẩn riêng, và rất được các lập trình viên ưa chuộng. Về cơ bản Bootstrap có ba ưu điểm: dễ sử dụng được xây dựng trên HTML, CSS, Javascript, ưu điểm tiếp theo là tính năng responsive tương tác với mọi loại thiết bị vì vậy bạn chỉ cần học cách sử dụng chúng, tính năng này giúp tiết kiệm rất nhiều thời gian cho người dùng khi tạo ra các Website thân thiện. Nó tương thích với tất cả các trình duyệt (Chrome, Firefox, Internet Explorer, Safari, Opera). Tuy nhiên, với IE, Bootstrap 4 chỉ hỗ trợ từ IE10 trở lên.

* 1. **Jquery**

Jquery không phải là một ngôn ngữ lập trình riêng biệt mà hoạt động liên kết với JavaScript. Với Jquery, bạn có thể làm được nhiều việc hơn mà lại tốn ít công sức hơn. Jquery cung cấp các API giúp việc duyệt tài liệu HTML, hoạt ảnh, xử lý sự kiện và thao tác Ajax đơn giản hơn. Jquery hoạt động tốt trên nhiều loại trình duyệt khác nhau.

Các tính năng quan trọng của Jquery nó là một thư viện khá gọn nhẹ có kích cỡ khoảng 19KB (kilobyte). Tương thích đa nền tảng chạy được trên Chrome, Safari, Android,…Dễ dàng tạo Ajax nhờ thư viện Jquery ta có thể tạo Ajax có thể dễ dàng tương tác với server và cập nhật nội dung tự động mà không cần tải lại trang.

* 1. **Entity Framework**

Entity Framework ra đời nhằm hỗ trợ sự tương tác giữa các ứng dụng trên nền tảng .NET với các cơ sở dữ liệu quan hệ. Entity Framework chính là công cụ giúp ánh xạ giữa các đối tượng trong ứng dụng, phần mềm của bạn với các bảng của một cơ sở dữ liệu quan hệ.

Entity Framework giúp các nhà phát triển Web tương tác với dữ liệu quan hệ theo phương pháp hướng đối tượng với ít mã hơn so với các ứng dụng truyền thống. Lợi ích lớn nhất của nó là giúp lập trình viên giảm thiểu việc lập trình mã nguồn để thực hiện truy cập và tương tác với cơ sở dữ liệu.

Nếu bạn thực hiện truy vấn dữ liệu. Bạn có thể sử dụng Linq to Entities để thao tác với objects được sinh ra từ Entity Framework, nghĩa là bạn không phải viết code sql. Việc update các classes, commands dễ dàng mỗi khi cơ sở dữ liệu có sự thay đổi, điều này giúp bạn tiết kiệm thời gian đáng kể. Entity Framework sẽ tự động tạo ra các classes, commands tương ứng cho việc select, insert, update, delete dữ liệu từ cơ sở dữ liệu quan hệ.

* 1. **ASP.NET Framework**

ASP.NET là một nền tảng ứng dụng web (web application framework) được phát triển và cung cấp bởi Microsoft, cho phép những người lập trình tạo ra những trang web động, những ứng dụng web và những dịch vụ web. Lần đầu tiên được đưa ra thị trường vào tháng 1 năm 2002 cùng với phiên bản 1.0 của.NET framework, là công nghệ nối tiếp của Microsoft's Active Server Pages (ASP). ASP.NET được biên dịch dưới dạng Common Language Runtime (CLR), cho phép những người lập trình viết mã ASP.NET với bất kỳ ngôn ngữ nào được hỗ trợ bởi.NET language.

Những trang ASP.NET, được biết đến như những web form, là khối chính trong phát triển ứng dụng. Những web form được chứa trong những file có phần mở rộng ASPX; những nhà phát triển có thể đặt nội dung tĩnh hoặc động vào trang aspx dưới dạng server-side Web Control và User Control. Ngoài ra, có thể viết mã bằng cách chèn <% -- mã cần viết -- %> vào trang web giống như những công nghệ phát triển web khác PHP, JSP và ASP, nhưng những công nghệ nào không hỗ trợ data binding khi nó phát sinh nội dung trang web.

* 1. **HƯỚNG DẪN SỬ DỤNG ỨNG DỤNG**
  2. **Triển khai ứng dụng trên nền tảng Web**

Ứng dụng được triển khai dựa trên ASP.NET Framework và thư viện ML.NET để triển khai sử dụng các thuật toán machine learning, một vài thư viện khác như Bootstrap, Jquery. Hoạt động của ứng dụng này là quét file apk và tìm ra file AndroidManifest.xml tiếp tục có module decompile file này trở thành dạng có thể đọc được, sau đó đọc các permission trong file.

Ứng dụng này thực hiện cơ chế upload file lên server sao đó quét file, cơ chế upload file sử dụng đối tượng XMLHttpRequest trong Javascript. Sau khi đã có file ứng dụng tiến hành quét file đọc permission trong file AndroidManifest.xml, khi đã có permission ứng dụng chuyển sang một module nữa là dự đoán kết quả thông qua các model máy học đã được training data sẵn. Và cuối cùng là xác nhận file có tồn tại virus hay không.

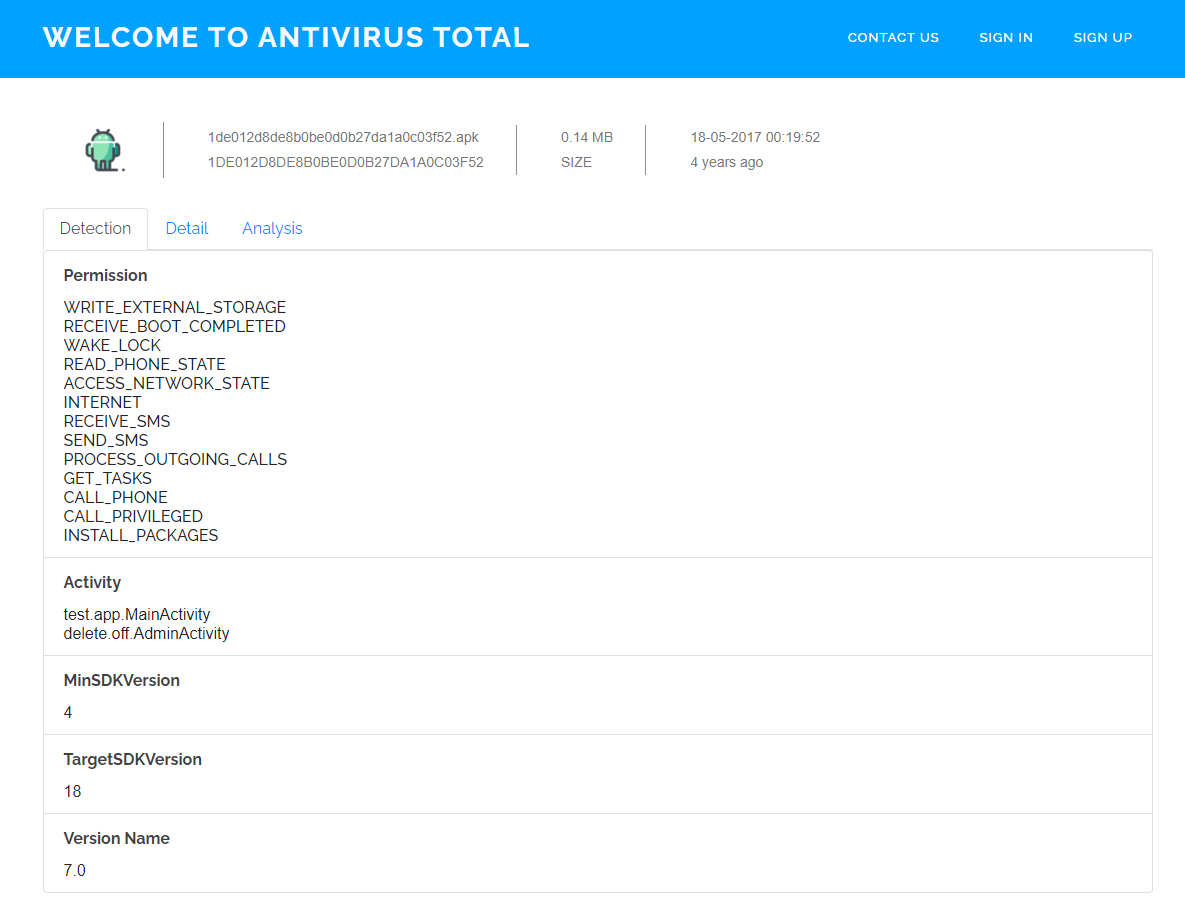
Quá trình thu thập dữ liệu bao gồm 569 mẫu virus và 452 mẫu lành tính, ứng dụng đã độc toàn bộ các file này đưa vào một file csv. Những permission được lấy trên trang chủ của Android và thu được 175 permission, khi đọc các file apk nếu file nào tồn tại các permission thì được đánh dấu là 1 ngược lại đánh dấu là 0.

Khi mà đã phân tích xong ứng dụng thực hiện cơ chế Ajax không load lại page để tăng tính giao tiếp giữa người dùng.



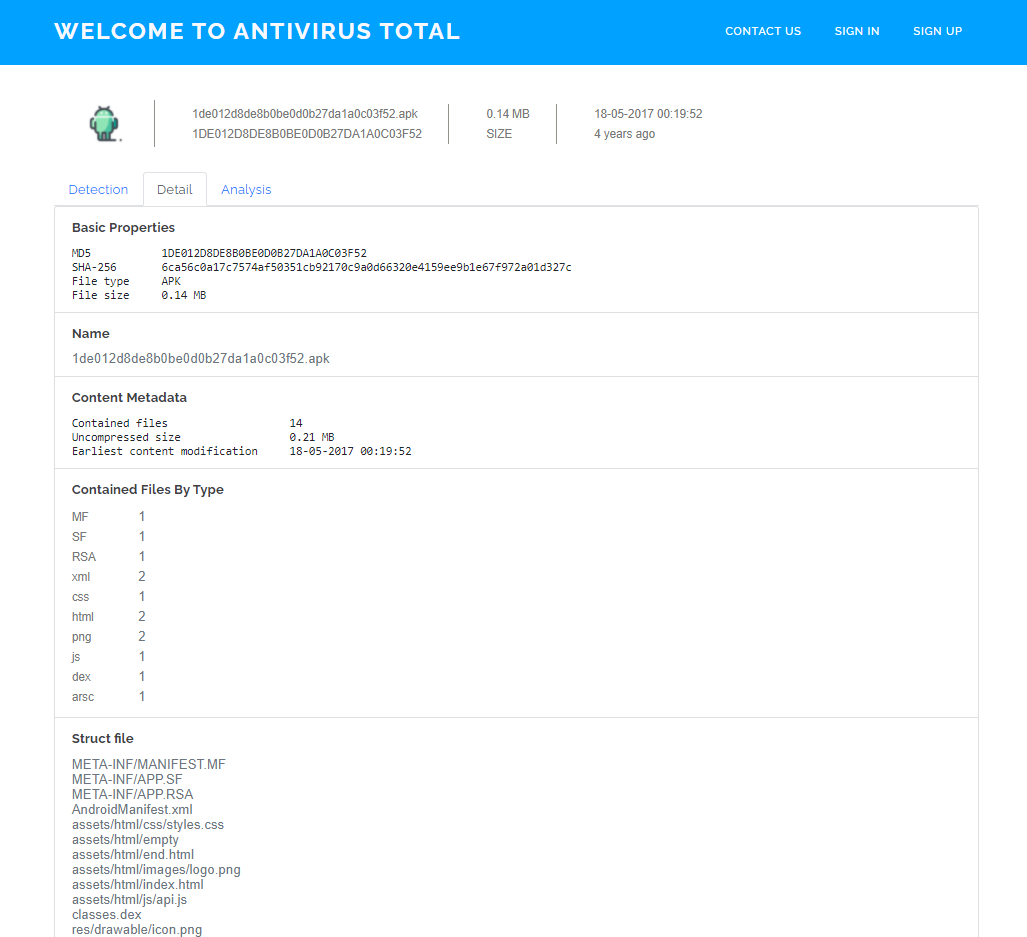
**Hình 4-1. Trang chủ ứng dụng web**

Đây là màn hình trang chủ web, nút button choose file dùng để chọn một file apk và sau đó nó sẽ tiến hành upload và phân tích.



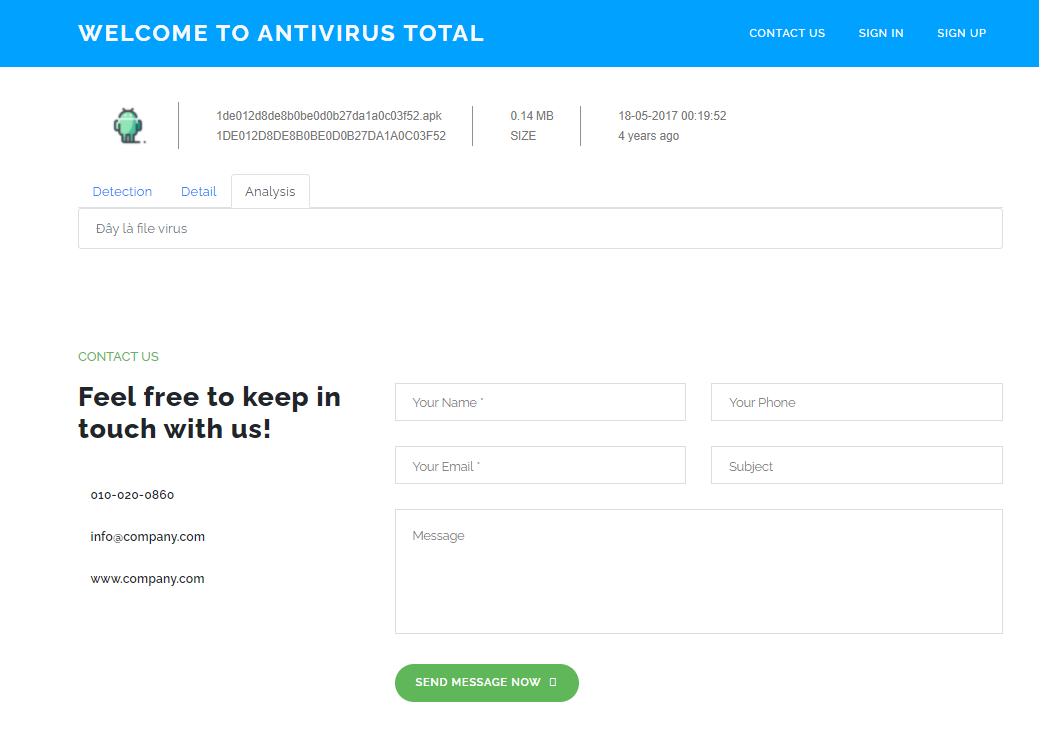
**Hình 4-2. Kết quả sau khi phân tích file apk**

Sau khi phân tích sẽ hiển thị thông tin về file như permission, activity, MinSDKVersion, TargetSDKVersion, VersionName.



**Hình 4-3. Kết quả chi tiết file apk**

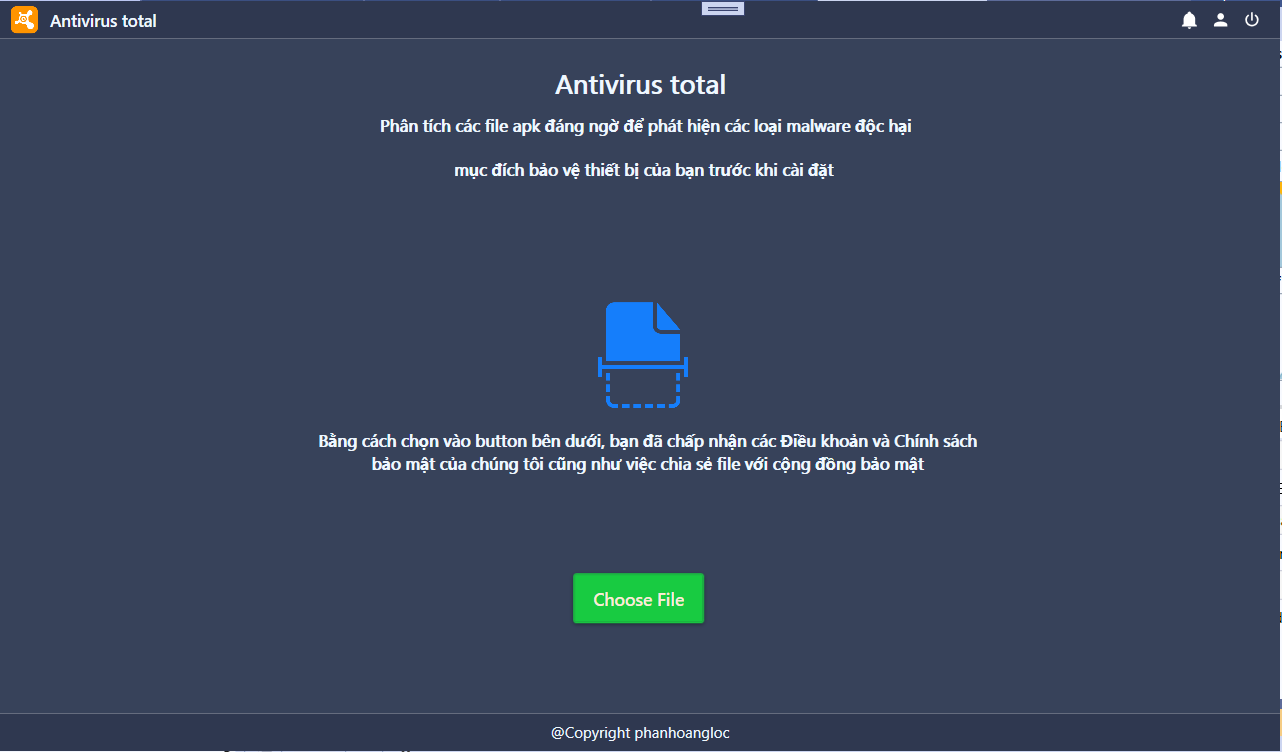
Đây là hiển thị kết quả chi tiết của một file apk và còn có mã hash MD5, SHA-256, cấu trúc file.



**Hình 4-4. Kết quả dự đoán**

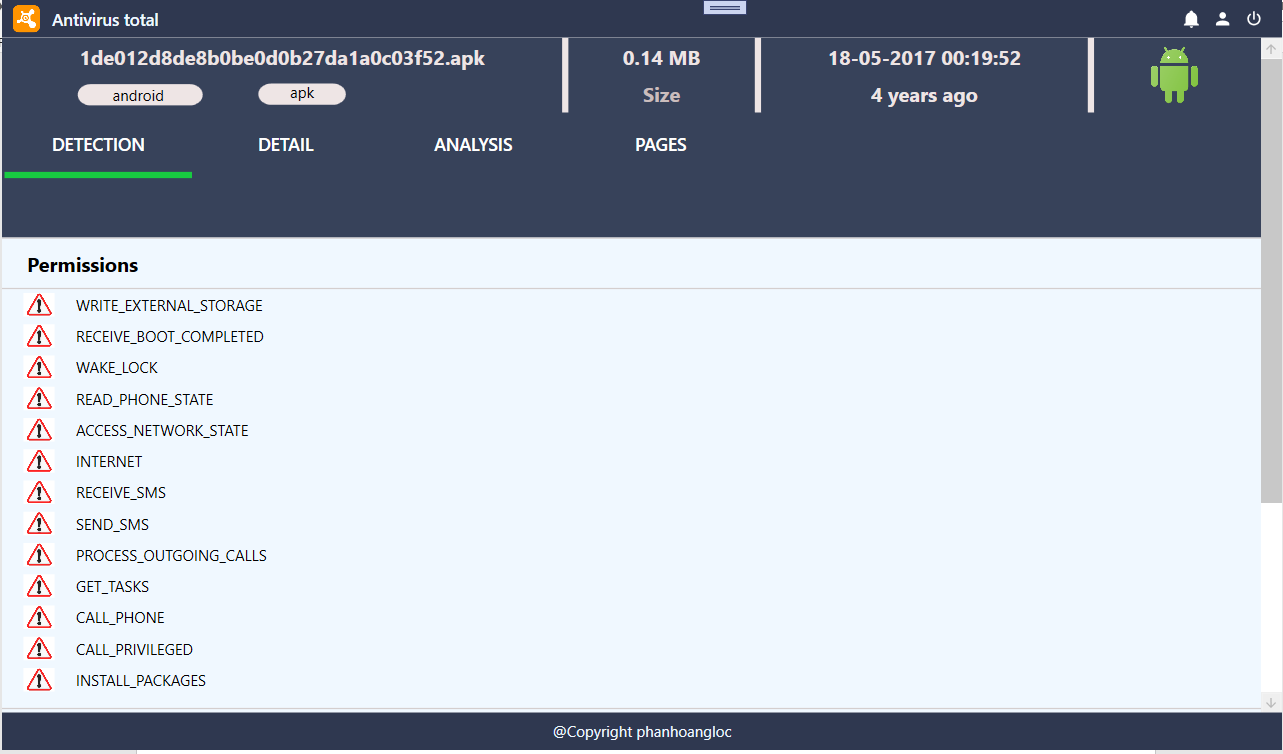
* 1. **Triển khai trên nền tảng Window**

Ngoài ứng dụng trên nền tảng web còn có trên nền tảng Window, cơ chế hoạt động cũng như đọc file apk sau đó đọc permission trong file AndroidManifest.xml sau đó đưa vào mô hình dự đoán machine learning và xuất kết quả ra màn hình.

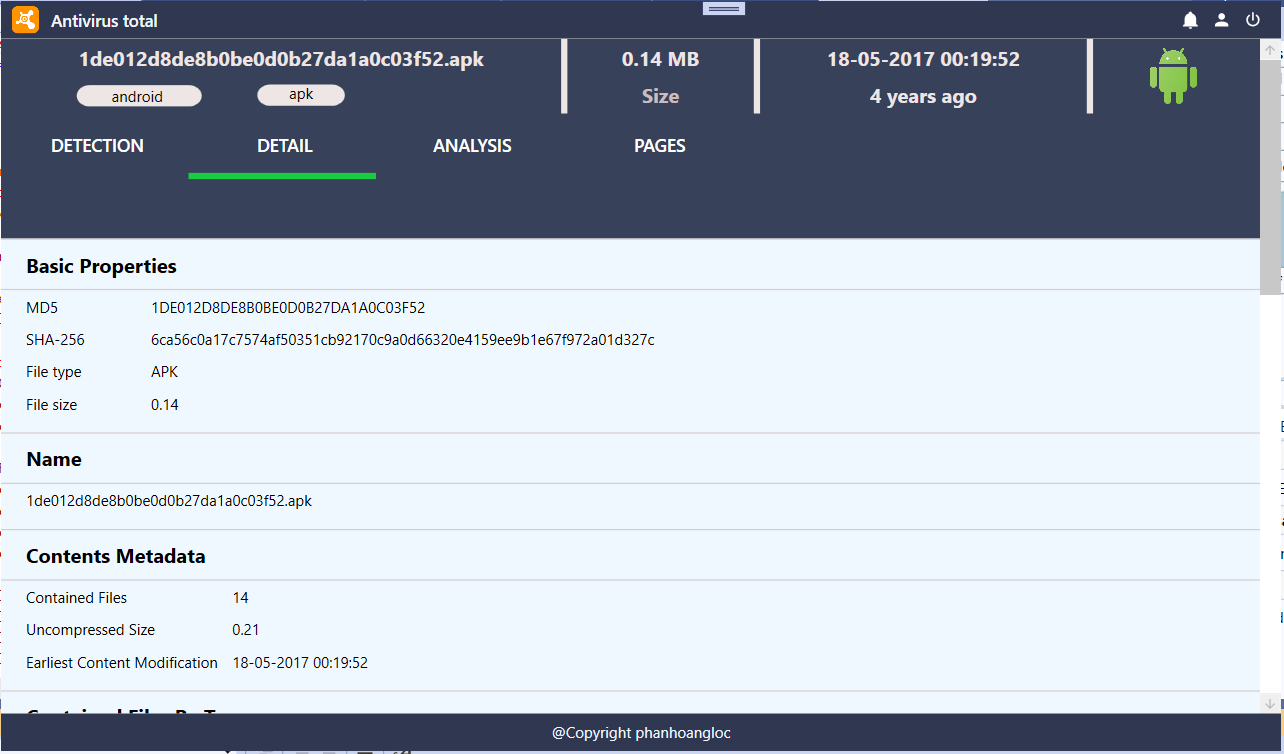


**Hình 4-5. Trang chủ màn hình**

Khi ứng dụng bắt đầu sẽ hiển thị trang chủ và một button dùng để chọn file, khi đã chọn file ứng dụng bắt đầu lấy permission từ file AndroidManifest.xml. Sau đó sẽ chạy các thuật toán máy học thông qua một mô hình training data đã có sẵn.



**Hình 4-6. Kết quả sau khi phân tích**



**Hình 4-7. Kết quả chi tiết**

Thông tin được hiển thị bao gồm danh sách permission, mã hash file MD5, SHA-256, tên file, kích thước file. Đếm xem có bao nhiêu file trong file apk, xem ngày chỉnh sửa lần gần nhất.

# 

# TỔNG KẾT

* 1. **KẾT QUẢ ĐẠT ĐƯỢC**

Do kiến thức có hạn và thời gian đầu tư cho đề tài chưa nhiều nên đề tài chỉ đạt được các kết quả như:

* Hiểu được về kiến trúc hệ điều hành Android, cơ chế bảo mật của hệ điều hành Android, phân vùng dữ liệu, hiểu được các tiến trình hoạt động trong Android. Quá trình đóng gói file apk có những cơ chế nào, trải qua những khâu như thế nào, các file trong file apk có ý nghĩa gì. Ngoài ra còn nắm bắt được quá trình lịch sử phát triển của hệ điều hành Android.
* Hiểu thêm về các loại mã độc trên hệ điều hành Android quá trình tấn công gây tổn hại như thế nào đến hệ điều hành Android, phân loại mã độc dựa trên hành vi, mô phỏng lại quá trình tấn công của một số loại malware. Có các phương pháp phân tích malware như phân tích động và phân tích tĩnh, các lợi ích và mục tiêu của việc phân tích này.
* Nhận biết thêm về Machine Learning là gì, áp dụng nó ra sau, các loại mô hình trong Machine Learning như: học tăng cường (reinforcement learning), học có giám sát (supervised learning), học không giám sát (unsupervised learning), học bán giám sát (semi-supervised learning). Tìm hiểu được thêm hai loại thuật toán cây quyết định và thuật toán SVM (Support vector machine).
* Phân tích được quá trình để xây dựng một mô hình máy học phải trải qua các giai đoạn nào, chọn những mô hình nào có hiệu quả khả quan cao, đánh giá bài toán, xác định mô, chọn lọc và xử lý dữ liệu.
* Luận văn đã trình bày một hướng tiếp cận có hiệu quả về ứng dụng học máy trong phát hiện nguy cơ mã độc trên nền tảng Android. Qua đây, ta có thể đánh giá việc sử dụng máy học trong phân tích, phát hiện nguy cơ mã độc là một giải pháp thành công, cần phát triển trong tương lai.
  1. **HƯỚNG PHÁT TRIỂN**

Ngoài phân tích tĩnh ta cần phải triển khai thêm mô hình phân tích động để hỗ trợ cho nhau trong việc tìm ra mã độc một cách chính xác nhất. Bằng phương pháp phân tích động ta có thể thấy được các hành vi của mã độc, còn có sử dụng lưu lượng pin, tần suất cpu. Do việc phân tích tĩnh cũng chưa chắc đã tìm ra virus vì công nghệ càng hiện đại sẽ xuất hiện các loại virus tấn công khác nhau đa dạng nên ta phải phân tích động để hiểu thêm về loại malware đó.

Ta cũng cần mở rộng với hướng tiếp cận bằng các thuật toán phát hiện mã độc khác thuộc nhóm học có giám sát, học không giám sát, phân tích động. Từ đó, xây dựng các hệ thống có khả năng phát hiện nguy cơ mã độc một cách chính xác, đa dạng.

**TÀI LIỆU THAM KHẢO**

1. [Triada: truly scary malware for Android | Kaspersky official blog](https://www.kaspersky.com/blog/triada-trojan/11481/)
2. [Malware Android Analysis (Part 1) - Trung tâm An toàn thông tin - VNPT](https://sec.vnpt.vn/2019/03/malware-android-analysis-part-1/)
3. [Kaspersky: Triada là trojan nguy hiểm nhất trên hệ Android | WhiteHat.vn](https://whitehat.vn/threads/kaspersky-triada-la-trojan-nguy-hiem-nhat-tren-he-android.6669/)
4. <https://ngocminhtran.com/2018/11/21/thu-vien-machine-learning-cho-nha-phat-trien-net/>
5. <https://viblo.asia/p/phan-tich-vai-mau-android-malware-tu-xua-ORNZq1knZ0n>
6. <https://xuanthulab.net/cac-thanh-phan-cua-mot-ung-dung-android.html>
7. <https://www.kaspersky.com/resource-center/threats/mobile>
8. <https://ichi.pro/vi/gioi-thieu-ve-ma-bytecode-cua-java-ma-ban-khong-biet-la-minh-can-37818474908344>
9. <https://www.netmotionsoftware.com/blog/security/mobile-malware-threats>
10. <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON?retiredLocale=vi>
11. <https://wpf-tutorial.com/vi/4/b%E1%BA%AFt-d%E1%BA%A7u/hello-wpf-/>
12. <https://jquery.com/>
13. <https://docs.microsoft.com/vi-vn/dotnet/machine-learning/how-to-guides/train-machine-learning-model-ml-net>
14. <https://www.kaspersky.com/blog/triada-trojan/11481/>
15. <https://source.android.com/devices/tech/dalvik>
16. <https://manifestsecurity.com/android-application-security-part-3/>
17. <https://developers.google.com/android/play-protect/phacategories>
18. <https://sectools.org/>