


THÔNG TIN CHUNG CỦA BÁO CÁO

- Link YouTube video của báo cáo (tối đa 5 phút): https://youtu.be/rEwJ8I_Qfes
- Link slides (dạng .pdf đặt trên Github):
https://github.com/vuongthinh209/CS2205.APR2023/blob/main/220201025%20-%20Android_Malware_Detection_Using_Federated_Learning%20-%20Slide.pdf

<ul style="list-style-type: none">• Họ và Tên: Nguyễn Vương Thịnh• MSSV: 220201025 	<ul style="list-style-type: none">• Lớp: CS2205.APR2023• Tự đánh giá (điểm tổng kết môn): 8/10• Số buổi vắng: 1• Số câu hỏi QT cá nhân: 8• Số câu hỏi QT của cả nhóm: 5• Link Github: https://github.com/vuongthinh209/CS2205.APR2023
--	--

ĐỀ CƯƠNG NGHIÊN CỨU

TÊN ĐỀ TÀI (IN HOA)

PHÁT HIỆN MÃ ĐỘC ANDROID DÙNG FEDERATED LEARNING

TÊN ĐỀ TÀI TIẾNG ANH (IN HOA)

ANDROID MALWARE DETECTION WITH FEDERATED LEARNING

TÓM TẮT (Tối đa 400 từ)

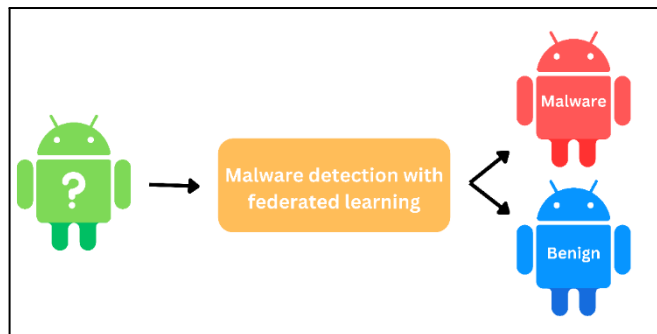
Những năm gần đây, song song với sự bùng nổ các ứng dụng tiện ích trên hệ điều hành Android là sự phát triển mạnh mẽ của các phần mềm độc hại (Malware), và chúng đang dần trở thành mối đe dọa đến sự an toàn và bảo mật thông tin của người dùng. Theo nhóm nghiên cứu [1], trong năm 2021, đã có 3,48 triệu ứng dụng Android tải về và xuất hiện 3,36 triệu phần mềm Android độc hại. Hơn nữa, các phương pháp máy học (Machine Learning) phát hiện mã độc Android thường tập trung xử lý dữ liệu trên một máy chủ. Điều này gây ra mối lo ngại về bảo mật dữ liệu cá nhân vì phải truyền tải dữ liệu nhạy cảm của người dùng lên máy chủ trung tâm. Để giải quyết thách thức về bảo mật, nghiên cứu này đề xuất kỹ thuật học liên kết (Federated Learning) để phát hiện phần mềm độc hại trên Android.

GIỚI THIỆU (Tối đa 1 trang A4)

Hệ điều hành Android ra đời vào năm 2008 và ngày càng phát triển, cho đến tận bây giờ, nó vẫn là hệ điều hành phổ biến nhất đối với các thiết bị di động thông minh [2]. Tuy nhiên, điều này cũng dẫn đến sự gia tăng đáng kể của các phần mềm độc hại nhắm vào nền tảng Android. Mã độc Android đã và đang gây ra mối nguy hại đối với sự bảo mật thông tin người dùng, tổ chức và cả sự an toàn của hệ sinh thái di động. Các kỹ thuật máy học truyền thống chỉ tập

trung dữ liệu vào một máy để xử lý và phát hiện mã độc Android. Cụ thể là dữ liệu từ các thiết bị cá nhân được gửi đến một máy chủ trung tâm để phân tích và phát hiện mã độc nếu có. Tuy nhiên, phương pháp này có thể xâm phạm đến quyền bảo mật riêng tư của người dùng vì vi phạm truy cập dữ liệu cá nhân trái phép.

Nhằm giải quyết vấn đề này, đề cương nghiên cứu đề xuất phương pháp mới là ứng dụng federated learning cho việc phát hiện mã độc Android.



Hình 1. Input và Output của chương trình phát hiện phần mềm độc hại trên nền tảng Android

Hình trên thể hiện:

- Input: một phần mềm hệ điều hành Android
- Output: kết quả xác nhận phần mềm có độc hại (Malware) hay không độc hại (Benign)

Federated learning là một kỹ thuật học máy phi tập trung để huấn luyện các mô hình máy học. Điều này có nghĩa là thay vì gửi dữ liệu nhạy cảm từ thiết bị người dùng lên một máy chủ trung tâm, Federated learning cho phép đào tạo mô hình học máy cục bộ trên từng thiết bị cá nhân, và chỉ chia sẻ thông tin cập nhật từ các mô hình học máy cục bộ đến máy chủ để tổng hợp. Phương pháp này đảm bảo quyền riêng tư và bảo mật dữ liệu, vì dữ liệu cá nhân vẫn nằm trên các thiết bị của người dùng. Chính vì lý do này, Federated learning ngày càng được ứng dụng rộng rãi ở những lĩnh vực cần sự bảo mật cao như y tế [3], internet vạn vật (IoT) [4] và an ninh mạng [5].

MỤC TIÊU (Viết trong vòng 3 mục tiêu, lưu ý về tính khả thi và có thể đánh giá được)

- Xây dựng mô hình phát hiện mã độc Android sử dụng kỹ thuật Federated learning
- Giải quyết mối lo ngại liên quan vi phạm quyền riêng tư, bảo mật người dùng do huấn luyện mô hình máy học trên dữ liệu tập trung

NỘI DUNG VÀ PHƯƠNG PHÁP *(Viết nội dung và phương pháp thực hiện để đạt được các mục tiêu đã nêu)*

Nội dung 1: Tổng quan về phần mềm mã độc Android

- Tìm hiểu về cấu trúc cấu tạo phần mềm Android
- Tìm hiểu các phương tấn công của các mã độc Android và cách phân loại chúng

Nội dung 2: Tổng quan về bài toán phát hiện mã độc Android bằng học máy truyền thống

- Khảo sát các công trình liên quan trong và ngoài nước có sử dụng phương pháp máy học để phát hiện mã độc Android
- Phân tích ưu và nhược điểm của các phương pháp đó

Nội dung 3: Tổng quan về Federated learning

- Tìm hiểu cách thức vận hành của Federated learning
- Tìm hiểu các bài toán máy học có ứng dụng kỹ thuật Federated learning

Nội dung 4: Xây dựng bộ dữ liệu huấn luyện các mô hình học máy

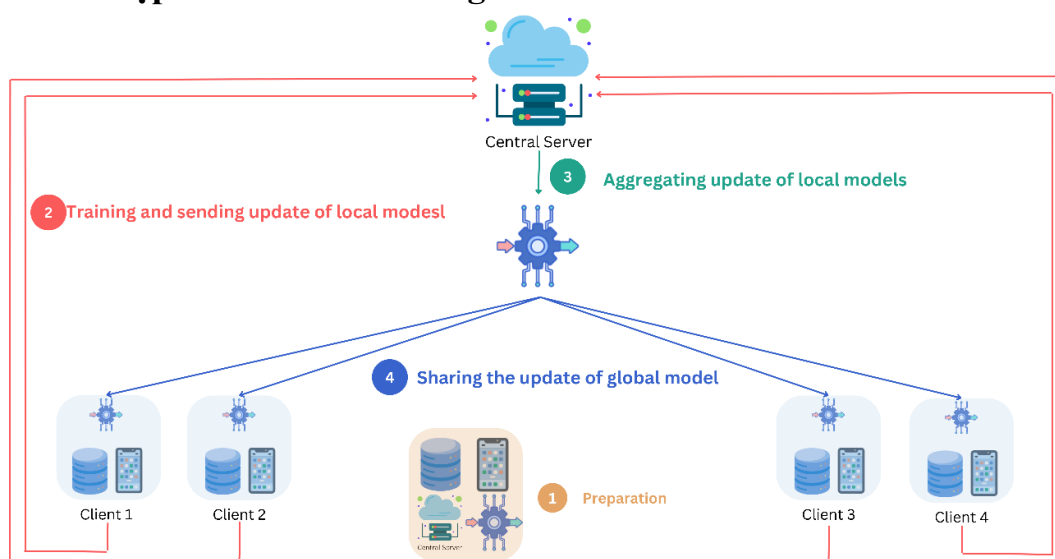
- Bộ dữ liệu (10.000 mẫu) gồm 2 loại phần mềm Android là:
 - Lớp độc hại (Malware Class) (5000 mẫu)
 - Lớp không độc hại (Benign Class) (5000 mẫu)
- Cách thu thập dữ liệu
 - Truy cập vào các nguồn dataset sẵn có trong các nghiên cứu
 - Liên hệ các nhà cung cấp bảo mật và các công ty virus
- Đảm bảo tính độc hại của phần mềm
 - Sử dụng phần mềm chống virus (Bkav) hoặc công cụ phân tích malware (Wireshark) để xác nhận phần mềm mang tính độc hại, tránh kết quả dương tính giả
- Lưu trữ và xử lý dữ liệu
 - Sử dụng máy ảo (virtual machine) hoặc hộp cát (sandbox) để lưu trữ và phân tích dữ liệu, tránh malware gây tổn hại đến hệ thống thực.
- Rút trích đặc trưng

- Rút trích các đặc trưng phần mềm Android để nắm bắt hành vi và đặc điểm của chúng
- Đặc trưng được rút trích bao gồm permissions, API calls, system calls, và network activities.

Nội dung 5: Thiết lập các loại mô hình máy học

- Mô hình Decision Trees [6]: Decision Trees (Cây quyết định) sẽ được huấn luyện từ các đặc trưng trích xuất và xây dựng cấu trúc cây để nắm bắt ranh giới quyết định giữa Malware và Benign
- Mô hình Random Forests [7]: Random Forests (Rừng ngẫu nhiên) là tập hợp nhiều Decision Trees. Mỗi một cây quyết định sẽ được huấn luyện trên một tập con ngẫu nhiên và quyết định cuối cùng sẽ được đưa ra bằng số biểu quyết của các cây quyết định. Random Forests sẽ giảm thiểu hiện tượng overfitting ở Decision Trees. Overfitting là hiện tượng khi model đạt hiệu suất rất tốt trên tập train nhưng lại kém ở tập test.
- Mô hình Support Vector Machines (SVM) [8]: SVM là một thuật toán nhị phân tìm một siêu phẳng (hyperplane) để phân tách các điểm dữ liệu trong một không gian đặc trưng nhiều chiều.
- Mô hình Naïve Bayes [9]: Naïve Bayes là mô hình phân loại xác suất dựa trên định lý Bayes. Nó giả định rằng các đặc trưng là độc lập có điều kiện, đơn giản hóa mô hình nhưng vẫn hiệu quả về mặt tính toán

Nội dung 6: Thiết lập Federated learning framework



Hình 2. Federated learning framework

- **Bước 1.** Chuẩn bị cần thiết
 - **Bước 1.1.** Mô phỏng môi trường Federated learning gồm 4 thiết bị Android (clients), 1 máy chủ (server)
 - **Bước 1.2.** Khởi tạo các mô hình máy học (model) ở server và clients. Trong đó, một model ở server có nhiệm vụ tổng hợp các số liệu được gửi về từ quá trình huấn luyện các models ở các clients
 - **Bước 1.3.** 90% bộ dữ liệu được chia đều ở 3 clients và 10% sẽ dành cho đánh giá hiệu suất của mô hình ở server. Cụ thể, ở mỗi client sẽ được cung cấp 1.500 mẫu malware và 1.500 mẫu benign. Riêng mô hình máy học ở server sẽ được đánh giá kết quả trên 1000 mẫu (gồm 500 malware và 500 benign).
 - **Bước 1.4.** Triển khai các giao thức liên lạc (protocols) như SSL (Secure Sockets Layer) hoặc SMPC (Secure Multi-Party Computation) để đảm bảo quyền riêng tư trong quá trình chia sẻ thông tin giữa clients và server.
- **Bước 2.** Sử dụng bộ dữ liệu đã được cấp sẵn ở mỗi client để tiến hành huấn luyện mô hình máy học
- **Bước 3.** Model ở server sẽ cập nhật các thông tin từ quá trình huấn luyện model ở các clients.
 - Thông tin cập nhật gồm các tham số (parameter) hoặc gradient.
 - Model ở server sẽ sử dụng thuật toán FedSGD (Federated stochastic gradient descent) hoặc FedAvg (FedAvg) để tổng hợp thông tin và cập nhật parameter hoặc gradient cho chính nó.
 - Hiệu suất của model ở server sẽ được đánh giá trên 1000 mẫu và trên các thang đo Accuracy, Recall, Precision, và F1-score.
- **Bước 4.** Model ở server sẽ gửi bản cập nhật của mình (parameters hoặc gradients) đến các clients và model ở clients sẽ sử dụng bản cập nhật cho lần training tiếp theo
- **Bước 5.** Tiếp tục lặp lại từ bước 2 đến 4 cho đến khi kết quả đánh giá model ở server hội tụ trên các thang đo.

Nội dung 7: Thực nghiệm và đánh giá kết quả

- Sử dụng Python để lập trình, ngoài ra tận dụng TFF (TensorFlow Federated) để thực hiện **nội dung 6 - Thiết lập Federated learning framework**
- Thực nghiệm Federated learning framework với mỗi loại mô hình máy học ở **nội dung 5 – thiết lập các loại mô hình máy học**
- Đánh giá và so sánh kết quả đạt của từng loại mô hình trên các thang đo Accuracy, Recall, Precision, và F1-score.

❖ Accuracy là tỉ lệ giữa số mẫu được model dự đoán đúng (T) và tổng số mẫu trong dữ liệu test (T+F)

$$Accuracy = \frac{T}{T+F}$$

❖ Precision là tỉ lệ giữa số mẫu của một class “dương tính” được model dự đoán đúng (TP) và tổng số mẫu được model dự đoán là class “dương tính” (TP+FP)

$$Precision = \frac{TP}{TP+FP}$$

❖ Recall là tỉ lệ giữa số mẫu của một class “dương tính” được model dự đoán đúng (TP) và tổng số mẫu của class “dương tính” trong dữ liệu test (TP+FN)

$$Recall = \frac{TP}{TP+FN}$$

❖ F1-score là số dung hòa giữa Recall và Precision

$$\frac{2}{F1} = \frac{1}{Precision} + \frac{1}{Recall}$$

KẾT QUẢ MONG ĐỢI (*Viết kết quả phù hợp với mục tiêu đặt ra, trên cơ sở nội dung nghiên cứu ở trên*)

- Một mô hình phát hiện mã độc trên hệ điều hành Android sử dụng kỹ thuật học liên kết (Federated learning)
- Một bảng số liệu so sánh kết quả giữa các loại mô hình máy học sử dụng Federated learning để phát hiện mã độc Android

TÀI LIỆU THAM KHẢO (*Định dạng DBLP*)

1. Zakaria Sawadogo, Gervais Mendy, Jean Marie Dembele, Samuel Ouya: Android malware detection: Investigating the impact of imbalanced data-sets on the performance of machine learning models. *ICACT 2022*: 435-441
2. Huijuan Zhu, Huahui Wei, Liangmin Wang, Zhi-cheng Xu, Victor S. Sheng: An effective end-to-end android malware detection method. *Expert Syst. Appl.* 218: 119593 (2023)
3. Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, Björn M. Eskofier: Federated Learning for Healthcare: Systematic Review and Architecture Proposal. *ACM Trans. Intell. Syst. Technol.* 13(4): 54:1-54:23 (2022)
4. Tuo Zhang, Lei Gao, Chaoyang He, Mi Zhang, Bhaskar Krishnamachari, Amir Salman Avestimehr: Federated Learning for the Internet of Things: Applications, Challenges, and Opportunities. *IEEE Internet Things Mag.* 5(1): 24-29 (2022)
5. Mohamed Amine Ferrag, Othmane Friha, Djallel Hamouda, Leandros A. Maglaras, Helge Janicke: Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* 10: 40281-40306 (2022)
6. Aqil Zulkifli, Isredza Rahmi A. Hamid, Wahidah Md Shah, Zubaile Abdullah: Android Malware Detection Based on Network Traffic Using Decision Tree Algorithm. *SCDM* 2018: 485-494
7. Mohammed S. Alam, Son Thanh Vuong: Random Forest Classification for Detecting Android Malware. *GreenCom/iThings/CPScom* 2013: 663-669
8. Tanuvir Singh, Fabio Di Troia, Corrado Aaron Visaggio, Thomas H. Austin, Mark Stamp: Support vector machines and malware detection. *J. Comput. Virol. Hacking Tech.* 12(4): 203-212 (2016)
9. Fengjun Shang, Yalin Li, Xiaolin Deng, Dexiang He: Android malware detection method based on naive Bayes and permission correlation algorithm. *Clust. Comput.* 21(1): 955-966 (2018)