**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN**

---------



**BÁO CÁO ĐỒ ÁN**

**TÌM HIỂU GAN ĐỂ PHÁT SINH MÃ ĐỘC PE**

**Sinh viên thực hiện:**

Nguyễn Tấn Dũng - 21521977

Bùi Minh Huy - 21520910

**Giảng viên:**

Thầy Nguyễn Tấn Cầm

Thành phố Hồ Chí Minh, tháng 11 năm 2023

**TÌM HIỂU GAN ĐỂ PHÁT SINH MÃ ĐỘC PE**

|  |  |
| --- | --- |
| Nguyễn Tấn Dũng  Đại học Công nghệ thông tin  21521977@gm.uit.edu.vn | Bùi Minh Huy  Đại học Công nghệ thông tin  21520910@gm.uit.edu.vn |

**Nghiên cứu này tập trung vào Generative Adversarial Networks (GANs) và ứng dụng chúng trong việc tạo mã độc PE (Portable Executable). GANs đã chứng minh khả năng tạo ra dữ liệu mới giữ tính chất tương tự như dữ liệu huấn luyện. Chúng tôi sử dụng GANs để phát sinh mã độc PE, một dạng mã độc phổ biến trong môi trường Windows. Nghiên cứu giúp hiểu rõ hơn về quá trình đối kháng giữa mô hình sinh (generator) và mô hình phân biệt (discriminator) trong tạo ra mã độc. Kết quả nghiên cứu cung cấp cái nhìn sâu sắc về tính chất và cách thức hoạt động của mã độc được tạo ra bằng GANs. Đồng thời, cho thấy khả năng của GAN trong mô phỏng đặc điểm của các mẫu đã tồn tại, đặt ra thách thức cho hệ thống phòng ngừa truyền thống. Nghiên cứu không chỉ phân tích khả năng tấn công của GAN mà còn đề xuất biện pháp cụ thể để nâng cao khả năng phòng ngừa của hệ thống an ninh. Kết quả có thể hỗ trợ ngành an ninh mạng hiểu rõ hơn về rủi ro của GAN trong tạo ra mã độc PE và phát triển chiến lược đối mặt với thách thức ngày càng nâng cao**.

**CHƯƠNG 1 GIỚI THIỆU**

Chủ đề nghiên cứu xoay quanh sự ứng dụng của Mạng GAN (Generative Adversarial Network) trong việc tạo ra mã độc PE (Portable Executable), tập trung vào tăng cường hiểu biết về khả năng tấn công và phòng ngừa trong lĩnh vực an ninh mạng. Điều này đặt ra từ một động lực ngày càng lớn về nhu cầu đối mặt với các thách thức ngày càng phức tạp từ các mối đe dọa mạng, đặc biệt là liên quan đến mã độc PE.

Nghiên cứu này bắt đầu bằng việc phác thảo cơ sở lý thuyết về Mạng GAN và đặc điểm cụ thể của mã độc PE. Làm rõ vấn đề với nguyên tắc làm thế nào GAN có thể được áp dụng để tạo ra các biểu diễn mã độc PE mới, đồng thời tập trung vào những yếu điểm và thách thức tiềm ẩn trong quá trình này.

Chúng tôi sẽ tiến hành một quy trình huấn luyện chặt chẽ trên một tập dữ liệu đa dạng gồm mã độc PE đã biết và chưa biết trước, với mục tiêu tối ưu hóa khả năng mô phỏng và tạo ra các biểu diễn có tính giới thiệu cao. Chúng tôi cũng sẽ đề xuất và thử nghiệm các biện pháp cụ thể nhằm tăng cường khả năng phòng ngừa, bao gồm cả sử dụng kỹ thuật tăng cường dữ liệu và tinh chỉnh siêu tham số để đạt được hiệu suất tốt nhất.

Trong quá trình nghiên cứu, chúng tôi sẽ chi tiết giải thích cách GAN có thể tạo ra các biểu diễn mã độc PE và làm thế nào chúng có thể mô phỏng đa dạng các đặc điểm của các mẫu đã tồn tại. Đồng thời, chúng tôi sẽ đề xuất các chiến lược cụ thể để cải thiện khả năng phòng ngừa và đối phó với thách thức ngày càng nâng cao từ mô hình GAN trong ngữ cảnh an ninh mạng.

Nhìn chung, nghiên cứu này không chỉ giúp hiểu rõ hơn về khả năng tấn công của GAN trong tạo ra mã độc PE mà còn đề xuất những biện pháp cụ thể để củng cố hệ thống an ninh mạng, đặt ra cơ sở cho sự hiểu biết và ứng phó với mối đe dọa ngày càng tiên tiến từ phía mã độc PE.

**CHƯƠNG 2 BÀI VIẾT LIÊN QUAN**

2.1. Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN [1]

Bài viết đề xuất một thuật toán mới có tên MalGAN để tạo ra các ví dụ về phần mềm độc hại đối nghịch có thể vượt qua các hệ thống phát hiện phần mềm độc hại dựa trên máy học, giả định rằng những kẻ tấn công chỉ biết các tính năng được hệ thống phát hiện sử dụng chứ không biết chi tiết về mô hình hoặc tham số.

Bài viết sử dụng mạng đối nghịch tổng quát (GAN) để sinh động các mẫu đối nghịch theo phản hồi của bộ phát hiện hộp đen và một bộ phát hiện thay thế để phù hợp với bộ phát hiện hộp đen và cung cấp thông tin gradient để huấn luyện bộ tạo. Bài viết đánh giá hiệu suất của MalGAN trên các thuật toán học máy khác nhau, chẳng hạn như rừng ngẫu nhiên, hồi quy logistic, cây quyết định, máy vectơ hỗ trợ, perceptron nhiều lớp và tập hợp dựa trên biểu quyết.

Dữ liệu thử nghiệm: Tác giả sử dụng một tập dữ liệu gồm 180 nghìn chương trình được tải về từ một trang web chia sẻ chương trình, trong đó khoảng 30% là mã độc. Tác giả sử dụng 160 API hệ thống làm đặc trưng nhị phân cho mỗi chương trình.

Thuật toán phát hiện mã độc: Tác giả thử nghiệm MalGAN với nhiều thuật toán phát hiện mã độc khác nhau, bao gồm rừng ngẫu nhiên (RF), hồi quy logistic (LR), cây quyết định (DT), máy vector hỗ trợ (SVM), perceptron nhiều lớp (MLP), và bầu chọn dựa trên các thuật toán này (VOTE).

Cách chia tập dữ liệu: Tác giả sử dụng hai cách chia tập dữ liệu khác nhau để đánh giá khả năng chuyển giao của các mẫu đối kháng sinh ra bởi MalGAN. Cách thứ nhất là chia tập dữ liệu thành 80% làm tập huấn luyện và 20% làm tập kiểm tra. MalGAN và thuật toán phát hiện mã độc chia sẻ cùng một tập huấn luyện. Cách thứ hai là chia tập dữ liệu thành ba phần: 40% làm tập huấn luyện cho MalGAN, 40% làm tập huấn luyện cho thuật toán phát hiện mã độc, và 20% làm tập kiểm tra.

Tham số và phương pháp tối ưu: Tác giả sử dụng Adam làm phương pháp tối ưu hóa cho cả bộ sinh và bộ phát hiện thay thế của MalGAN. Tác giả điều chỉnh các tham số của MalGAN trên tập xác thực, bao gồm kích thước của vector nhiễu, số lượng nơ-ron trong các lớp ẩn, tốc độ học, và số lượng epoch. Tác giả chọn epoch có tỷ lệ phát hiện mã độc thấp nhất trên tập xác thực để kiểm tra hiệu quả của MalGAN.

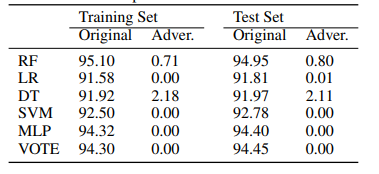
Kết quả cho thấy MalGAN có thể giảm tỉ lệ phát hiện mã độc xuống gần bằng không trên cả tập huấn luyện và tập kiểm tra, đồng thời có khả năng chuyển giao các mẫu đối kháng giữa các mô hình khác nhau.

A table with numbers and text

Description automatically generated

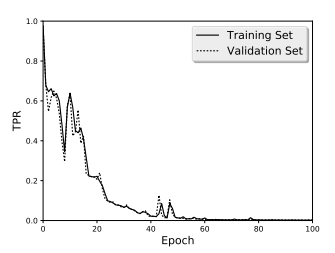
Bảng 1: True positive rate (tính bằng phần trăm) trên các mẫu ban đầu và các ví dụ đối nghịch khi MalGAN và black-box detector được huấn luyện trên cùng một tập huấn luyện.

Bảng 1 so sánh tỷ lệ phát hiện đúng (TPR) của các thuật toán phát hiện mã độc trên các mẫu ban đầu và các mẫu đối kháng khi MalGAN và thuật toán phát hiện mã độc được huấn luyện trên cùng một tập dữ liệu. Kết quả cho thấy MalGAN có thể giảm TPR xuống gần bằng không cho hầu hết các thuật toán phát hiện mã độc, nghĩa là MalGAN có thể tạo ra các mẫu đối kháng có thể lừa được các thuật toán phát hiện mã độc.



Bảng 2: True positive rate (tính bằng phần trăm) trên các mẫu ban đầu và các ví dụ đối nghịch khi MalGAN và black-box detector được huấn luyện trên các tập huấn luyện khác nhau.

Bảng 2 so sánh TPR của các thuật toán phát hiện mã độc trên các mẫu ban đầu và các mẫu đối kháng khi MalGAN và thuật toán phát hiện mã độc được huấn luyện trên các tập dữ liệu khác nhau. Kết quả cho thấy MalGAN vẫn có thể giảm TPR xuống gần bằng không cho một số thuật toán phát hiện mã độc, và chỉ tăng lên một chút cho một số thuật toán khác. Điều này cho thấy MalGAN có khả năng chuyển giao các mẫu đối kháng cho các thuật toán phát hiện mã độc khác nhau.



Hình 1: Sự thay đổi của TPR trong training set và validation set theo thời gian.

Hình 1 cho thấy sự thay đổi của TPR trên tập huấn luyện và tập kiểm tra theo thời gian khi huấn luyện MalGAN. Thuật toán phát hiện mã độc được sử dụng là rừng ngẫu nhiên. Đồ thị cho thấy TPR hội tụ về gần bằng không xung quanh lần lặp thứ 40, nhưng đồ thị có chút rung lắc, không phải là một đường cong mượt mà. Điều này phản ánh thực tế rằng việc huấn luyện GAN thường không ổn định.

Bài viết cũng so sánh MalGAN với phương pháp dựa trên gradient để tạo mẫu đối kháng, và cho thấy MalGAN có hiệu quả cao hơn và linh hoạt hơn trong việc đối phó với các phương pháp phòng thủ dựa trên việc huấn luyện lại mô hình phát hiện mã độc.

2.2. Generative adversarial networks and image-based malware detection [2]

Bài viết này đi sâu vào ứng dụng GAN, đặc biệt là MalGAN, trong bối cảnh phân tích hình ảnh phần mềm độc hại. Các tác giả khám phá tiềm năng của MalGAN trong việc tạo ra các hình ảnh phần mềm độc hại đối nghịch có thể vượt qua các hệ thống phát hiện phần mềm độc hại dựa trên hình ảnh. Họ chứng minh tính hiệu quả của MalGAN bằng cách tạo ra các hình ảnh phần mềm độc hại đối nghịch có thể tránh được sự phát hiện thành công của nhiều trình phát hiện phần mềm độc hại dựa trên hình ảnh.

2.3. Bringing a GAN to a Knife-Fight: Adapting Malware Communication to Escape Network Traffic Analysis. [3]

Bài viết này đề xuất việc sử dụng GAN để tạo ra các mẫu lưu lượng truy cập mạng thực tế bắt chước giao tiếp của phần mềm độc hại. Lưu lượng được tạo có thể được sử dụng để phá vỡ các hệ thống phân tích lưu lượng mạng và tăng cường khả năng tàng hình của phần mềm độc hại. Các tác giả chứng minh tính hiệu quả của phương pháp tiếp cận của họ bằng cách tạo ra lưu lượng truy cập mạng đối nghịch có thể tránh được sự phát hiện thành công của các nhà phân tích lưu lượng mạng khác nhau.

2.4. DeepDetectNet vs RLAttackNet: An adversarial method to improve PE malware detection [4]

Bài viết này tận dụng khái niệm GAN để phát triển DeepDetectNet, một mô hình phát hiện phần mềm độc hại PE tĩnh. DeepDetectNet nắm bắt một cách hiệu quả các đặc điểm của cả tệp PE lành tính và độc hại, cho phép nó đạt được hiệu suất phát hiện phần mềm độc hại mạnh mẽ. Các tác giả so sánh DeepDetectNet với RLAttackNet, một phương pháp phát hiện phần mềm độc hại dựa trên GAN khác và chứng minh hiệu suất vượt trội của DeepDetectNet.

2.5. Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN [5]

Bài viết này tập trung vào việc tạo các ví dụ về phần mềm độc hại PE đối nghịch bằng cách sử dụng GAN để vượt qua các hệ thống phát hiện phần mềm độc hại dựa trên deep learning. Các tác giả đề xuất một chiến lược huấn luyện đối thủ mới giúp tránh bị phát hiện một cách hiệu quả và cải thiện khả năng chuyển giao của các cuộc tấn công đối nghịch. Họ chứng minh tính hiệu quả của phương pháp tiếp cận của mình bằng cách tạo ra các ví dụ về phần mềm độc hại PE đối nghịch có thể trốn tránh thành công sự phát hiện của nhiều trình phát hiện phần mềm độc hại dựa trên công nghệ học sâu.

2.6. MAB-Malware: A Reinforcement Learning Framework for Attacking Static Malware Classifiers [6]

Bài viết này đề xuất MAB-Malware, một khung dựa trên học tập tăng cường để tạo các mẫu phần mềm độc hại PE đối nghịch. MAB-Malware tạo ra các ví dụ đối nghịch một cách hiệu quả có thể trốn tránh sự phát hiện của nhiều trình phát hiện phần mềm độc hại khác nhau. Các tác giả chứng minh tính hiệu quả của MAB-Malware bằng cách tạo ra các ví dụ về phần mềm độc hại PE đối nghịch có thể trốn tránh thành công sự phát hiện của nhiều công cụ phát hiện phần mềm độc hại.

2.7. CNN and GANbased classification of malicious code families: A code visualization approach [7]

Bài viết này trình bày một mô hình phân loại mới cho các họ mã độc kết hợp Convolutional Neural Networks (CNN) với GAN. Mô hình đề xuất phân loại hiệu quả mã độc thành các họ khác nhau dựa trên mẫu mã của chúng. Các tác giả chứng minh tính hiệu quả của mô hình của họ bằng cách so sánh nó với các phương pháp phân loại hiện có và đạt được hiệu suất vượt trội.

2.8. Arms Race in Adversarial Malware Detection: A Survey [8]

Bài viết cung cấp một tổng quan về cuộc chạy đua vũ trang giữa các cuộc tấn công và phòng thủ phần mềm độc hại đối kháng. Các tác giả phân loại các cuộc tấn công đối kháng thành các loại khác nhau và thảo luận về các chiến lược phòng thủ hiện có. Họ cũng thảo luận về các vấn đề và thách thức trong lĩnh vực này.

2.9. A survey on practical adversarial examples for malware classifiers [9]

Bài viết tập trung vào việc nghiên cứu các ví dụ đối địch thực tế cho các bộ phân loại malware. Bài báo nhấn mạnh rằng các giải pháp dựa trên máy học đã hữu ích trong việc giải quyết các vấn đề liên quan đến việc phát hiện và phân loại malware. Tuy nhiên, các mạng nơ-ron sâu đã được phát hiện là dễ bị tấn công bởi các ví dụ đối địch, tức là các mẫu dữ liệu được tinh chỉnh nhằm đánh lừa bộ phân loại. Bài báo này cũng tìm hiểu về các cuộc tấn công đối địch thực tế đối với các bộ phân loại malware. Nó giới thiệu về các kiến thức cơ bản về ví dụ đối địch và so sánh giữa mô hình tấn công đối địch lý thuyết và mô hình tấn công đối địch thực tế. Bài báo cũng trình bày các phương pháp tấn công đối địch thực tế hiện có.

2.10. GAN-Based Training for Binary Classifier [10]

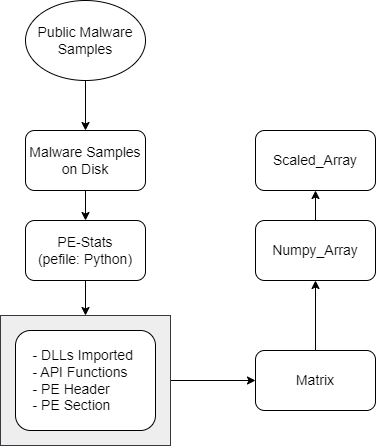
Bài viết này khám phá việc sử dụng GAN để đào tạo bộ phân loại nhị phân. Các tác giả chứng minh tính hiệu quả của việc đào tạo dựa trên GAN trong việc cải thiện hiệu suất của các nhiệm vụ phân loại nhị phân. Họ so sánh đào tạo dựa trên GAN với các phương pháp đào tạo truyền thống và cho thấy rằng đào tạo dựa trên GAN có thể dẫn đến những cải thiện đáng kể về độ chính xác của phân loại.

**CHƯƠNG 3 HỆ THỐNG ĐỀ XUẤT**

Phương pháp nghiên cứu của chúng tôi đặt trọng điểm vào việc áp dụng Mạng GAN (Generative Adversarial Network) để tạo ra các biểu diễn mã độc PE (Portable Executable) đa dạng. Để hiểu rõ quy trình và ảnh hưởng của mô hình, chúng tôi thực hiện các bước, mô tả chi tiết dữ liệu và quá trình tiền xử lý trong Hình 1.

1. Tiền xử lý dữ liệu

Tập dữ liệu của chúng tôi được thu thập theo hai bước. Hình 1 tóm tắt những bước này. Trong bước đầu tiên, chúng tôi thu thập dữ liệu từ Cơ sở dữ liệu MalwareBazaar bằng cách sử dụng API của nó. Chỉ có các tệp PE (Portable Executable) của Windows được chọn làm mục tiêu trong các cuộc gọi API, và hơn 20,000 mẫu đã được tải về. Chúng tôi sử dụng thư viện pefile 2 của Python để trích xuất thống kê hoặc đặc điểm PE từ những mẫu đó. Những mẫu có giá trị không chính xác hoặc thiếu trong tiêu đề PE đã bị loại bỏ. Tương tự, chúng tôi cũng loại bỏ các mẫu có sự ẩn dụ mã nguồn (code obfuscation). Sau khi loại bỏ những mẫu không mong muốn, chúng tôi có tổng cộng 18,500 mẫu trong bộ dữ liệu của mình và lưu chúng dưới dạng Comma Separated Values file (.CSV)



Hình 1. Quá trình tiền xử lý dữ liệu

Bước thứ hai là quá trình quan trọng giúp chuẩn bị dữ liệu cho mô hình GAN. Chúng tôi chuyển đổi dữ liệu từ định dạng .csv sang vector một chiều đã được chuẩn hóa để thuận tiện cho việc sử dụng trong mô hình. Chúng tôi cô lập 13500 mẫu cho quá trình huấn luyện mô hình và giữ lại 5000 mẫu để kiểm thử độ hiệu quả của mô hình.

|  |  |  |
| --- | --- | --- |
| Malware Family | Số lượng | Mô tả |
| Generic Malware | 6231 | Là những phần mềm độc hại không thuộc bất kỳ hạng mục cụ thể nào dưới đây. |
| Spyware | 5766 | Nó giám sát hoạt động trên một hệ thống bị nhiễm và đánh cắp thông tin nhạy cảm như mật khẩu và chi tiết ngân hàng. |
| Downloader | 2438 | Hỗ trợ tải về các phần mềm độc hại khác vào một hệ thống bị nhiễm. |
| Ransomware | 2376 | Ngăn chặn người dùng truy cập vào hệ thống hoặc tệp cá nhân của họ và đòi hỏi thanh toán tiền chuộc để khôi phục quyền truy cập. |
| Backdoor | 1740 | Một loại trojan vô hiệu hóa các thủ tục xác thực bình thường để truy cập vào một hệ thống. Nó cung cấp quyền truy cập từ xa vào các tài nguyên đang bị tấn công. |
| Tổng | **18551** |  |

Bảng 1. Phân phối số lượng theo từng họ mã độc

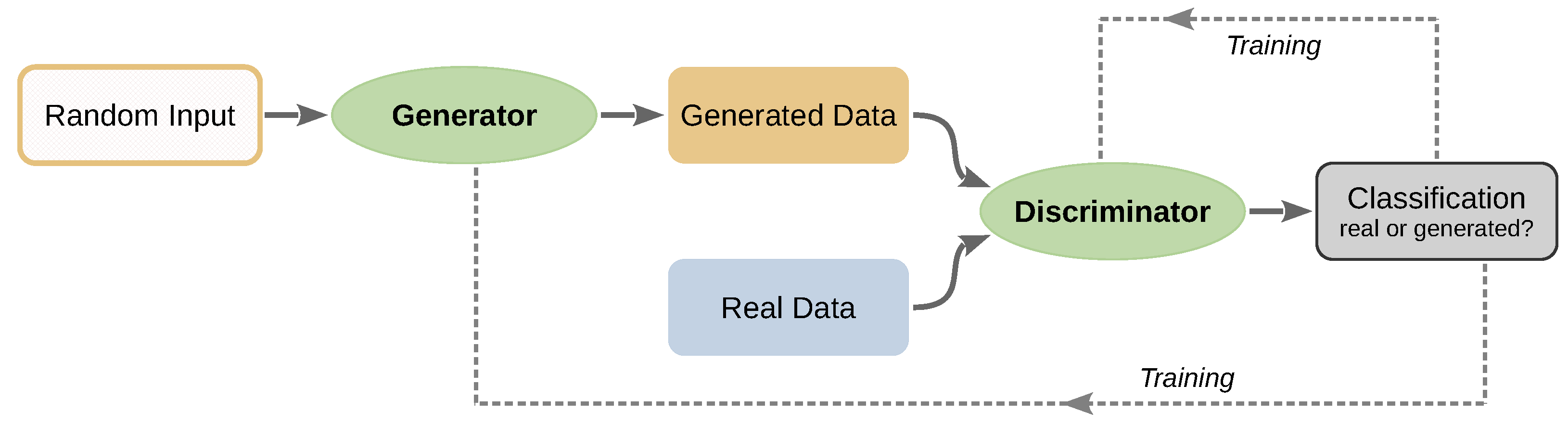
1. GAN model

Mô hình GAN mà chúng tôi triển khai bao gồm hai thành phần chính: mô hình sinh (Generator) và mô hình phân biệt (Discriminator).

**Generator**: Mô hình sinh chịu trách nhiệm tạo ra các biểu diễn mới của mã độc PE. Trong mạng neural này, có bốn lớp tuyến tính (fully connected layers) và ba hàm kích hoạt Leaky ReLU và một hàm kích hoạt Sigmoid. Generator nhận vào một vectơ nhiễu và cố gắng tạo ra một vectơ dữ liệu mới

**Discriminator:** Được thiết kế để đánh giá sự khác biệt giữa mẫu được tạo và mẫu thực tế. Nó cũng có bốn lớp tuyến tính và ba hàm kích hoạt Leaky ReLU, một hàm kích hoạt Sigmoid và một lớp dropout. Discriminator nhận vào một vectơ dữ liệu và cố gắng đưa ra dự đoán về việc liệu vectơ đó là thật hay giả.

**GAN:** Lớp GAN kết hợp Generator và Discriminator để tạo ra và đánh giá dữ liệu giả mạo. Trong hàm forward của GAN, dữ liệu được truyền qua Generator và sau đó truyền qua Discriminator để đánh giá. Quá trình này đặt ra một thách thức giữa Generator và Discriminator, nơi Generator cố gắng tạo ra dữ liệu tốt đến mức làm cho Discriminator khó phân biệt được giữa dữ liệu thật và giả mạo, trong khi Discriminator cố gắng cải thiện khả năng phân biệt giữa chúng.



Hình 2. Quá trình train của GAN model

Trong mô hình GAN trên, chúng tôi sử dụng hàm mất mát nhị phân chéo (BCELoss) của PyTorch để thực hiện đo lường sự sai khác giữa đầu ra dự đoán và nhãn thực tế với công thức sau:

Trong đó:

* N là số lượng mẫu
* là xác suất dự đoán (đầu ra của mô hình), nằm trong khoảng[0,1][0,1].
* là nhãn thực tế (0 hoặc 1).

Trong quá trình huấn luyện, chúng tôi tối ưu hóa mô hình để tạo ra các biểu diễn có tính giới thiệu cao, nhằm mô phỏng nhiều đặc điểm nguy hiểm của các mẫu đã tồn tại. Không chỉ tập trung vào việc tạo ra biểu diễn mới, chúng tôi cũng đảm bảo rằng chúng phản ánh đầy đủ các đặc điểm nguy hiểm của mã độc PE, từ các kỹ thuật che dấu đến các biện pháp né tránh phòng ngừa.

Mục tiêu của chúng tôi không chỉ là tạo ra một mô hình GAN có khả năng tạo ra biểu diễn mã độc PE mới, mà còn là hiểu sâu sắc về khả năng tấn công của nó và đề xuất chiến lược phòng ngừa. Chúng tôi xem xét sự thay đổi trong quá trình huấn luyện, đánh giá độ hiệu quả của mô hình trên tập dữ liệu kiểm thử và điều chỉnh tham số để tối ưu hóa hiệu suất.

**CHƯƠNG 4 KẾT QUẢ THỬ NGHIỆM MÔ HÌNH**

Kết quả của mô hình được đánh giá thông qua theo dõi hàm loss của Discriminator và Generator qua các epochs. Hàm loss này là một phép đo quan trọng về hiệu suất của GAN trong quá trình huấn luyện. Hình 3 thể hiện sự biến động của hai loại loss này qua thời gian.

A graph of a number of different colored lines

Description automatically generated with medium confidence

Hình 3. Biểu đồ loss của Dicriminator và Generator qua các epochs

* Hình 3 thể hiện sự biến động của hàm loss của Discriminator và Generator qua các epochs trong quá trình huấn luyện. Discriminator loss và Generator loss được đánh dấu trên trục y, trong khi trục x biểu thị số lượng epochs.
* Trong giai đoạn đầu tiên của quá trình huấn luyện, Discriminator loss giảm dần, điều này phản ánh việc mô hình Generator ngày càng tạo ra dữ liệu giống thật. Ngược lại, Generator loss tăng, chỉ ra rằng Generator đang cố gắng tạo ra dữ liệu khó phân biệt hơn.
* Trong giai đoạn tiếp theo, Discriminator loss và Generator loss đều hội tụ đến một giá trị ổn định, cho thấy sự cân bằng giữa hai mô hình và chất lượng của mẫu được tạo.

**CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TƯƠNG LAI**

Trong nghiên cứu này, chúng tôi đã tiến hành phân tích và triển khai Mạng GAN để tạo ra các biểu diễn mã độc PE trong bối cảnh an ninh mạng. Qua quá trình nghiên cứu, chúng tôi đã chiêm nghiệm được sức mạnh và tiềm năng đặc biệt của GAN trong việc tạo ra các biểu diễn mới, đặt ra thách thức lớn đối với các hệ thống phòng ngừa truyền thống. Kết quả của chúng tôi cung cấp sự hiểu biết sâu rộng về khả năng tấn công của GAN và đề xuất chiến lược phòng ngừa cụ thể để đối mặt với mối đe dọa từ mã độc PE. Trong tương lai chúng tôi dự định sẽ tối ưu mô hình qua việc tăng cường độ phức tạp của mô hình GAN để đảm bảo rằng nó có khả năng tái tạo các đặc điểm chi tiết và phức tạp của mã độc PE và tối ưu hóa quy trình huấn luyện để giảm thời gian và tài nguyên cần thiết, đặc biệt là khi xử lý các tập dữ liệu lớn.

**TÀI LIỆU THAM KHẢO**  
[1] Generating adversarial malware examples for black-box ... - arxiv.org, https://arxiv.org/pdf/1702.05983.pdf (accessed Dec. 1, 2023).

[2] H. Nguyen, F. Di Troia, G. Ishigaki, and M. Stamp, “Generative Adversarial Networks and Image-Based Malware Classification,” *arXiv.org*, Jun. 08, 2022. https://arxiv.org/abs/2207.00421

[3] M. Rigaki and S. Garcia, “Bringing a GAN to a Knife-Fight: Adapting Malware Communication to Avoid Detection,” *IEEE Xplore*, May 01, 2018. https://ieeexplore.ieee.org/abstract/document/8424635 (accessed Feb. 11, 2021).

[4] Y. Fang, Y. Zeng, B. Li, L. Liu, and L. Zhang, “DeepDetectNet vs RLAttackNet: An adversarial method to improve deep learning-based static malware detection model,” *PLOS ONE*, vol. 15, no. 4, p. e0231626, Apr. 2020, doi: https://doi.org/10.1371/journal.pone.0231626.

[5] W. Hu and Y. Tan, “Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN,” *arXiv:1702.05983 [cs]*, Feb. 2017, Available: https://arxiv.org/abs/1702.05983

[6] W. Song, X. Li, S. Afroz, D. Garg, D. Kuznetsov, and H. Yin, “MAB-Malware: A Reinforcement Learning Framework for Attacking Static Malware Classifiers,” *arXiv.org*, Apr. 29, 2021. https://arxiv.org/abs/2003.03100# (accessed Oct. 25, 2023).

[7] Z. Wang, W. Wang, Y. Yang, Z. Han, D. Xu, and C. Su, “CNN‐ and GAN‐based classification of malicious code families: A code visualization approach,” *International Journal of Intelligent Systems*, vol. 37, no. 12, pp. 12472–12489, Oct. 2022, doi: https://doi.org/10.1002/int.23094.

[8] D. Li, Q. Li, Y. (Fanny) Ye, and S. Xu, “Arms Race in Adversarial Malware Detection: A Survey,” *ACM Computing Surveys*, vol. 55, no. 1, pp. 1–35, Jan. 2023, doi: https://doi.org/10.1145/3484491.

[9] D. Park and B. Yener, “A survey on practical adversarial examples for malware classifiers,” *arXiv.org*, Nov. 19, 2020. https://arxiv.org/abs/2011.05973 (accessed Nov. 27, 2023).

[10] “GAN-Based Training for Binary Classifier,” *kaggle.com*. https://www.kaggle.com/competitions/gan-based-training-for-binary-classifier/overview/timeline (accessed Nov. 27, 2023).

‌

Top of Form