ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

Tel. (+84.0236) 3736949, Fax. (+84.0236) 3842771

Website: <http://dut.udn.vn/khoacntt> , E-mail: cntt@dut.udn.vn



BÁO CÁO

THỰC TẬP TỐT NGHIỆP

NGÀNH CÔNG NGHỆ THÔNG TIN

**PHÁT TRIỂN TƯỜNG LỬA TRÊN HỆ ĐIỀU HÀNH WEBOS**

CÔNG TY THỰC TẬP:

LG

SINH VIÊN : Thân Nguyên Minh Quân

MÃ SINH VIÊN : 102200148

LỚP SH : 20TCLC-DT3

NHÓM HP : 20Nh14

CBHD : Nguyễn Thế Xuân Ly

Đà Nẵng, 01/2025

LỜI CẢM ƠN

Lời đầu tiên, em xin cám ơn công ty LG đã tạo điều kiện để em được thực tập, học hỏi hiểu biết về mảng an ninh mạng, hệ điều hành WebOS.

Tiếp theo, em xin cám ơn thầy Nguyễn Thế Xuân Ly đã tận tình hướng dẫn chi tiết để có thể hoàn thành báo cáo đúng hạn và đầy đủ chính xác.

Em thực sự cảm ơn vì được thực tập LG Electronics Development Vietnam và tin tưởng rằng những kỹ năng và kinh nghiệm có được trong thời gian thực tập sẽ là nền tảng vững chắc cho những nỗ lực trong tương lai của em trong lĩnh vực bảo mật mạng và an toàn thông tin.

Một lần nữa, xin cảm ơn LG vì sự hướng dẫn và trải nghiệm học tập vô giá. Em rất mong có cơ hội hợp tác lần nữa trong tương lai.

**Thân Nguyên Minh Quân**

|  |  |
| --- | --- |
| **LG ELECTRONICS DEVELOPMENT VIETNAM.** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  *Độc lập - Tự do - Hạnh Phúc* |

**PHIẾU ĐÁNH GIÁ KẾT QUẢ THỰC TẬP TỐT NGHIỆP**

*Họ và Tên sinh viên: Thân Nguyên Minh Quân Lớp: 20TCLC-DT3 Nhóm: 20Nh14*

*Cơ quan/Đơn vị thực tập:* LG ELECTRONICS DEVELOPMENT VIETNAM.

*Địa chỉ:* LG CyberSecurity Lab, Trường Đại học Bách khoa - Đại học Đà Nẵng (Địa chỉ: 54 Nguyễn Lương Bằng, Hoà Khánh Bắc, Liên Chiểu, Đà Nẵng)

*Thời gian thực tập từ 25/11/2024. đến 17/01/2025*

*Người hướng dẫn: Nguyễn Thế Xuân Ly Email:* [*nguyenthexuanly@gmail.com*](mailto:nguyenthexuanly@gmail.com)

*Điện thoại:05113736949*

1. **Đánh giá về năng lực chuyên môn**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nội dung đánh giá** | **Xuất sắc** | **Tốt** | **Khá** | **T.Bình** | **Yếu** |
| *Năng lực chuyên môn đáp ứng công việc* |  |  |  |  |  |
| *Hoàn thành các công việc được giao* |  |  |  |  |  |
| *Khả năng sử dụng ngoại ngữ* |  |  |  |  |  |
| *Ứng dụng kết quả thực tập cho cơ quan* |  |  |  |  |  |

1. **Đánh giá về ý thức làm việc**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nội dung đánh giá** | **Xuất sắc** | **Tốt** | **Khá** | **T.Bình** | **Yếu** |
| *Tinh thần, thái độ làm việc* |  |  |  |  |  |
| *Tuân thủ kỷ luật (thời gian làm việc, báo nghỉ…)* |  |  |  |  |  |
| *Giao tiếp, quan hệ với cán bộ, công nhân viên* |  |  |  |  |  |

1. **Đánh giá kết quả công việc**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nội dung đánh giá** | **Xuất sắc** | **Tốt** | **Khá** | **T.Bình** | **Yếu** |
| *Khả năng phân tích thiết kế hệ thống* |  |  |  |  |  |
| *Kỹ năng lập trình* |  |  |  |  |  |
| *Khả năng học hỏi, nắm bắt công nghệ mới* |  |  |  |  |  |

1. **Các nhận xét khác** *(nếu có)*

1. **Điểm đánh giá.** *Ghi bằng số:............/10 Ghi bằng chữ:...................................*

|  |  |
| --- | --- |
| ***Xác nhận của cơ quan/đơn vị thực tập***  *(Ký, ghi rõ họ tên và đóng dấu)* | *Ngày .... tháng ...... năm 20......*  *Người hướng dẫn*  *(Ký và ghi rõ họ tên)* |

NHẬN XÉT CỦA NGƯỜI HƯỚNG DẪN

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

MỤC LỤC

[CHƯƠNG 1: GIỚI THIỆU CƠ QUAN THỰC TẬP 4](#_Toc187845782)

[1.1. TỔNG QUAN VỀ CƠ QUAN THỰC TẬP 4](#_Toc187845783)

[1.2. KẾT CHƯƠNG 5](#_Toc187845784)

[CHƯƠNG 2: NỘI DUNG THỰC TẬP 6](#_Toc187845785)

[2.1. TỔNG QUÁT CÁC CÔNG VIỆC ĐƯỢC GIAO 6](#_Toc187845786)

[2.1.1. WebOS 6](#_Toc187845787)

[2.1.2. Tường lửa trên WebOS 6](#_Toc187845788)

[2.2. CHI TIẾT CÁC NỘI DUNG ĐƯỢC GIAO 7](#_Toc187845789)

[2.3. BÁO CÁO CÁC NHIỆM VỤ CÁ NHÂN 10](#_Toc187845790)

[2.3.1. [Tuần 1] Iptables 10](#_Toc187845791)

[2.3.2. [Tuần 2] Scapy tcpdump testing Firewall 11](#_Toc187845792)

[2.3.3. [Tuần 3] IPv4/IPv6, Multicast, Broadcast, Anycast, Unicast 13](#_Toc187845793)

[2.3.4. [Tuần 4] ULOGD 23](#_Toc187845794)

[2.3.5. [Tuần 5] Nfacct API 27](#_Toc187845795)

[2.3.6. [Tuần 6] Nflog API 32](#_Toc187845796)

[2.4. TỔNG KẾT 36](#_Toc187845797)

[CHƯƠNG 3: TỔNG KẾT 37](#_Toc187845798)

[3.1. KẾT QUẢ TRIỂN KHAI 37](#_Toc187845799)

[3.2. NHẬN XÉT VÀ KẾT CHƯƠNG 37](#_Toc187845800)

**DANH SÁCH HÌNH**

[Hình 1: Hình ảnh về phòng lab LG 5](#_Toc187845600)

[Hình 2: Sơ đồ Sequence Nfacct API 8](#_Toc187845601)

[Hình 3: Sơ đồ Sequence Nflog API 9](#_Toc187845602)

[Hình 4: Kết quả sau khi triển khai Iptables 10](#_Toc187845603)

[Hình 5: Kết quả đổi IP đích khi gửi payload bằng Scapy 13](#_Toc187845604)

[Hình 6: Kết quả gửi bằng Unicast IPv4 15](#_Toc187845605)

[Hình 7: Kết quả gửi bằng BroadCast IPv4 16](#_Toc187845606)

[Hình 8: Kết quả gửi bằng Multicast IPv4 17](#_Toc187845607)

[Hình 9: Thêm thành viên vào nhóm bằng giao thức IGMP 18](#_Toc187845608)

[Hình 10: Xóa thành viên vào nhóm bằng giao thức IGMP 18](#_Toc187845609)

[Hình 11: Config IPv6 18](#_Toc187845610)

[Hình 12: Kết quả gửi Unicast IPv6 19](#_Toc187845611)

[Hình 13: Kết quả gửi Multicast IPv6 20](#_Toc187845612)

[Hình 14: Config IPv6 cho Anycast 20](#_Toc187845613)

[Hình 15: Kết quả gửi Anycast IPv6 21](#_Toc187845614)

[Hình 16: Log thu thập được ở các dạng file khác nhau 24](#_Toc187845615)

[Hình 17: Log thu thập dưới dạng Database SQLite 25](#_Toc187845616)

[Hình 18: Kết quả thu thập của các tài khoản Nfacct 30](#_Toc187845617)

[Hình 19: Hiển thị các địa chỉ lưu file log 33](#_Toc187845618)

[Hình 20: List file log dưới dạng bảng 34](#_Toc187845619)

[Hình 21: List log dưới dạng Dataframe 34](#_Toc187845620)

# GIỚI THIỆU CƠ QUAN THỰC TẬP

## TỔNG QUAN VỀ CƠ QUAN THỰC TẬP

**Tên công ty**: LG ELECTRONICS DEVELOPMENT VIETNAM.

Địa chỉ thực tập: LG CyberSecurity Lab, Trường Đại học Bách khoa - Đại học Đà Nẵng (Địa chỉ: 54 Nguyễn Lương Bằng, Hoà Khánh Bắc, Liên Chiểu, Đà Nẵng).

**Lịch sử hình thành và phát triển:**

Năm 2023, LG thành lập công ty con chuyên về R&D tại Việt Nam: LG Electronics R&D Việt Nam (tên gọi tắt: LGEDV).

LGEDV xây dựng hệ thống nhân lực bản địa hoàn thiện và bền vững, thực hiện các dự án độc lập.

**Lĩnh vực hoạt động:**

LGEDV chịu trách nhiệm chính về phát triển và kiểm thử phần mềm Thông tin giải trí trên xe ô tô (IVI; In-Vehicle Infotainment) – là cốt lõi trong hoạt động kinh doanh giữa doanh nghiệp với doanh nghiệp (B2B) ở lĩnh vực giải pháp di động.

Ngoài ra, LGEDV phát triển công nghệ webOS – nền tảng cho hệ điều hành thiết bị thông minh của riêng mình - từ năm 2022 và mở rộng phạm vi nghiên cứu sang lĩnh vực thiết bị gia dụng bắt đầu từ năm nay.



Hình 1: Hình ảnh về phòng lab LG

## KẾT CHƯƠNG

Mục tiêu nghiên cứu chủ yếu là WebOS và cách dựng tường lửa trên hệ điều hành này.

# NỘI DUNG THỰC TẬP

## TỔNG QUÁT CÁC CÔNG VIỆC ĐƯỢC GIAO

**BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Tuần 1 | Tuần 2 | Tuần 3 | Tuần 4 | Tuần 5 | Tuần 6 |
| Minh Quân | Iptables  (Create Rules) | Scapy (Code) | IPv4/IPv6, Multicast, Broadcast,  Anycast,  Unicast  (Code) | Ulogd  (Nflog) | Nfacct API  (Code) | Nflog API  (Code) |
| Tấn Sang | Iptables  (Create Rules) | Scapy (Config, Fixbug) | Config (Ethernet) | Ulogd  (Nfct) | Config Nfacct | Config Nflog |
| Hoàng Huân | Iptables  (Create Rules) | Scapy  (Config, Fixbug) | Ipv4/Ipv6  Anycast  Unicast | Config Ulogd | Config Nfacct | Config Nflog |

### WebOS

1. Cách xây dựng hệ điều hành WebOS từ mã nguồn: [Building webOS OSE | webOS Open Source Edition](https://www.webosose.org/docs/guides/setup/building-webos-ose/)

2. Yêu cầu phần cứng để xây dựng WebOS: [System Requirements | webOS Open Source Edition](https://www.webosose.org/docs/guides/setup/system-requirements/)

3. Cách nạp WebOS vào thẻ nhớ của RaspberryPi4: [Flashing webOS OSE | webOS Open Source Edition](https://www.webosose.org/docs/guides/setup/flashing-webos-ose/)

### Tường lửa trên WebOS

Tìm hiểu iptables & netfilter, scapytcpdump in firewall test, IPv4IPv6 type, NfacctNflog, Ulogd2 Logrotate, Nftable.

Sau đó code các API để theo dõi các tiến trình của Nfacct, Nflog theo sơ đồ tuần tự

## CHI TIẾT CÁC NỘI DUNG ĐƯỢC GIAO

**[Tuần 1]** **Cài đặt hệ điều hành WebOS, triển khai Iptables lên hệ thống:**

• Nạp các rules để chống dò quét cổng (port scanning) và tấn công từ chối dịch vụ (DoS/ DDoS attack):  
• Từ máy thật hoặc máy ảo thử tấn công WebOS trên Raspberry với các kiểu tấn công tiêu biểu sạu  
*SYN flood attack: hping3 -S -p 80 169.254.160.86 --flood*  
*Random Source Attack: hping3 -S -p 80 169.254.160.86 --flood --rand-source*  
*Smurf Attack: hping3 --icmp --flood --spoof 169.254.153.216 169.254.160.86*  
*LAND Attack: hping3 -S -p 80 169.254.160.86 -a 169.254.160.86*

**[Tuần 2]** **Scapy tcpdump In Firewall Test:**

Trong chủ đề này, chúng ta sẽ tập trung vào Scapy với việc gửi gói tin từ Windows đến Raspberry Pi 4 để kiểm tra xem tường lửa có hoạt động tốt hay không.  
scapy có thể giúp sửa đổi các gói tin Ethernet với trường này:  
- Ethernet (địa chỉ MAC nguồn/đích, Ethertype)  
- ID Vlan  
- IPv4/IPv6  
- giao thức: TCP/UDP/ICMP...

Sau khi thiết lập IPv6 và VLAN trên Windows thì phát triển các chương trình python cho phép thay đổi các trường của gói tin Ethernet.

**[Tuần 3] IPv4/IPv6 Unicast, Multicast, Broadcast, Anycast:**

1. Xây dựng hệ thống gồm 3 thiết bị Raspberry đã cài đặt sẵn WebOS kết nối qua LAN

2. Xây dựng các chương trình python để gửi/ nhận các thông điệp bằng IPv4 (Unicast, Broadcast và Multicast) //Multicast sẽ sử dụng giao thức IGMP

3. Xây dựng các chương trình python để gửi/ nhận các thông điệp bằng IPv6 (Unicast, Multicast, Anycast)

4. Đánh giá hiệu năng và an toàn để đưa ra sự lựa chọn khi gửi/nhận thông điệp.

**[Tuần 4]** **Ulogd**

ulogd là một trình nền báo cáo các vấn đề liên quan đến netfilter/iptables. Việc này bao gồm báo cáo vi phạm bảo mật và kiểm toán trên mỗi gói tin. Ulogd còn cho phép báo cáo bảo mật và kiểm toán 1 cách linh hoạt (do người dùng tuỳ chỉnh) trên mỗi luồng lưu lượng.  
*Giả sử WebOS trên các thiết bị Rasberry đã được cài đặt sẵn ulogd2. Để thực thi báo cáo tuỳ biến thì cần tìm hiểu các plugins đầu vào (Input), đầu ra (Output) và thực hiện các công việc sau:*

1. Tìm hiểu các plugins

1. 1. Input plugins: *ULOG, NFLOG, NFCT --> để làm gì và cách vận hành?*

1.2. Output plugins: *LOGEMU, OPRINT, SYSLOG, MYSQL, PGSQL, SQLITE3, PCAP, IPFIX, NACCT, JSON --> để làm gì và cách vận hành?*

*1.3. Tìm hiểu trên Ulogd trên WebOS hiện tại hỗ trợ những plugins (Input, Output) nào?*

2. Xây dựng Ulogd tuỳ biến dựa vào các plugins

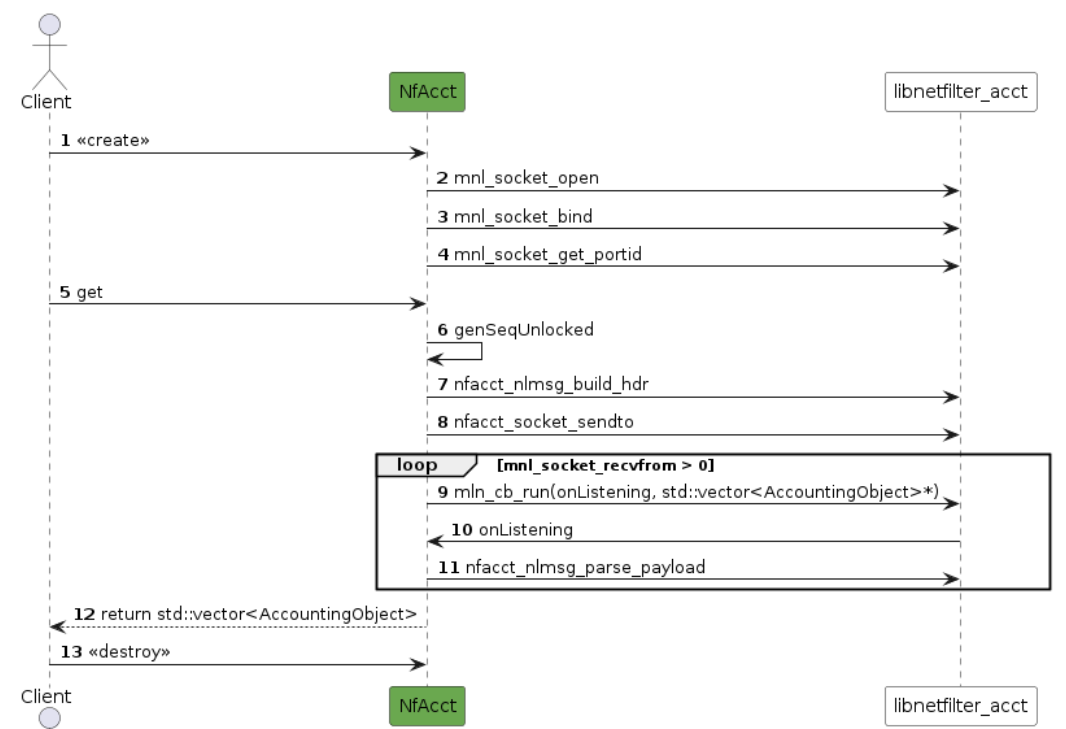
*Giả sử đã tìm hiểu được các Input/ Output plugins trên hệ thống WebOS của mình. Xây dựng Ulogd tuỳ biến để báo cáo các vi phạm/ kiểm toán về gói tin/ luồng lưu lượng (tất nhiên cần xây dựng trước chính sách bảo mật --> iptables --> kiểm tra vi phạm/kiểm toán dựa vào Ulogd tuỳ biến)*

**[Tuần 5] Nfacct API**

Yêu cầu triển khai:

*1. Dựa vào gợi ý về khái niệm và tính năng của Nfacct hãy xây dựng các file cấu hình của Nfacct để thêm/sửa/xoá các tài khoản. Chú ý kết hợp với iptables để kiểm tra có thực hiện được các thao tác trên hay không?*

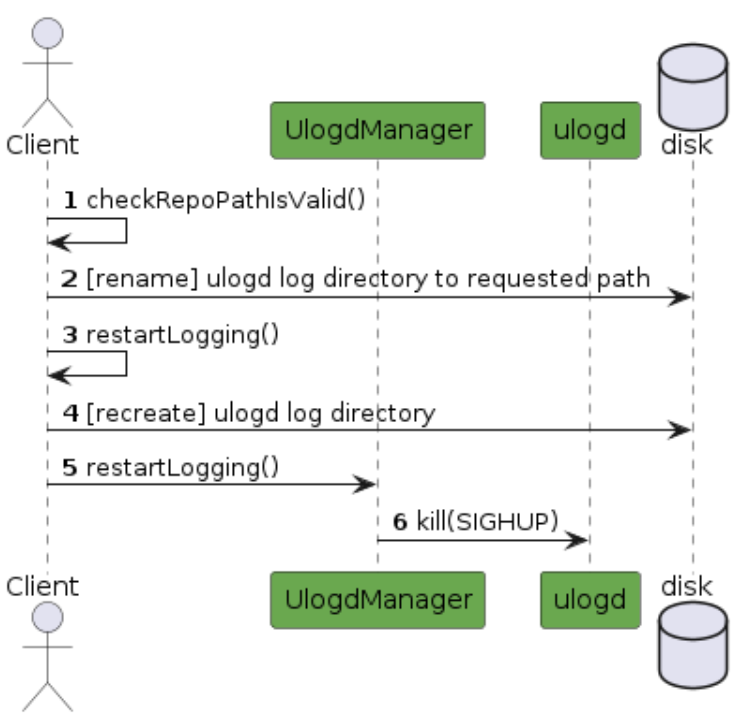
*2. Xây dựng API thực thi các công việc dựa vào Sequence Diagram*



Hình 2: Sơ đồ Sequence Nfacct API

**[Tuần 6] Nflog API**

*Dựa vào Sequence Diagram hãy xây dựng API của Nflog để thu thập và báo cáo các thao tác của người dùng khi tương tác với WebOS trên Raspberry. Có thể chỉ ra các vi phạm vào chính sách tường lửa của người dùng.*



Hình 3: Sơ đồ Sequence Nflog API

**[Tuần 7,8] Viết báo cáo**

## BÁO CÁO CÁC NHIỆM VỤ CÁ NHÂN

### [Tuần 1] Iptables

**(Create Rules)**

Xây dựng các Rules chống PortScanning và DdoS

Triển khai các quy tắc theo thứ tự trên iptables, sử dụng Netfilter để xử lý các gói tin mạng trên Raspberry4. Sử dụng bảng mangle của iptables, đây là bảng chủ yếu để xử lý các gói tin ở mức chi tiết.

**Chống PortScanning**:

*iptables -N port-scanning*   
*iptables -A port-scanning -p tcp –tcp-flags SYN,ACK,FIN,RST RST -m limit –limit 1/s –limit-burst 2 -j RETURN*   
*iptables -t mangle -A INPUT -p tcp –syn -m state –state NEW -m recent –set*   
*iptables -t mangle -A INPUT -p tcp –syn -m state –state NEW -m recent –update –seconds 60 –hitcount 5 -j DROP*   
*iptables -A port-scanning -j DROP*

Tạo chain mới Port-Scanning, gghi lại địa chỉ IP nguồn của các kết nối mới trong danh sách "recent" để theo dõi.

Nếu có **5 kết nối SYN** từ cùng một địa chỉ IP trong vòng 60 giây, hành vi này sẽ bị coi là quét cổng hoặc tấn công.

Sau đó chặn các kết nối đó.

**Chống DdoS:  
Limit các kết nối mới, TCP, ICMP…**

*iptables -A INPUT -p tcp -m conntrack --ctstate NEW -m limit --limit 60/s --limit-burst 20 -j ACCEPT*

*iptables -A INPUT -p tcp -m conntrack --ctstate NEW -j DROP*

*iptables -A INPUT -p tcp -m conntrack --ctstate NEW -m limit --limit 60/s --limit-burst 20 -j ACCEPT*

*iptables -A INPUT -p tcp -m conntrack --ctstate NEW -j DROP*

*iptables -A INPUT -p tcp --tcp-flags RST RST -m limit --limit 2/s --limit-burst 2 -j ACCEPT*

*iptables -A INPUT -p tcp --tcp-flags RST RST -j DROP*

Giới hạn tốc độ kết nối mới ở mức 60 kết nối mỗi giây, với một burst tối đa là 20 kết nối trước khi bắt đầu áp dụng giới hạn. Nếu vượt quá thì DROP.

**Kết quả cuối cùng sau khi kết hợp Rules giữa các thành viên với nhau để tạo:**

A screenshot of a computer screen

Description automatically generated

Hình 4: Kết quả sau khi triển khai Iptables

**Nhận xét kết quả tuần 1:**   
- Đã triển khai các Rules trong Iptables cùng với Netfilter trên Raspberry4 khá tốt. Có kết quả khả quan, và chống được các cuộc tấn công.   
*-* Các gói tin có cờ TCP hoặc các cờ khác giả mạo đều bị chặn ở rules đầutiên(Invalid).   
*-* Các kĩ thuật DoS/DDoS đều bị Drop.   
*-* Không thể PortScanning được, các cổng đã được filter*.*

### [Tuần 2] Scapy tcpdump testing Firewall

**(Code)**

Gửi gói tin ở máy nguồn, tới máy đích, sử dụng payload để thay đổi Ipv6, Ipv4, Protocol, của máy đích, và sử dụng code ở máy đích để xử lí payload trong gói tin

**Import các thư viện cần thiết** *from scapy.all import \**

*from scapy.layers.inet import \**

*from scapy.layers.inet6 import \**

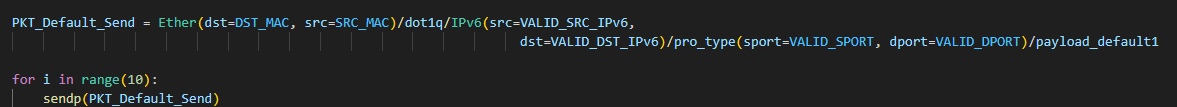
*from netaddr import \**

*from sqlalchemy import false*

**Các thông số cần thiết cho gói tin:**

SRC\_MAC,DST\_MAC,VLAN\_ID,VALID\_SRC\_IPv6,VALID\_DST\_IPv6VALID\_SPORT, VALID\_DPORT,pro\_type = TCP.

**Code gửi gói tin:**

****

**Payload đổi Ipv6 của máy đích:**

**A computer screen with text and numbers

Description automatically generated  
Nhận gói tin và xử lí payload ở máy đích:**Lắng nghe các gói tin IPv4 và Ipv6 sau đó kiểm tra payload (raw\_payload) trong các gói tin đó. Nếu payload chứa lệnh os.system, nó sẽ thực thi lệnh đó trên hệ thống bằng cách sử dụng exec().

**A screen shot of a computer program

Description automatically generated**

**Dựng tường lửa để kiểm tra xem khi gói tin được gửi đi thì tường lửa có block được không?**

Sử dụng Rules chặn IP của máy nguồn

iptables -A INPUT -s 10.10.22.115 -j DROP

**Kết quả:**

Thông báo chặn 10 gói tin

**A screenshot of a computer

Description automatically generated**Nhưng payload vẫn được thực thi và Ipv4 đã bị đổi:

**A computer screen with white text

Description automatically generated**

**A screen shot of a computer

Description automatically generated**Với Ipv6 cũng thế, cũng bị thay đổi bằng payload **A screen shot of a computer

Description automatically generated**

Hình 5: Kết quả đổi IP đích khi gửi payload bằng Scapy

**Nhận xét tuần 2:**

Lí do là tường lửa chỉ chặn gói tin dựa trên địa chỉ IP hoặc cổng mà không thể kiểm tra nội dung của payload. Vì vậy gói tin vẫn được đọc và thực thi. Vậy nếu không kiểm soát lỗ hổng ứng dụng thì tường lửa cũng không chặn được tấn công và bị khai tháng bằng Payload.

### [Tuần 3] IPv4/IPv6, Multicast, Broadcast, Anycast, Unicast

**(Code)**

Xây dựng chương trình python gửi nhận thông điệp , sử dụng socket.

**IPv4:**

**- Unicast: Gửi tới địa chỉ đích từ máy nguồn, và chỉ máy đích nhận được.**

# Function Send Unicast(IPv4)

def send\_unicast\_ipv4(message, target\_ip, target\_port):

    with socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM) as s:

        s.sendto(message.encode(), (target7\_ip, target\_port))

**- Broadcast: Gửi tới toàn mạng.**

# Function Send Broadcast(IPv4)

def send\_broadcast\_ipv4(message, target\_port):

    with socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM) as s:

        s.setsockopt(socket.SOL\_SOCKET, socket.SO\_BROADCAST, 1)

        s.sendto(message.encode(), (255.255.255.255', target\_port))

**- Multicast (Sử dụng giao thức IGMP):  Gửi tới nhóm được tham gia.**

Send: Thay đổi IP nhóm để các thành viên tham gia vào.

sock=socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM, socket.IPPROTO\_UDP)

sock.setsockopt(socket.IPPROTO\_IP, socket.IP\_MULTICAST\_TTL, MULTICAST\_TTL)

sock.sendto(b"MulticastIPv4forMember",(MCAST\_GRP,MCAST\_PORT))

Receive: Chỉ định địa chỉ IP nhóm, khi máy send gửi tới thì sẽ nhận được.

sock = socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM, socket.IPPROTO\_UDP)

 sock.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

mreq=struct.pack("4sl",socket.inet\_aton(MCAST\_GRP),socket.INADDR\_ANY)

sock.setsockopt(socket.IPPROTO\_IP,socket.IP\_ADD\_MEMBERSHIP, mreq)

**IPv6:**

**- Unicast**

#Function Send Unicast(IPv6)

def send\_unicast\_ipv6(message, target\_ip, target\_port):

    with socket.socket(socket.AF\_INET6, socket.SOCK\_DGRAM) as s:

        s.sendto(message.encode(), (target\_ip, target\_port))

**- Multicast**

Send:

sock=socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM, socket.IPPROTO\_UDP)

sock.setsockopt(socket.IPPROTO\_IP,socket.IP\_MULTICAST\_TTL, MULTICAST\_TTL)

Receive:

mreq = struct.pack("16sI", socket.inet\_pton(socket.AF\_INET6,MCAST\_GRP), 0)

sock.setsockopt(socket.IPPROTO\_IPV6, socket.IPV6\_JOIN\_GROUP, mreq)

**- Anycast**

#Function Send Anycast(IPv6)

def send\_anycast\_ipv6(message, target\_ips, target\_port):

    for ip in target\_ips:

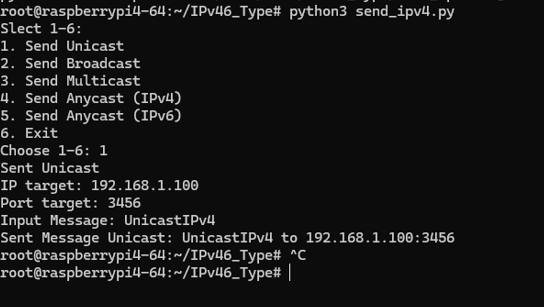
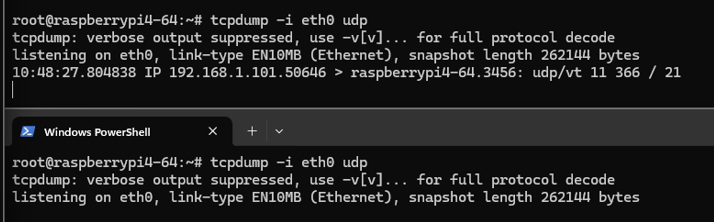
        with socket.socket(socket.AF\_INET6, socket.SOCK\_DGRAM) as s:

            s.sendto(message.encode(), (ip, target\_port))

            print(f"Send Message Anycast to {ip}:{target\_port}")

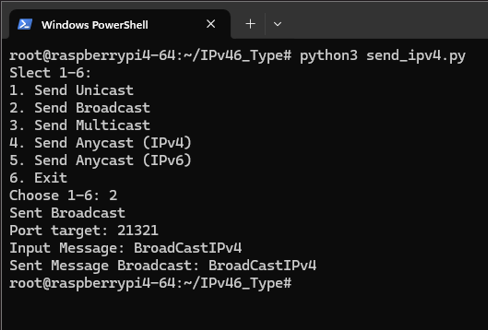
**Kết quả:**

**Unicast** **(IPv4):**

Hình 6: Kết quả gửi bằng Unicast IPv4

**Broadcast (IPv4):**

A screenshot of a computer

Description automatically generated

Hình 7: Kết quả gửi bằng BroadCast IPv4

**Multicast (IPv4):**

A screen shot of a computer program

Description automatically generated  
A screenshot of a computer program

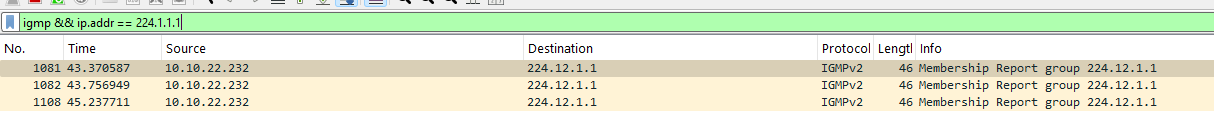
Description automatically generated

Hình 8: Kết quả gửi bằng Multicast IPv4

**Giao thức IGMP:**

**- Nhóm 224.12.1.1:**

- Thêm vào nhóm:



Hình 9: Thêm thành viên vào nhóm bằng giao thức IGMP

- Rời nhóm: Thông báo qua địa chỉ đích 224.0.0.2 về việc nó rời khỏi nhóm



Hình 10: Xóa thành viên vào nhóm bằng giao thức IGMP

**Unicast(IPv6):**

Config IPv6 cho các máy raspberry:

Máy send(abcd:1234::20/64) , hai máy nhận (abcd:1234::7/64 và abcd:2345::5/64)   
A screenshot of a computer screen

Description automatically generated

Hình 11: Config IPv6

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

Hình 12: Kết quả gửi Unicast IPv6

**Multicast (IPv6):**

Chia ra 2 nhóm để kiểm tra (ff02:12:1:1 và ff02:1:1:1)

Máy gửi sẽ gửi thông điệp tới từng nhóm để kiểm tra.

A screenshot of a computer

Description automatically generated  
A screen shot of a computer

Description automatically generated

Hình 13: Kết quả gửi Multicast IPv6

Mỗi nhóm sẽ nhận tin nhắn riêng, chỉ những thành viên thuộc nhóm thì mới nhận được tin nhắn của nhóm đó.

**Anycast(IPv6):**

Config 2 máy có cùng địa chỉ IP giống nhau, gửi gói tin, gói tin sẽ được gửi tới máy gần nhất theo nhiều nghĩa. Nhưng chỉ được một máy nhận. A screenshot of a computer screen

Description automatically generated

Hình 14: Config IPv6 cho Anycast

Gửi gói tin từ máy nguồn: abcd:1234::20/64

A screenshot of a computer

Description automatically generated

Hình 15: Kết quả gửi Anycast IPv6

Kết quả cho thấy chỉ một máy nhận được thông điệp theo phương thức anycast, và máy nhận được là máy “gần nhất” với máy gửi.

**Nhận xét tuần 3:**

**- Multicast** cũng là phương thức gửi hiệu quả nhất trong các trường hợp có nhiều thiết bị nhận thông tin giống nhau.

**- Broadcast**: Gửi đến tất cả thiết bị trong mạng con. Trường hợp của Raspberry, với phần cứng hạn chế, có thể gặp khó khăn trong việc xử lý quá nhiều broadcast.

**- Anycast** có thể giúp tối ưu hóa độ trễ và tăng tính sẵn sàng cho các dịch vụ, vì nó gửi tới nút gần nhất. Khó bị tấn công vì nếu bị DoS chẳng hạn thì gói tin sẽ chuyển qua nút khác gần thứ hai.

**- Unicast** nếu không mã hóa và xác thực thì dễ bị tấn công, **Broadcast** là phương thức dễ bị tấn công nhất, do mọi thiết bị trong mạng đều nhận được thông điệp.

### [Tuần 4] ULOGD

**(nflog)**

Xác định Nflog hỗ trợ các loại Input và Output gì sau đó config stack và xuất các loại file đó ra  ****

Các loại file mà Nflog hỗ trợ, trích xuất log, các gói tin đến hoặc đi qua các quy tắc firewall (iptables).

**Xây dựng Rules Iptables ghi lại (logging) các gói tin vào log của hệ thống:**

iptables -A INPUT -p icmp -m limit --limit 1/s --limit-burst 1 -j NFLOG --nflog-prefix "ping!!!" --nflog-group 1

* Rule này ghi lại các gói ICMP (ping) đến hệ thống, với tốc độ giới hạn là 1 gói mỗi giây, và prefix"ping!!!" sẽ được thêm vào log.

iptables -A INPUT -p tcp -m limit --limit 1/s --limit-burst 1 -j NFLOG --prefix "tcp!!!" --nflog-group 1

* Rule này ghi lại các gói TCP đến hệ thống, với tốc độ giới hạn là 1 gói mỗi giây, và prefix"tcp!!!" sẽ được thêm vào log.

Log nhóm chọn là Log2 và khai báo group=1

**Với LOGEMU:**

**Stack:** stack=log2:NFLOG,base1:BASE,ifi1:IFINDEX,ip2str1:IP2STR,print1:PRINTPKT,emu1:LOGEMU

**Cấu hình:**

BASE: Lưu trữ thông tin cơ bản của gói tin.

IFINDEX: Lấy chỉ số giao diện mạng.

IP2STR: Chuyển đổi địa chỉ IP thành dạng chuỗi dễ đọc.

PRINTPKT: In thông tin chi tiết của gói tin.

Log2:

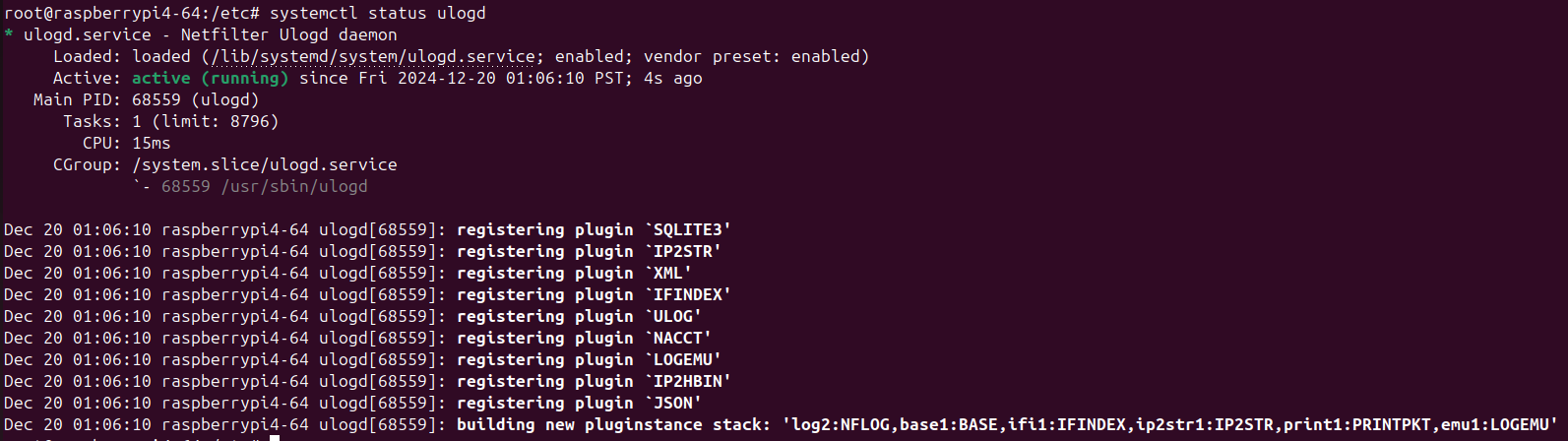
A screenshot of a computer program

Description automatically generated  
[emu1]

A white text on a purple background

Description automatically generated  
**Kết quả:**

Register

Log bắt đượcA screen shot of a computer screen

Description automatically generated

Json

A screenshot of a computer

Description automatically generatedPcap( đọc gói tin log bằng tcpdump)

A black and white screen

Description automatically generated

Hình 16: Log thu thập được ở các dạng file khác nhau

Các loại file khác cũng tương tự và trích xuất ra file log lưu vào thư mục (Gprint, Oprint, XML, NFACCT…)  
Với dạng log lưu vào Database, vì dùng Rasp4 nên chỉ triển khai được trên SQLite

**Stack:** stack=log20:NFLOG,base1:BASE,ifi1:IFINDEX,ip2str1:IP2STR,mac2str1:HWHDR,boot\_sqlt:SQLITE3

**Cấu hình:**

Log20: Group20

[boot\_sqlt]   
    table="all\_in"   
    db="/var/log/ulog/bootlog-sqlt3.db"

Tạo bảng all\_in bằng lệnh SQL với các trường là các thông tin của gói tin.

**Kết quả:**

A computer screen shot of a computer

Description automatically generated

Truy vấn bằng lệnh: SELECT \* FROM all\_in LIMIT 10;   
A close up of a screen

Description automatically generated

Hình 17: Log thu thập dưới dạng Database SQLite

**Nhận xét tuần 4:**

Về cá nhân em đã triển khai Ulogd thành công với Input: Nflog với Output: Logemu, Json Xml, PCAP, NACCT, Oprint, Gprint, SQLITE3. Các kết quả cho thấy đã trích xuất log thành công, với Nflog là tùy biến theo Iptables để theo dõi và ghi lại các gói tin mạng trong không gian người dùng.

### [Tuần 5] Nfacct API

**(Code)**

Dựa vào gợi ý về khái niệm và tính năng của Nfacct hãy xây dựng các file cấu hình của Nfacct để thêm/sửa/xoá các tài khoản.

Chú ý kết hợp với iptables để kiểm tra có thực hiện được các thao tác trên hay không?

**Tóm tắt sơ đồ:**

**Khởi tạo (Create)**:

* Mở socket (mnl\_socket\_open).
* Ràng buộc socket (mnl\_socket\_bind).
* Lấy portid từ socket (mnl\_socket\_get\_portid).

**Lấy dữ liệu (Get)**:

* Tạo một thông điệp Netlink (nfacct\_nlmsg\_build\_hdr).
* Gửi thông điệp (nfacct\_socket\_sendto).
* Nhận dữ liệu (mnl\_socket\_recvfrom trong vòng lặp).
* Gọi callback (mln\_cb\_run).

**Hủy kết nối (Destroy)**:

* Đóng socket.

Code các chức năng:

addNfacct():

Thiết lập thông điệp Netlink:

**seq = time(NULL); nlh = nfacct\_nlmsg\_build\_hdr(buf, NFNL\_MSG\_ACCT\_NEW, NLM\_F\_CREATE | NLM\_F\_ACK, seq); nfacct\_nlmsg\_build\_payload(nlh, acct);**

Tạo socket Netlink -> Gửi thông điệp đến kernel( chứa tài khoản)

Nhận phản hồi từ kernel -> Lặp để xử lí phản hồi bằng mnl\_socket\_recvfrom.

Đóng socket Netlink: mnl\_socket\_close(nl);

Xóa tài khoản: delNfacct()

Giống với Add Nfacct, Remove Nfacct cũng xử lí như vậy nhưng khác là xóa account. Các bước bao gồm:

Tìm và xác định đối tượng cần xóa: **bool objectFound = false;**

Duyệt qua danh sách objects để tìm đối tượng theo tên

Thiết lập thông điệp Netlink để yêu cầu xóa tài khoản:

**seq = time(nullptr);**

**nlh = nfacct\_nlmsg\_build\_hdr(buf, NFNL\_MSG\_ACCT\_DEL, NLM\_F\_ACK, seq);**

**nfacct\_nlmsg\_build\_payload(nlh, acct);**

Mở socket Netlink -> Gửi thông điệp đến kernel để xóa tài khoản:

**if (mnl\_socket\_sendto(nl, nlh, nlh->nlmsg\_len) < 0) {**

**mnl\_socket\_close(nl);**

**throw std::runtime\_error("Failed to send Netlink message");**

**}**

Nhận và xử lý phản hồi từ kernel, tương tự như trong hàm thêm (addAccountingObject)

Xóa đối tượng khỏi danh sách và dọn dẹp tài nguyên: Tài khoản sẽ được xóa khỏi danh sách objects và bộ nhớ sẽ được giải phóng

List tài khoản (GetNfacct):

Hàm get lấy các đối tượng kế toán từ hệ thống và lưu chúng vào vector objects

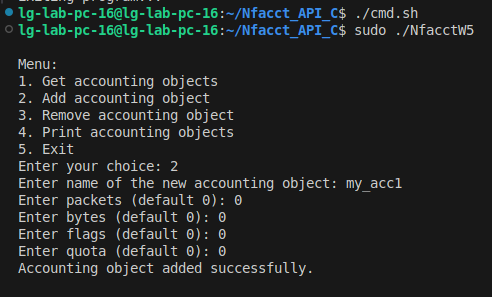
Tạo một thông điệp Netlink (nlmsghdr) để yêu cầu các đối tượng kế toán từ hệ thống -> Sau khi gửi yêu cầu, hàm get sẽ nhận dữ liệu từ socket Netlink.

Hàm sử dụng hàm callback onListening để xử lý thông điệp Netlink nhận được. Nếu một đối tượng kế toán được tìm thấy, nó sẽ được thêm vào vector objects.

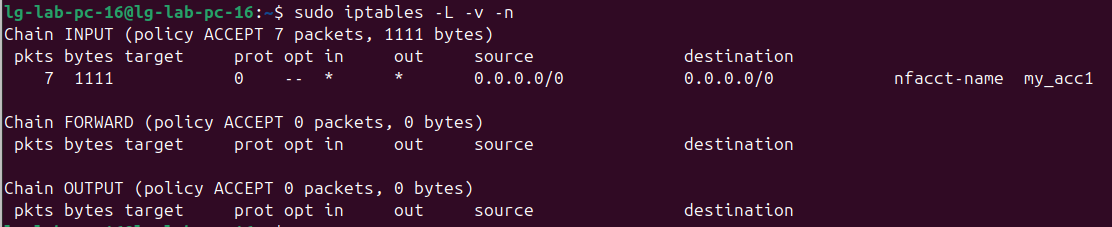
Sau đó, sử dụng print để in các đối tượng ra màn hình

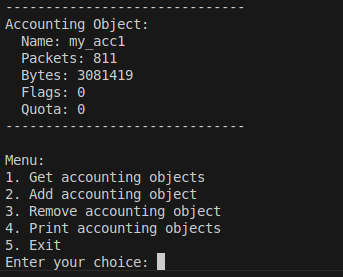
**Kết quả:**

Thêm account:

****Dùng Iptables thêm my\_acc1 vào để thống kê lưu lượng của INPUT

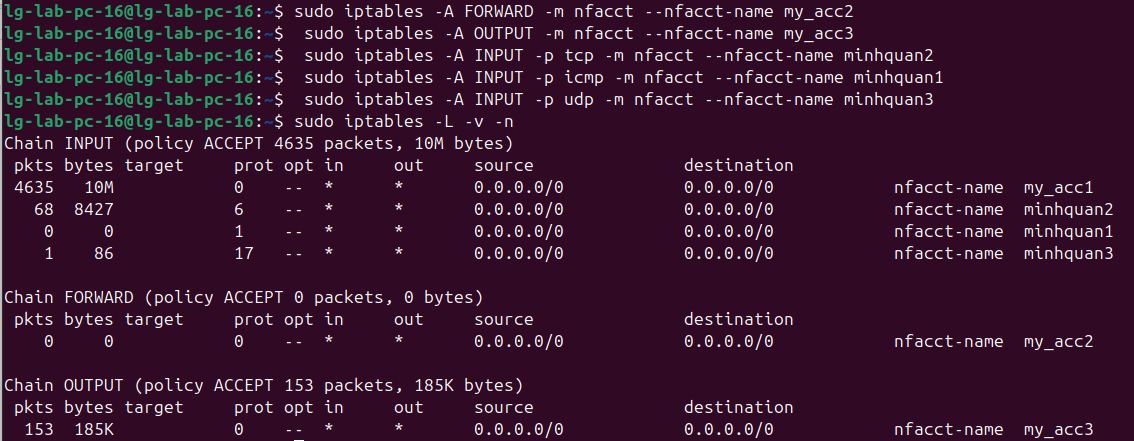
sudo iptables -A INPUT -m nfacct --nfacct-name my\_acc1

****Xem lại lưu lượng bằng code

  
Bây giờ, thêm hàng loạt tài khoản và thiết lập Iptables để thống kê lưu lượng mạng cho mỗi chain, protocol ...   
Thiết lập các rules:

**Rules IPtables:**

sudo iptables -A INPUT -m nfacct --nfacct-name my\_acc1   
sudo iptables -A INPUT -p tcp -m nfacct --nfacct-name minhquan2   
sudo iptables -A INPUT -p icmp -m nfacct --nfacct-name minhquan1   
sudo iptables -A INPUT -p udp -m nfacct --nfacct-name minhquan3   
sudo iptables -A FORWARD -m nfacct --nfacct-name my\_acc2   
sudo iptables -A OUTPUT -m nfacct --nfacct-name my\_acc3

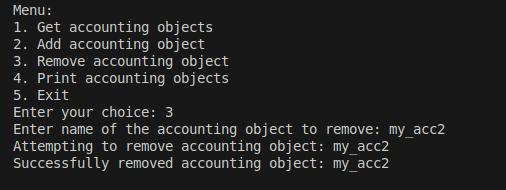


**Kết quả thu thập:**



Hình 18: Kết quả thu thập của các tài khoản Nfacct

Nhận thấy tài khoản my\_acc2 không thu thập được gì, không hiệu quả từ chain Forward => xóa tài khoản.



**Nhận xét tuần 5:**

Về cá nhân em đã triển khai code và xây dựng API theo Sequence Diagram, code hoạt động tốt, các chức năng cho ra kết quả chính xác rõ ràng, liên kết với netlink. Thống kê của mỗi tài khoản đưa ra là đầy đủ phù hợp với các rules Iptables đã thêm. (Tham khảo source code: [*https://git.netfilter.org/nfacct/tree/src/nfacct.c*](https://git.netfilter.org/nfacct/tree/src/nfacct.c)).

### [Tuần 6] Nflog API

**(Code)**Dựa vào Sequence Diagram hãy xây dựng API của Nflog để thu thập và báo cáo các thao tác của người dùng khi tương tác với WebOS trên Raspberry.

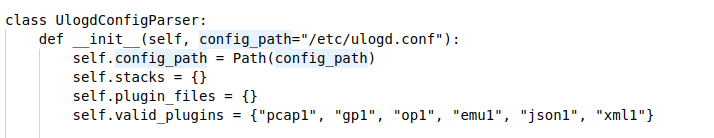
Chỉ ra các vi phạm vào chính sách tường lửa của người dùng.

Cài đặt stack phù hợp cho Ulogd, tạo các group và các log file tương ứng với các group.

Sau khi cài đặt thì code chương trình để list log ra màn hình, trước khi list ra thì check đường dẫn file log valid chưa sau đó list các log đó ra dạng bảng, phù hợp với từng loại file.

Đầu tiên sẽ detect ra được những cái stack sử dụng trong trong ulogd.conf.

Dùng Python để phân tích cấu hình của ulogd, thu thập và hiển thị các log từ các plugin khác nhau (như PCAP, JSON, syslogEMU, v.v.).

  
Ở đây set cứng các tham số plugin như pcap1, emu1...

* Kiểm tra sự tồn tại của file cấu hình (/etc/ulogd.conf).
* Đọc từng dòng trong file cấu hình và xử lý:
* Nếu dòng bắt đầu với stack=, tìm kiếm tên của stack và các plugin kèm theo.
* Nếu có file= hoặc directory=, tìm kiếm đường dẫn của các file log hoặc thư mục.

Sau đó có tùy chọn trích dẫn đường dẫn các file đã tìm thấy trong cấu hình:

**Phương thức display\_logs**:

Kiểm tra xem có logs hay không.

Duyệt qua các stack và plugin, xác định xem log có ở dạng file hay thư mục.

Gọi các phương thức phù hợp để thu thập log từ file, thư mục hoặc PCAP.

Cuối cùng, hiển thị các log dưới dạng bảng.

**Các phương thức thu thập log từ file hoặc thư mục**:

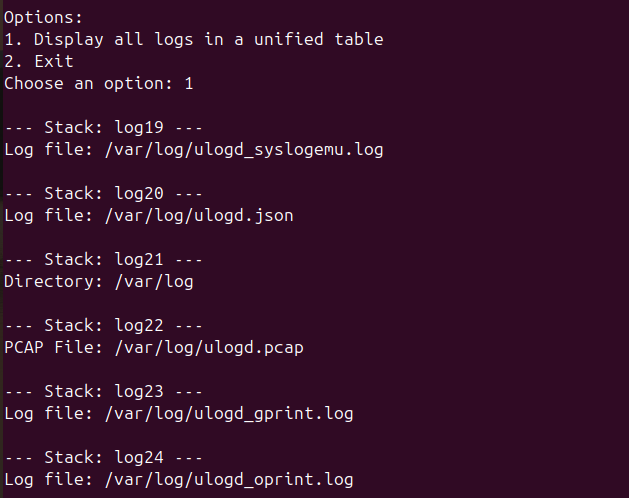
**collect\_log\_file**: Đọc log từ một file đơn lẻ và xử lý dựa trên loại plugin.

**collect\_directory\_logs**: Duyệt qua tất cả các file trong thư mục có phần mở rộng phù hợp với plugin (ví dụ: .json, .xml).

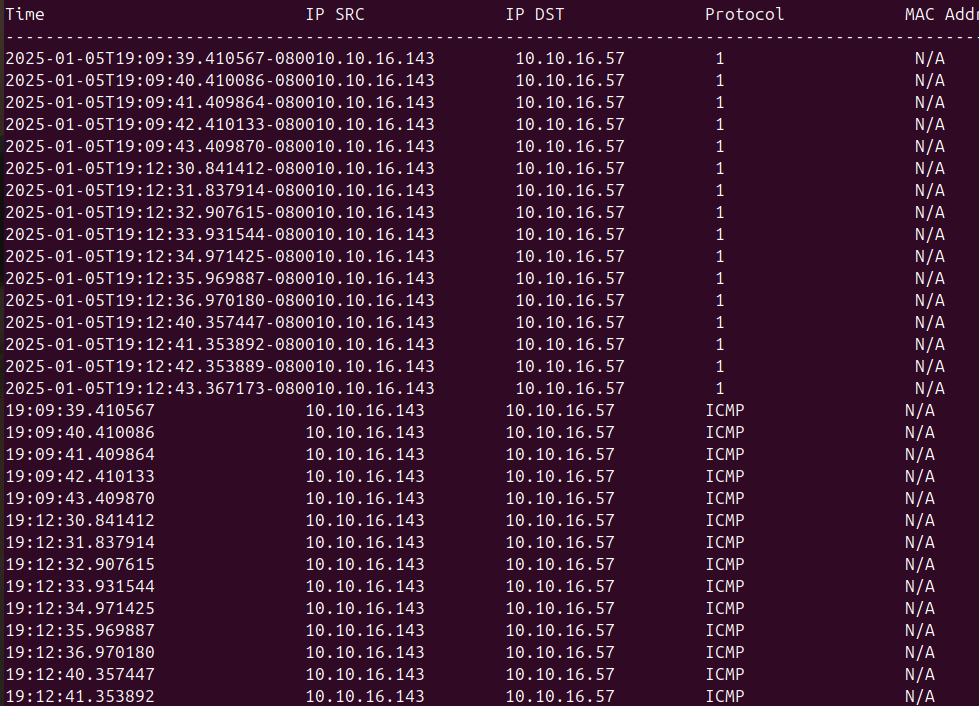
**collect\_pcap\_file**: Dùng tcpdump để đọc và phân tích một file PCAP.

Với các file rõ ràng như Json và Pcap sẽ list dạng bảng, còn các file như Gprint, Oprint, LogEmu sẽ list dạng dataframe.

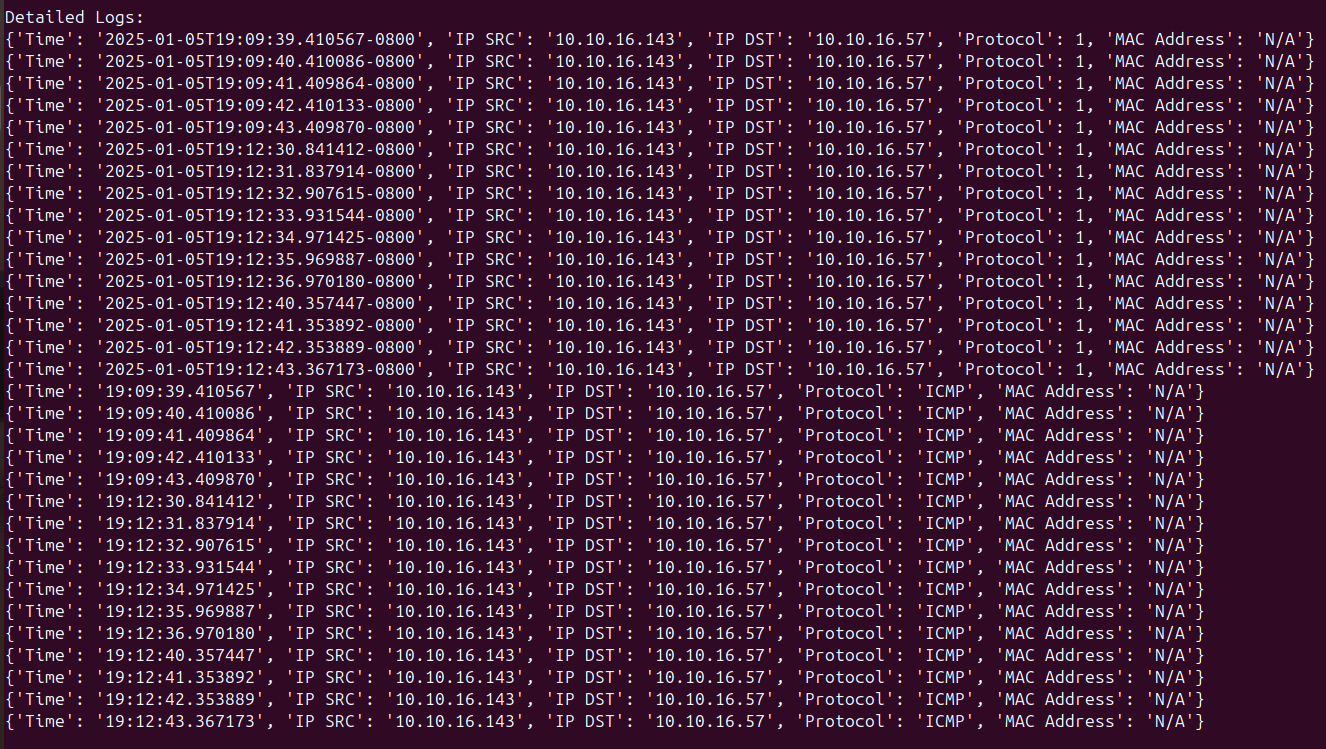
**Kết quả:**

****

Hình 19: Hiển thị các địa chỉ lưu file log

**Dạng bảng:  
**

Hình 20: List file log dưới dạng bảng

**Dạng Dataframe:  
**

Hình 21: List log dưới dạng Dataframe

**Nhận xét tuần 6:**

Triển khai code để thu thập log trong Ulogd, các stack được cấu hình đúng, đã thu thập được log và hiển thị trong file.

Xây dựng code detect tự động và list file ra các định dạng khác nhau tùy với cơ chế của loại file đó.

Các kết quả cho thấy code hoạt động tốt và detect cũng như kiểm tra link file hợp lệ, đầu ra cho kết quả trực quan và rõ ràng.

## TỔNG KẾT

Chương này trình bày việc phân chia công việc nhóm và chi tiết về công việc cá nhân mà em đã thực hiện trong suốt quá trình thực tập.

Các kết quả cho thấy các vấn đề đã được hoàn thành đầy đủ, rõ ràng và chi tiết.

# TỔNG KẾT

## KẾT QUẢ TRIỂN KHAI

Nhóm đã triển khai các nhiệm vụ tuần thành công, các kết quả rõ ràng và chi tiết cho từng phần.

Nhóm đã cấu hình thành công tường lửa để chống các cuộc tấn công mạng. Code Scapy để gửi các gói tin kiểm tra tường lửa.

Cấu hình và code gửi các thông điệp trong IPv4, IPv6 bằng các phương thức gửi khác nhau.

Sử dụng Ulogd để thu thập các log thống kê lưu lượng mạng, và các kết nối mạng.

Sử dụng Nfacct để quản lí các account để nắm rõ và trực quan hóa các thông tin lưu lượng mạng.

Sau đó code NflogAPI, NfacctAPI để giúp người dùng dễ quản lí.

## NHẬN XÉT VÀ KẾT CHƯƠNG

Sau thời gian được thực tập, cá nhân đã được học về an ninh mạng, sử dụng tường lửa và các công nghệ, công cụ được sử dụng trong quản lý và bảo mật mạng.

Giúp bảo vệ hệ thống khỏi các mối đe dọa và cho phép giám sát, ghi log lưu lượng mạng.

Đặc biệt là được thực hành trên hệ điều hành WebOS khá mới lạ, nắm rõ các dịch vụ được hỗ trợ trên hệ điều hành. Nếu có cơ hội thì sẽ xây dựng các ứng dụng trên WebOS và bảo mật cho các ứng dụng đó.

Nhóm đã làm việc và kết hợp để hoàn thành các công việc được giao. Các thành viên cũng hoàn thành các nhiệm vụ cá nhân đúng thời hạn.

TÀI LIỆU THAM KHẢO

1. Nạp WebOS vào thẻ nhớ:

<https://www.webosose.org/docs/guides/setup/flashing-webos-ose/> .

1. Xây dựng hệ điều hành WebOS:

<https://www.webosose.org/docs/guides/setup/building-webos-ose/>

[3] Source Code Nfaact:

[*https://git.netfilter.org/nfacct/tree/src/nfacct.c*](https://git.netfilter.org/nfacct/tree/src/nfacct.c)

[4] Thư viện libnetfilter\_acct:

<https://netfilter.org/projects/libnetfilter_acct/doxygen/html/group__nfacct.html>