TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: **CHUYÊN ĐỀ HỆ ĐIỀU HÀNH LINUX**

BÁO CÁO ĐỒ ÁN THỰC HÀNH 1
MÔN: CHUYÊN ĐỀ HĐH LINUX

Sinh viên thực hiện:
Nguyễn Quốc Bảo - 1412661
Nguyễn Ngọc Vũ - 1412647
Sengdavong Khammakan - 1412690

TP.HCM, ngày 20 tháng 11 năm 2016

Yêu cầu

1. Sử dụng các thư viện hỗ trợ lập trình mạng LIBPCAP, LIBNET để minh họa kiểu tấn công Eavesdropping.
2. Xây dựng kịch bản kiểm tra việc tấn công Eavesdropping với chương trình viết được. Nêu kết quả test với các giao thức: Telnet, FTP
3. Đề xuất giải pháp phòng, chống cách tấn công Eavesdropping. Áp dụng giải pháp này để chống lại kiểu tấn công Eavesdropping. Kết quả?
4. *(Yêu cầu nâng cao): Nghiên cứu và xây dựng để chương trình hoạt động được trên mạng không dây 802.11*

Yêu cầu 1

Xem clip minh hoạ tại link sau
https://www.youtube.com/watch?v=3pglqX750KA

Trong đó:

- Cả ba máy tính đều nằm chung trên một đường mạng.
- Máy của attacker là máy thật chạy Ubuntu 16.04
- Máy của nạn nhân là hai máy ảo Computer 1 và Computer 2 chạy Debian 8

Chương trình được thiết kế tối ưu và test cẩn thận để không có bất kì warning nào trong quá trình biên dịch cũng như không để xảy ra lỗi memory leakage sau khi thoát (kiểm tra bằng Valgrind)

Trên thực tế thư viện libnet chỉ hữu dụng khi ta phải build các gói tin phức tạp. Trên thực tế mặc dùng thư viện LIBPCAP dùng để bắt gói tin nhưng trong nó vẫn có hỗ trợ hàm pcap_inject() cho phép gửi gói tin dạng thô đi (đã đóng ethernet header).

Yêu cầu 2

Để kiểm tra xem liệu một máy tính có bị tấn công Eavesdropping hay không, ta có thể kiểm tra bằng 2 cách sau

1. So sánh độ trễ của các giao thức như ping (ICMP) giữa các máy tính trong mạng.
2. Kiểm tra xem khi gửi gói tin ARP Request, gói tin ARP Response phản hồi hoặc ARP Broadcast có bị trùng lặp một địa chỉ IP gắn với nhiều địa chỉ MAC hay không.

Cách 1 chỉ được áp dụng trong môi trường mạng ổn định và có độ tin cậy cao. Tức là bình thường nếu ta ping đến n-2 máy còn lại trong mạng mà thời gian phản hồi trung bình là a ms. Trong khi nếu ping đến máy thứ n-1 thì thời gian phản hồi trung bình là b ms lớn hơn nhiều so với a (ví dụ b = 10a) thì ta có thể suy đoán rằng có thể đã xảy ra tấn công Eavesdropping.
Kịch bản ở cách 2 có thể mô tả như sau:

Quy ước:
Máy A, B là máy thật, máy C là máy của bên thứ ba.

Cách 1: (dựa trên nguyên tắc tấn công Eavesdropping (Man-In-The-Middle) sẽ có độ trễ)

1. Máy A gửi ARP request cho địa chỉ IP là K
2. Máy A nhận được 2 gói ARP reponse cho 2 địa chỉ MAC1 và MAC2 khác nhau cùng gắn với K.
3. A ping 10 lần đến K theo MAC1 thì thời gian phải hồi trung bình là a ms
4. A ping 10 lần đến K theo MAC2 thì thời gian phải hồi trung bình là b ms
    a. Nếu a >>> b thì kết luận có tấn công Eavesdropping (ARP Spoofing) và MAC1 là địa chỉ MAC của máy C.
    b. Nếu b >>> a thì kết luận có tấn công Eavesdropping (ARP Spoofing) và MAC2 là địa chỉ MAC của máy C.
    c. Nếu a ~ b thì chưa kết luận được có tấn công Eavesdropping (ARP Spoofing) vì đó có thể là sai sót của người quản trị đã cấu hình trùng địa chỉ IP cho 2 máy B và C trong mạng.

Cách 2: (dựa trên nguyên tắc tấn công Eavesdropping (Man-In-The-Middle) sẽ có độ trễ)

5. Máy A gửi 10 ARP request cho địa chỉ IP là K
6. Máy A nhận được 10 gói ARP reponse cho 2 địa chỉ MAC1 và MAC2 khác nhau cùng gắn với K.
    a. Nếu phần lớn gói ARP response chứa MAC1 luôn đến sau các gói ARP response chứa MAC2 thì ta kết luận có tấn công Eavesdropping (ARP Spoofing) và MAC1 là địa chỉ MAC của máy C.
    b. Nếu phần lớn gói ARP response chứa MAC2 luôn đến sau các gói ARP response chứa MAC1 thì ta kết luận có tấn công Eavesdropping (ARP Spoofing) và MAC2 là địa chỉ MAC của máy C.
    c. Các gói ARP response chứa MAC1 và MAC2 đến gần như cùng lúc thì chưa kết luận được có tấn công Eavesdropping (ARP Spoofing) vì đó có thể là sai sót của người quản trị đã cấu hình trùng địa chỉ IP cho 2 máy B và C trong mạng.

Kết quả test với các giao thức TELNET và FTP đã được nếu trong clip minh hoạ.

Yêu cầu 3

Ta có thể chống tấn công Eavesdropping bằng cách đặt địa chỉ MAC address tĩnh trong bảng ARP table ngay từ lúc mới thiết lập mạng. Cách này hoàn toàn chống được tấn công Eavesdropping, không có gì phải bàn nhưng trên thực tế, nó sẽ gây ra sự bất tiện nhất định trong tương lai khi ta cần thay đổi topology của mạng.

Ví dụ, trên Debian, để đặt địa chỉ MAC tĩnh là 12:34:12:34:12:34 cho IP 192.168.1.1 trên interface vmnet8, ta chạy lệnh sau

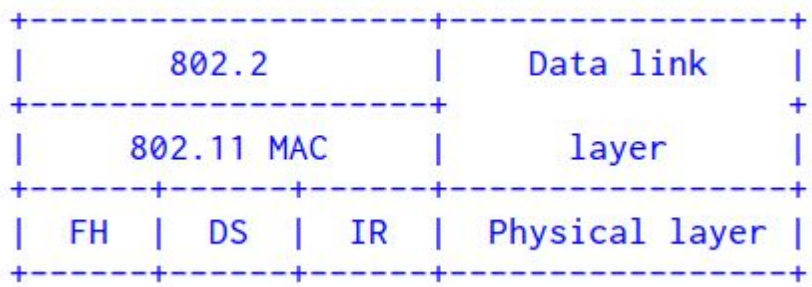
```
sudo arp -s 192.168.1.1 12:34:12:34:12:34 -i vmnet8
```

Đối với những mạng công cộng, khi nhận được ARP response gắn 2 hoặc nhiều địa chỉ MAC vào một địa chỉ IP, ta có thể áp dùng 2 cách đã đề xuất ở **Yêu cầu 2** nhưng để an toàn thì nên drop luôn tất cả, không tin ai hết.

Ngoài ra, còn một cách nữa cũng chống được tấn công Eavesdropping đó là tự thiết lập một đường mạng riêng ảo VPN ra ngoài Internet (bằng cách thuê VPS) mã hoá dữ liệu bằng các thuật toán mã hoá đối xứng (tối thiểu phải tương đương với AES-128bit). Như vậy, bằng cách này, ta không tin và không giao dịch với bất cứ máy tính nào trong mạng LAN, kể cả router (vì ta chỉ dùng router này để kết nối đến VPS ở ngoài qua VPN để truy cập Internet. Để triển khai cách này khá là tốn kém nên nhóm chưa thử nghiệm được.

Yêu cầu 4

Nhóm chưa thể làm được yêu cầu này do chưa thể giả lập được môi trường để chạy thử nghiệm. Tuy nhiên, về nguyên tắc, trên mạng không dây sử dụng công nghệ chuẩn 802.11, nó chỉ khác mạng Ethernet dùng công nghệ chuẩn 802.3 Ethernet header sẽ bị thay bằng 2 lớp header khác là 802.11MAC và 802.2



Như vậy, để chương trình chạy được, ta chỉ cần tìm ra vị trí lưu MAC address trong header 802.11MAC để chỉnh sửa là xong. Đối với ethernet header, Source MAC được lưu trong 6byte đầu tiên và Dest MAC được lưu trong 6 byte tiếp theo trong tổng số 14byte chiều dài (nếu không có 802.1Q Tagging).

Thuật toán

Mã giả của chương trình thực hiện tấn công cơ bản như sau

```
void spoof()
{
// Gửi các gói tin fake ARP response theo định kì mỗi 2 giây
}

void forwarder()
{
// Nếu có gói tin tới mà Dest MAC trùng với MAC của máy tấn công
thì sửa Dest MAC thành địa chỉ MAC của máy nạn nhân kia và đẩy ra
ngoài mạng trở lại.
// Lưu lại payload của gói tin để xử lí sau này.
```

```
}


void signalTrap()
{
// Ngưng lặp hàm forwarder sẽ dẫn đến thread2 bị ngưng
}

int main()
{
// Init signal trap SIGINIT để ngừng chương trình bằng tín hiệu SIGINT có điều khiển

// Mở thread1 chạy hàm spoof()
// Mở thread2 chạy hàm forwarder()
// Join thread2
// Cancel thread1
// Join thread1

// Free bộ nhớ
}
```

Đánh giá và phân công công việc

Mức độ hoàn thành đồ án: 100%
Nguyễn Ngọc Vũ - 1412647 (20%)

- Viết hàm Forward gói tin
- Tìm hiểu cấu trúc gói tin (các loại header) và chỉ lại cho 2 thành viên còn lại.
- Làm giao diện cho chương trình, tham số dòng lệnh.

Nguyễn Quốc Bảo - 1412661 (60%)

- Cài đặt các xử lí có liên quan đến multithread cho chương trình.
- Viết hàm gửi gói tin fake ARP response để tấn công ARP Spoof và debug bằng Wireshark.
- Ráp nối công việc của các thành viên còn lại và debug chương trình.
- Kiểm tra lỗi memory leakage bằng Valgrind.
- Viết báo cáo và quay clip.
- Viết lại comment trong mã nguồn theo chuẩn.
- Tìm hiểu và cài đặt Makefile.

Sengdavong Khammakan - 1412690 (20%)

- Viết các hàm đọc địa chỉ MAC của máy nạn nhân.
- Viết các hàm kiểm tra, chuyển đổi, báo lỗi.

Hướng dẫn biên dịch và chạy

(tham khảo trong clip minh hoạ)

Giải nén file mã nguồn và chạy lệnh make tại thư mục hiện hành

```
make
```

Để chạy chương trình tấn công 2 máy có địa chỉ IP là 192.168.1.2 và 192.168.1.3 trong mạng trên interface eth0, ta gọi tham số dòng lệnh như sau

```
sudo ./libnet -d eth0 -a 192.168.1.2 -b 192.168.1.3
```

Để debug lỗi memory leakage dùng valgrind, ta chạy lệnh như sau

```
sudo valgrind --leak-check=full --show-leak-kinds=all ./libnet -d eth0 -a 192.168.1.2 -b 192.168.1.3
```

Tài liệu tham khảo

https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol
https://en.wikipedia.org/wiki/IPv4
https://en.wikipedia.org/wiki/EtherType
https://en.wikipedia.org/wiki/List_of_IP_protocol_numbers
http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/arp.html
http://www.itsyourip.com/Security/how-to-disable-icmp-redirects-in-linux-for-security-redhatdebianubuntususe-tested/
http://www.tcpdump.org/linktypes.html
http://www.tcpdump.org/sniffex.c
http://libnet.sourcearchive.com/documentation/1.1.4-2.1/annotated.html
https://repolinux.wordpress.com/2011/09/18/libnet-1-1-tutorial/
http://libnet.sourcearchive.com/documentation/1.1.4/libnet-functions_8h.html
http://www.faqs.org/docs/securing/chap5sec57.html
http://unix.stackexchange.com/questions/57941/linux-always-send-icmp-redirect
https://toschprod.wordpress.com/2012/03/04/mitm-8-countermeasures/
http://libnet.sourcearchive.com/documentation/1.1.4/libnet-functions_8h.html
http://libnet.sourcearchive.com/documentation/1.1.4/globals_defs.html
http://www.cas.mcmaster.ca/~rzheng/course/COSC6397sp2008/Libpcap_libnet.pdf
https://en.wikipedia.org/wiki/List_of_IP_protocol_numbers
http://stackoverflow.com/questions/25364677/is-it-possible-to-write-a-packet-read-by-libpcap-with-libnet-in-c
http://www.thegeekstuff.com/2012/05/ip-header-checksum/
http://stackoverflow.com/questions/15372011/configuring-arp-age-timeout
http://www.tcpipguide.com/free/t_DNSMessageHeaderandQuestionSectionFormat.htm
http://mars.netanya.ac.il/~unesco/cdrom/booklet/HTML/NETWORKING/node300.html

http://books.gigatux.nl/mirror/networksecuritytools/0596007949/networkst-CHP-10-SECT-3.html
http://weaknetlabs.com/files/libpcapandc.pdf
https://github.com/weaknetlabs/libpcap-80211-c/blob/master/802sniff.c
http://www.sss-mag.com/pdf/802_11tut.pdf
http://lxr.free-electrons.com/source/include/net/cfg80211.h#L3912
http://www.catb.org/esr/structure-packing/
https://www.kernel.org/doc/man-pages/
http://man7.org/linux/man-pages/index.html
https://www.cyberciti.biz/faq/linux-redirect-error-output-to-file/