

# Open Source Security Tools for Information Technology Professionals

School of Professional Studies (SPS)

The City University of New York (CUNY)

Aron Trauring  
Adjunct Professor  
CEO, Zoteca

Class 9 November 21st, 2005

# Class Agenda

**6:30-6:45** Article of the Week

**6:45-7:15** Knoppix and chkrootkit

**7:15-8:00** Network Sniffers — Part I

**8:00-8:15** Break

**8:00-9:30** Network Sniffer Lab

## **Instructor**

Aron Trauring – Adjunct Professor, CUNY SPS / CEO, Zoteca

**Email:** [atrauring@zoteca.com](mailto:atrauring@zoteca.com)

**Personal Website:** <http://aronst.org/>

**Zoteca Corporate Website:** <http://www.zoteca.com/>

## Network Sniffers

### Link Layer

- Not really part of the Internet protocol suite
- The method used to pass packets from the Internet layer of one device to the Internet layer of another
- Can be controlled both in the software device driver for the network card, as well as on firmware or specialist chipsets
- These perform data link functions such as adding a packet header to prepare it for transmission, then actually transmit the frame over a physical medium
- On receiving end, the link layer gets data frames, strip off the packet headers, and hand the received packets to the Internet layer
- Includes OSI physical layer (physical cabling or other media used to create the network) and data link layer (where data is first encoded to travel over some specific medium)

### Network Sniffers

- Listens or "sniffs" packets on a specified physical network segment
- Allows you to analyze the traffic for patterns, troubleshoot specific problems, and spot suspicious behavior.
- Sniffers are generally specific to the type of network they work on — you must have an Ethernet sniffer to analyze traffic on an Ethernet LAN.
- Commercial sniffers usually are dedicated and expensive hardware devices
- We will focus on Ethernet sniffers since it is mostly widely deployed data link protocol for LANs (now a de facto standard)

### History of Ethernet

- Bob Metcalfe invented and patented Ethernet in 1973 while at the Xerox Palo Alto Research Center (PARC)
- Went on to form a company dedicated to building equipment for this new protocol (3Com)
- Ethernet was released into the public domain so other companies could build to the specification.
- This was not true of Token Ring and most of the other network protocols of the day
- Eventually adopted as an official standard by the International Electrical and Electronic Engineers (IEEE) and a de facto industry standard

### What is Ethernet?

- Ethernet handles both the physical media control and the software encoding for data going onto a network.
- Since Ethernet is a broadcast topology, where every computer can potentially "talk" at once, it has a mechanism to handle collisions
- If a collision is detected, both sides retransmit the data after a random delay.

## MAC Addresses

- Ethernet networks use an addressing scheme called Medium Access Control (MAC) addresses.
- IEEE handles numbering — old standard 48 bits, new standard 64 bits
- They are 12-digit hexadecimal numbers, and are assigned to the card at the factory.
- Every manufacturer has its own range of numbers — first 3 octets of the MAC address.

## Changing MAC addresses

- While physical MAC addresses are permanent by design, mechanisms allow modification, or "spoofing" of the MAC address that is reported by the operating system.
- Can be useful for privacy reasons, for instance when connecting to a Wi-Fi hotspot, or to ensure interoperability.
- Some internet service providers bind their service to a specific MAC address; if the user then changes their network card or intends to install a router, the service won't work anymore.
- Changing the MAC address of the new interface will solve the problem. Similarly, some software licenses are bound to a specific MAC address.
- Changing the MAC address is not permanent: after a reboot, it will revert to the MAC address physically stored in the card.
- macchanger and other tools

## Ethernet and Security

- All computers attached to an Ethernet network are broadcasting on the same physical wire
- An Ethernet card on the network sees all the traffic passing it.
- The Ethernet card is designed to process only packets addressed to it but still sees and processes everything
- Nowadays, most Ethernet networks are switched to improve efficiency.
- This means that instead of each Ethernet port seeing all the traffic, it sees only traffic intended for the machine plugged into it.
- This helps alleviate some of the privacy and congestion issues, but plenty of broadcast traffic still goes to every port.

## Broadcast Traffic

- Broadcast traffic is sent out to every port on the network usually for discovery or informational purposes.
- DHCP, where the machine sends out a broadcast looking for any DHCP servers on the network to get an address from.
- Machines running Microsoft Windows are also notorious for putting a lot of broadcast traffic on the LAN.

- Address Resolution Protocol (ARP); this is when a machine first tries to figure out which MAC address relates to which IP address
- Send out ARP packets asking, "Who has this IP address?" Using reply, it then sends the rest of the communication to the proper MAC address.
- If crackers get access to the switch, they can sometimes turn their own ports into a "monitor" or "mirror" port that shows traffic from other ports.

### **Always Get Permission**

- By capturing every transmission on the wire, you are very likely to see passwords for various systems, contents of e-mails, and other sensitive data
- In the wrong hands, could obviously lead to serious security breaches.
- In addition, it could be a violation of your employees' privacy, depending on company and government policies.
- Always get written permission from a supervisor, and preferably upper management, before you start this kind of activity.
- Generally, network-sniffing logs should be purged from your system
- There are documented cases of well-intentioned system administrators being fired for capturing data without permission.

### **Understand Your Network Topology**

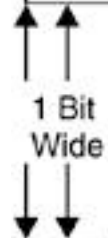
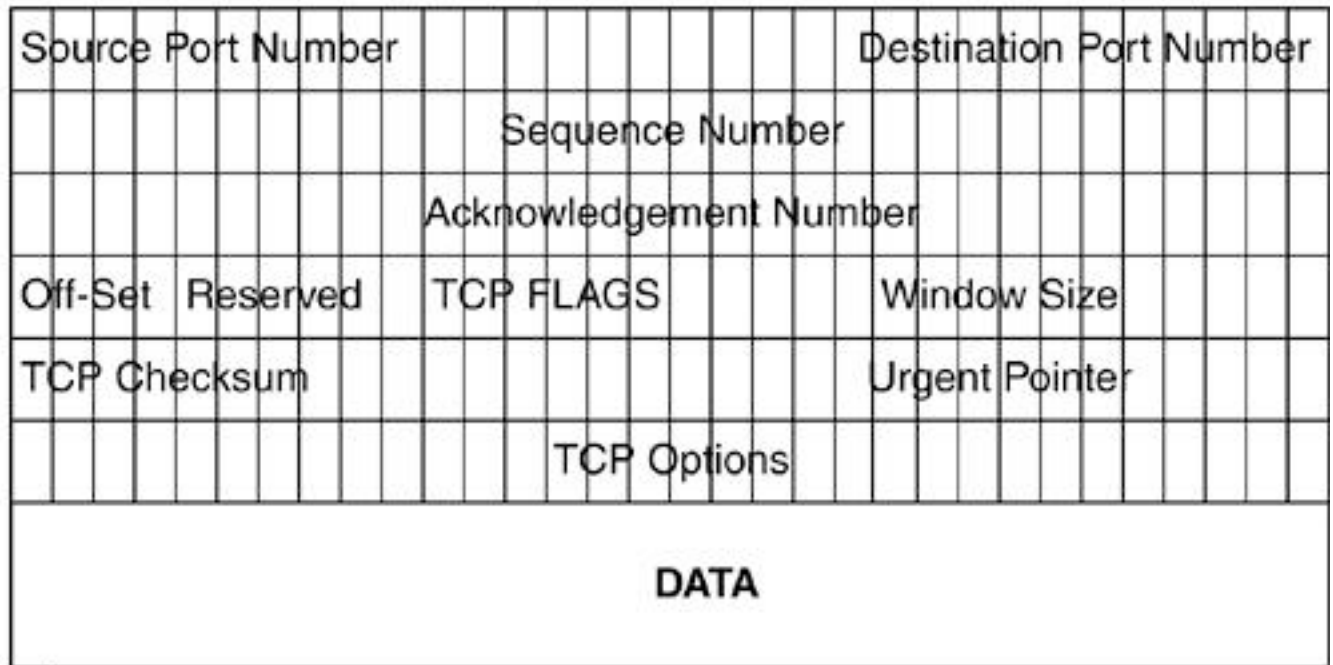
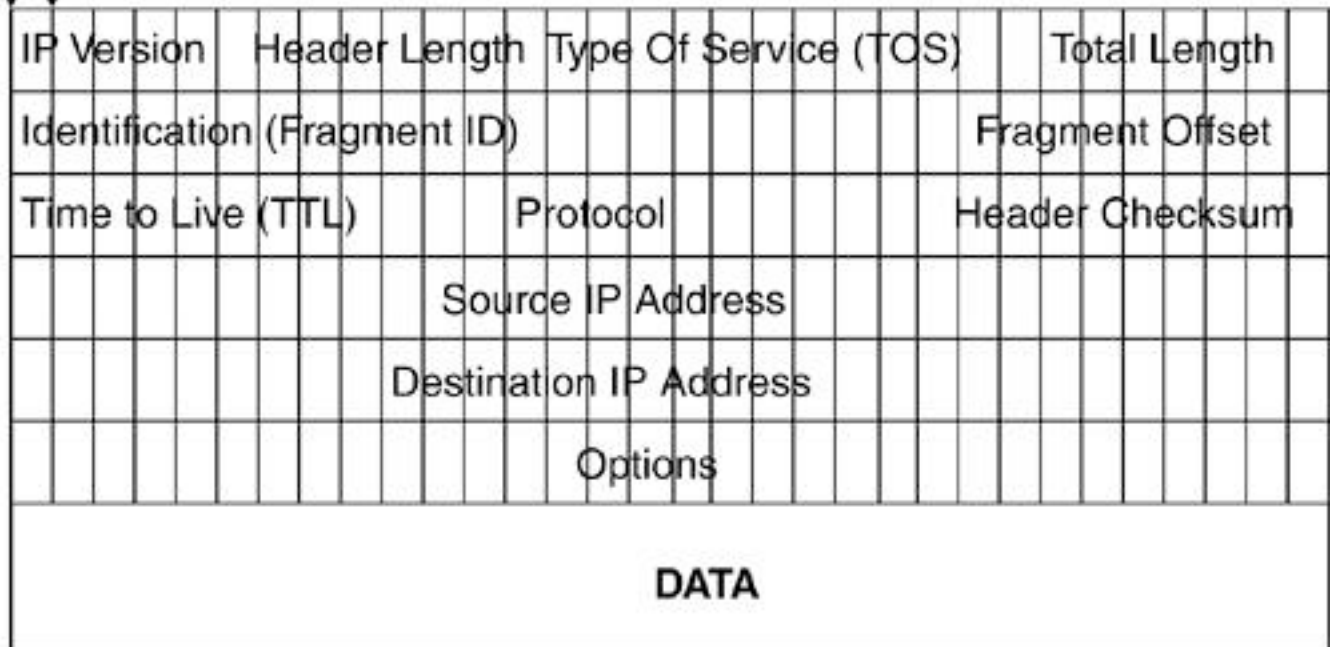
- Make sure you fully understand the physical and logical layout of your network before setting up your sniffer.
- Sniffing from the wrong place on the network will cause you either to not see what you are looking for or to get erroneous results.
- Make sure there is not a router between your sniffing workstation and what you are trying to observe.
- On a switched network, configure the port you are plugged into to be a "monitor" or "mirror" port (need the port to act like a hub so it sees all the traffic on that switch)
- Without this setting, all monitor port sees is the traffic addressed to the specific plugged into and the network's broadcast traffic.

### **Use Tight Search Criteria**

- Using an open filter (that is, seeing everything) will make the output data voluminous and hard to analyze.
- Use specific search criteria to narrow down the output that your sniffer shows.
- Filter even if you are not exactly sure what you are looking for,

**Establish a Baseline for Your Network**

- Use your sniffer to analyze your network during normal operation and record the summary results
- This serves as a baseline to compare it to when you are trying to isolate a problem.
- Monitor and record network traffic over time
- Ethereal sniffer creates several nice reports for this.

**TCP Header Diagram****TCP Header****IP Header**

## TCP/IP Packet Headers

- The layout of the TCP/IP packet is specified in RFC 793 for the TCP portion and RFC 791 for the IP portion.
- Both header types are at least 20 bytes long and are usually shown in five 32-bit (4-byte) sections with the addresses, options, and other settings for the session

## IP Header

- Contains the delivery address for the packet and its sender.
- IP version — 4
- Header Length — usually 20 bytes in IPv4 (may vary in IPv6)
- Type Of Service (TOS) — allows routers and NICs to differentiate the priority of packets (may be used in VoIP)
- Total Length – if subtract header length. get length of data
- Fragment identification — routers may need to break up original datagram
- TTL — number of routers hops packet goes through before it is dropped (decremented by each router)
- Protocol — TCP, UDP, ICMP etc.
- Checksum used to verify header integrity
- Source and destination IP addresses — each address is 32 bits (4 octets of 8 bits each) hence takes up 8 bytes.
- Options field — variable length, padded with zeros or any data (rarely used)

## TCP header

- Takes care of establishing a TCP session and higher-level functions — usually 20 bytes long
- Source port number of 16 bits and a destination port number of 16 bits (hence 65,535 ports)
- Sequence number — used to reassemble the packets in the right order at the other end, even if they arrive in a different order.
- Acknowledgment number — based on sequence number
- Data offset gives how many 32-bit lines or "words" are in this header (typically 4) and
- 6 bits that are reserved for future use
- 6-bit section for the TCP Flags (U,A,P,R,S,F)
- Window size — how much data that can be buffered before receiving an ACK
- TCP checksum — used to verify header and data integrity
- Urgent pointer — points to data which should be passed quickly
- TCP options — rarely used and normally padded with zeros  
`login > rtsg.1023: S 947648:947648(0)  
ack 768513 win 4096 <mss 1024>`
- The actual payload, the data of the packet, follows

## Sniffer Lab

### Useful tcpdump options

- `tcpdump <options> <selector expression>`
- `-a` — attempts to convert addresses to names
- `-c` — stop `tcpdump` after count number of packets
- `-F` — take packets from file (post-analysis)
- `-e` — print link-level header (MAC address on Ethernet)
- `-i <interface>`
- `-n` — don't convert addresses to names
- `-t` — no timestamp
- `-v -vv -vvv` — level of verbosity
- `-w <filename>` — write to file

### tcpdump Selectors

- `host <hostname>`
- `net <IP Network>`
- `port <TCP port>`
- Traffic Direction: `src dst`
- Protocol: `tcp udp icmp`
- Boolean: `and or not`

### tcpdump Output for TCP Protocol

```
12:25:44.578546 csam.login > rtsg.1023: S 947648:947648(0) ack 768513
win 4096 <mss 1024>
```

- Timestamp, broken down into fractions of a second — on a busy network many packets per second
- TCP sequence number
- Source IP address with port
- `>` (a greater than sign) — shows direction
- Destination address with port
- Flags — some combination of S (SYN), F (FIN), P (PUSH), R (RST), W (ECN CWR) or E (ECN-Echo), or a single `'.'` (no flags)
- Data-seqno describes the portion of sequence space covered by the data in this packet
- Ack is sequence number of the next data expected the other direction on this connection.

- Window is the number of bytes of receive buffer space available the other direction on this connection.
- Urg indicates there is 'urgent' data in the packet.
- Options are tcp options enclosed in angle brackets

### tcpdump Output for UDP DNS Request

```
h2opolo.1538 > helios.domain: 3+ A? ucbvax.berkeley.edu. (37)
```

- 3 — query id
- + — recursion
- A? — query type
- (37) — data length

### tcpdump Output for UDP DNS Reply

```
helios.domain > h2opolo.1538: 3 3/3/7 A 128.32.137.3 (273)
```

- 3 — query id
- 3/3/7 — 3 answer records, 3 name server records and 7 additional records
- A 128.32.137.3 — first record record is type A (answer) and IP address
- (273) — data length

```
helios.domain > h2opolo.1537: 2 NXDomain* 0/1/0 (97)
```

- 2 — query id
- NXDomain — response code of non-existent domain ('\*' indicates that the authoritative answer bit was set)
- 0/1/0 — no answers, one name server and no authority records

### tcpdump Output for ARP/RARP

- use -n option

```
arp who-has 128.3.254.6 tell 128.3.254.68
arp reply 128.3.254.6 is-at 02:07:01:00:01:c4
```

- 128.3.254.6 sent an arp packet asking for the Ethernet address of internet host csam 128.3.254.68
- 128.3.254.68 replies with its Ethernet MAC address

### Terminating `tcpdump`

- Hit Ctrl-C prints a summary of all the traffic it saw
- Packets received by filter — count of packets processed by the `tcpdump` filter
- Packets dropped by kernel — number of packets that were dropped due to a lack of resources on your system

### View All Traffic to and from a Particular Host

- To monitor only traffic to and from a specific host, filter everything else out with the `host` expression.
- can track network usage from/to a particular host

```
tcpdump -n host 192.168.1.1
tcpdump -n src host 192.168.1.1
tcpdump -n dst host google.com
```

### Watch Only Traffic Coming in or out on a Certain Port

- To track usage of a certain application
- Use `tcpdump` to trap all traffic for a particular TCP/UDP port.
- Can see who is trying to access
- Useful when someone is trying an attack, e.g. SSH password attack

```
tcpdump -n port 22
```

### View All Traffic to and from a Particular Host but Eliminate Some Kinds of Traffic

- Want to monitor a single host but want to filter out specific traffic
- e.g. if you were ssh'd into that host, unfiltered `tcpdump` output would show your own connection traffic
- Add `port` expression with a Boolean operator `not`
- Note that if using multiple expressions must also use Boolean operator `and`

```
tcpdump -n host 192.168.1.1 and not port 22
```

### Find a Rogue Workstation

- If you are having network problems and suspect a rogue computer is swamping your network
- e.g. a bad network card or a trojanized PC causing a denial of service attack
- First try running it wide open to see what is generating the most traffic.
- Use the `-a` and `-e` options to generate names and MAC addresses

```
tcpdump -ae
```

- If this causes the output to scroll off the screen too fast, use the `-c 1000` option to only count 1,000 packets and then stop.

### Monitor a Specific Workstation

- To log the traffic from a specific workstation to analyze later use `-w` switch to write to a file.
- `tcpdump -w logfile host 192.168.1.1`
- `logfile` is the file it will log to.
- May also use the `-c` or `-C` options to limit your output file size.

### Look for Suspicious Network Traffic

- If you are worried about what is happening on your network after hours, leave `tcpdump` running to flag traffic you might deem questionable.
- Run it with the `gateway <IP Address>` flag set, where IP address is that of your own Internet gateway (assuming you have access to that gateway)
- Assuming your network was in the IP Range of 192.168.0.0 through 192.168.0.254, this would flag any traffic coming or going from your Internet gateway.
- If you have an internal server and don't want to log that traffic since that would be valid traffic, add the statement: `and host != <address of server>`
- The exclamation point also acts as the Boolean `not` operator.

```
tcpdump -w logfile gateway 192.168.0.1 and host != 192.168.0.2
```

- If you are looking for users using a particular application, e.g. streaming video or audio program, specify that using port number.

```
tcpdump -w logfile gateway 192.168.0.1 and host != 192.168.1.2 and dst port 1000
```

### Look for abnormal traffic passing by or destined to your system

- To display start and end packets (SYN and FIN packets) of each TCP conversation involving a non-local host:

```
tcpdump 'tcp[13] & 3 != 0 and not src and dst net <localnet>'
```

- `<localnet>` is your local network
- Put in quotes so shell won't interpret punctuation characters
- To print traffic that does not have a local host as its source or destination:

```
tcpdump ip and not net <localnet>
```

- If your network has only one gateway between you and one other network, this traffic should *never* traverse your local network.
- To display IP broadcast or multicast packets that were not sent via Ethernet broadcast or multicast

```
tcpdump 'ether[0] & 1 = 0 and ip[16] >= 224'
```

- To display all ICMP packets that are not echo requests/replies (i.e., not ping packets)
- `tcpdump 'icmp[0] != 8 and icmp[0] != 0'`

## Ethereal

- Like tcpdump with a friendly graphical interface
- Offers many more analytical and statistical options.
- Output is much easier to read and understand than the raw packet captures of Tcpdump.
- Can interpret over 300 different network protocols, which covers just about every network type ever invented.
- More physical network formats are supported.
- Output can be saved as plain text or in Postscript format.
- A rich display filter mode. This includes the ability to highlight certain packets in color.
- There is a filter creation GUI to walk you through the process of creating filters easily.
- The ability to follow a TCP stream and view the content in ASCII.
- The ability to work with dedicated hardware
- The ability to save sessions in multiple formats.
- A command-line terminal mode. (so can run on servers without GUI - can dump file for analysis in GUI)
- Can also use serve as a general network analysis tool.

## Using Ethereal

- Choose Capture --> Start
- Choose Prepare
- Can do in real time or capture for later analysis

## Capture Options

- Interface — Picks the interface to capture from the pull-down menu.
- Ethereal automatically senses all the available interfaces and lists them.
- You can also choose to capture from all interfaces at once, just like Tcpdump
- Limit each packet to x bytes — Sets a maximum size for the packets captured.
- Use this if you fear some of the packets may be very large and you don't want to overload your machine.
- Capture packets in promiscuous mode — on by default.
- Filter — creates a filter using tcpdump-style expressions.
- Can name the filter so can the use in future sessions
- Capture file(s) — if want to read from a file rather than capture live data.
- Display options — these are disabled by default, but enable them if you want to watch the packets scroll by in real time.

- Not recommended for busy network or slow machine is slow,
- Capture limits — automatic stop
- After x number of packets or kilobytes of data have been captured,
- After x number of seconds have elapsed.
- Name resolution — specify whether you want Ethereal to resolve names at various levels of the network model.
- Enabling all of these, especially DNS, can slow down your capture significantly.

### **Packet List Window**

- The top third of the screen is where the packet stream is displayed in order of receipt,
- Can sort this in just about any way by clicking on the headings.
- Packet number — Assigned by Ethereal
- Time — The time the packet was received, set from the elapsed time from the start of the capture session.
- Alternately, this can be configured to show the clock time, the clock time and date, or even the time between packets (this is helpful for network performance analysis).
- Source address — Where the packet came from. This is an IP address on IP networks.
- Destination address — Where the packet is going to, also usually an IP address.
- Protocol — The level 4 protocol that the packet is using
- Info — Some summary information about the packet, usually a type field.

### **Packet Detail Window**

- Goes into more detail on each packet that is highlighted.
- Arranged in an order that basically conforms to the OSI model, so the first item listed is detail on the data link layer, and so on.
- The little pluses can be expanded to show even more information on each level. It is amazing how much detail you can see on each packet.

### **Packet Content Window**

- Actual packet contents, in both hexadecimal and translated into ASCII where possible.
- Binary files will still look like garbage, as will encrypted traffic, but anything in clear text will appear.
- Highlights the power (and danger) of having a sniffer on your network.

## Preferences

- Allows you to change layout
- Allows you to modify and add headings
- Allows you to adjust ports and such for protocols

## Display Filters

- Choose Expression
- Choose protocol and filter type
- Choose apply

## Ethereal Session Statistics Window

- Displays protocol statistic
- You can stop your session at any time by clicking Stop
- If you set a limit in the options, it will automatically stop when it reaches it.

## Analyze

- Follow TCP stream— shows connection traffic “conversation”

## Statistics

### tethereal

- `tethereal -w <filename>`
- `ethereal -r <filename>`

### ntop

- `ntop -u root`
- first time asks for password
- run in browser `http://localhost:3000/`