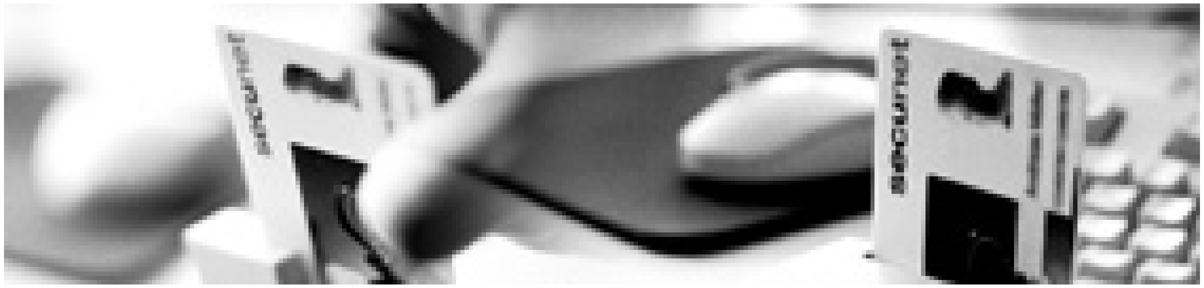


Network Flow Analysis

Tutorial

Nils Magnus
secu-CERT
secunet Security Networks AG
We create confidence

Forum of Incident Response Teams
Annual General Meeting 2004
Budapest, Hungary, June 12 – 18, 2004

Overview

- Network flow analysis: Why and when?
- Trends vs. flows

- Introduction in sniffing techniques
- Important tools

- How to actually use the tools in daily work
- Other fields that deal with sniffing

- Case studies with actual data

Part I:

Why and how to analyze network flows

Intended Audience

- Possibly not interesting for beginners:
  - I won’t explain basic networking
  - Prior understanding of TCP/IP and OSI-Layers is helpful
- Might be boring for experts:
  - No new technology is presented
  - All this has been done before

- Advanced users that have understood the theory but lack practise
  - Practical and efficent use in all our day jobs

- Exchange of experience, not necessarily expertise
- Intended as an interactive workshop: Ask questions!

Why am I doing this?

- About me …
  - Nils Magnus
  - based in Hamburg, Germany
  - working throughout the world
  - Senior consultant and team leader network security
- … and my company
  - 220 employees consulting in all fields of information technology
  - secu-CERT with a small constituency
  - Explaining how to do incident managemen
  - Technical expertise
  - Process oriented security management
- Fields of work and interest
  - Penetrations testing
  - Post mortem analysis
  - Forensics

Network Flow Analysis

- Time of action
  - Realtime analysis of what is currently happening
  - Post-mortem analysis of what did happen
- Interested in general state of the network:
  - TCP/5000 increases rapidly: A new worm?
  - Outgoing packets are directed to non-assigned networks: New scanning technique?
  - Mail traffic tenfolds: New waves of spam?
- Interested in a specific connection:
  - What data sent the worm to exploit the system?
  - What header flags actually made the firewall break or leak?

Overview vs. Details

- Several tools allow you get an idea about the nature of traffic:
  - Cisco/IOS netflows
  - Argus suite
  - other AGM and TC presentations covered that already

- Sometimes you need more data
- Sometimes you need to look inside the header
- Sometimes you need to look inside the packet payload
- Sometimes you have to correlate data from different packets

- We focus on investigating single packets
- Inspection of packet „off the wire“ is necessary

Legal Disclaimer

- Eavesdropping of data may be forbidden by law (e.g. StGB §203a in Germany)
- Precondition is that data is “suitably protected”
- There are way too little precedences
- Privacy issues may have to be considered

IANAL

- Ethics (a correlating, but not identical category compared to law)
  - after previous consent of the owner or operator
  - may be even more tricky when it comes to “public” networks

The Name of the Game

- How (and where) to get hold of the actual data?
  - at the target application: difficult if not open source
  - at the target operation system: difficult in production environments
  - at some relaying network components: possible, but often inintuitive and awkward
- Solution: passive „sniffing“ of packets
- Universal approach: no components involved in a communication need to be touched
- „Sniffing“ may sound „undergroundish“: Better suggestions?

Foundations of Sniffing

- Most popular network technology (in local networks) today is Ethernet
- CSMA/CD: Carrier Sense Multiple Access, Collision Detection
- Most important for Sniffing: Multiple Access
- All participants at the segment access the same medium

- In terms of the OSI model: link layer (layer 2)
- Generally every NIC sees every packet

Data Exchange on Layer 2

- Usually only such packets are considered by a systems, that has as destination address the address of the system’s own MAC
- A MAC address (medium access control) is easy to forge

```
# ifconfig eth0 down
# ifconfig eth0 ether hw aa:ff:ff:ee:00:11
# ifconfig eth0 up
```

- Thus sender and receiver are easy to forge
- Switches may make this a little more difficult, but this is another story …

Make Data Pass by

- It is crucial that all data passes the monitoring NIC
- For maximum transparency use a tapping device:
  - copies the very frame into a new segment only coupled by fiber optics
- Convenient way with many switches: Monitor port
  - copies some or all data to an special assigned port
- Sometimes only software solutions are the only option: ettercap
  - Uses ARP-spoofing techniques to divert traffic
  - Is not completely transparent

Monitoring the Medium

- Most (all) operating systems allow packets (precisely: frames) to be sent to the userspace of the operating system for inspection
- Promiscous Mode (is a flag of the interface structure)

- As per OS, packets can be picked up directly from the NIC: complex
- Today’s defacto interface: libpcap (Packet Capture Library)
- Simple command line tool for libpcap: tcpdump

- Result:

  Every passing packet can be analyzed or stored for later inspection

tcpdump

- Maps most library call of the libpcap 1-on-1 to a command shell tool:
  - switching interfaces into promiscous mode
  - storing packets in files using a “standard format”
  - Filtering by a number of packet properties (performance!)
  - very basic display of packet contents
- Several comparable tools are available: sniffit, snoop, several third-party products (NetXRay, Lananalyzer, …)
- Most tools create capture files or can be used to read them

Limitations of tcpdump

- Advantage: simple and reliable
- Advantage: useful to actually monitor a medium

- Disadvantage: visualization is unsatisfactory for complex tasks
- Disadvantage: correlation over packet boundaries is not (easily) possible

- My suggestion: Save now, analyze later

Ethereal

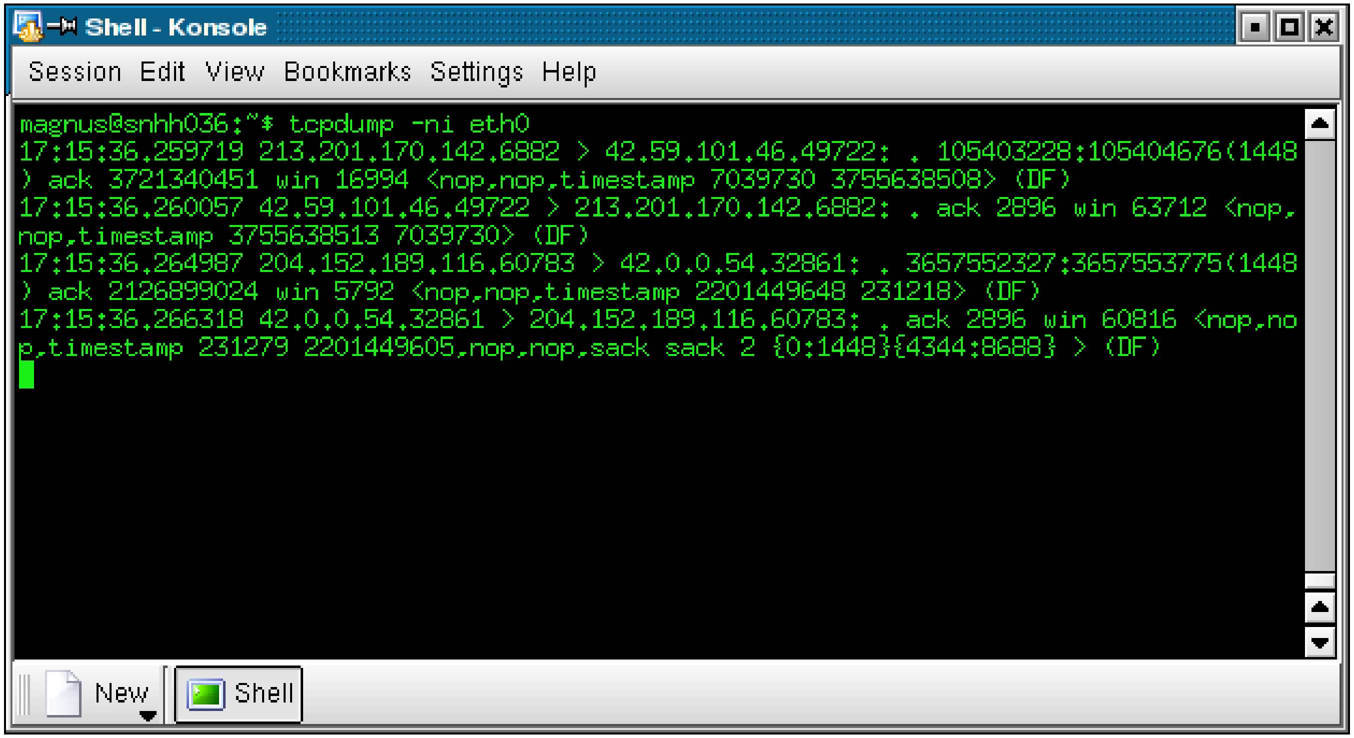
- One of not so many tools, the GUI actually means a benefit and a functional enhancement
- Ethereal is much more than a graphical tcpdump:
  - semantical analysis of packet payloads (“dissectors”)
  - correlation of packets, e. g. reassembly of TCP streams
  - traffic analysis, e. g. a treeview of used protocols
- Installation is quite simple, is based on gtk
- Available for most relevant platforms and Windows
- Suggested procedure:
  - capturing with tcpdump
  - analyzing with Ethereal

Part II:

Important tools and techniques

Using tcpdump

- Don‘t use DNS
- Capture interface
- Output
- Addresses are faked



```
magnus@snhh036:~$ tcpdump -ni eth0
17:15:36.259719 213.201.170.142.6882 > 42.59.101.46.49722: . 105403228:105404676(1448
) ack 3721340451 win 16994 <nop,nop,timestamp 7039730 3755638508> (DF)
17:15:36.260057 42.59.101.46.49722 > 213.201.170.142.6882: . ack 2896 win 63712 <nop,
nop,timestamp 3755638513 7039730> (DF)
17:15:36.264987 204.152.189.116.60783 > 42.0.0.54.32861: . 3657552327:3657553775(1448
) ack 2126899024 win 5792 <nop,nop,timestamp 2201449648 231218> (DF)
17:15:36.266318 42.0.0.54.32861 > 204.152.189.116.60783: . ack 2896 win 60816 <nop,no
p,timestamp 231279 2201449605,nop,nop,sack sack 2 {0:1448}{4344:8688} > (DF)
```

Limitations of tcpdump

- Performance
  - capturing, processing and storing of packets need some time
  - fully saturated segments (upstream exchange points) may not be completely covered
  - however, nowadays hardware is quite fast: effectively only a problem in high-end places
  - solution: filtering
- Visualization
  - Too many packets: “interesting” data may get lost
  - A lot of protocol know-how is necessary to figure out the meaning of all those values
- Postprocessing is quite difficult once the data is on stdout

Filtering

- libpcap provides powerful and flexible filter capabilities
  - for protocols of several layers (ip, tcp, ...)
  - for adresses (192.168.42.23 or 00:04:aa:bb:cc:dd)
  - directions (src or dst)
  - Network and netmask (net 172.16)

  - and generally for any payload at arbitrary offsets

Filter Syntax

- Structured though complex syntax
  - expression: logical combined primitives (and/or/not, braces)
  - primitive: id + qualifier
  - id: adress or value (“192.168.42.23”)

– qualifier:	type or	direction or	protocol
– example	net 192.168	dst 10.10.10.10	ip

- Several abbreviations and special rules exist

- Powerful filter rules are possible

Filter Examples

- All outgoing traffic from our system (192.168.47.11):

```
# tcpdump src host 192.168.47.11
```

- Any traffic directed to our webserver:

```
# tcpdump dst host and tcp and port 80
```

- Effectivly all link layer traffic

```
# tcpdump not ip
```

Important tcpdump Options

- Capture whole packet payload: -s 2048
- Save all packets in a file: -w file
- No annoying DNS lookups: -n

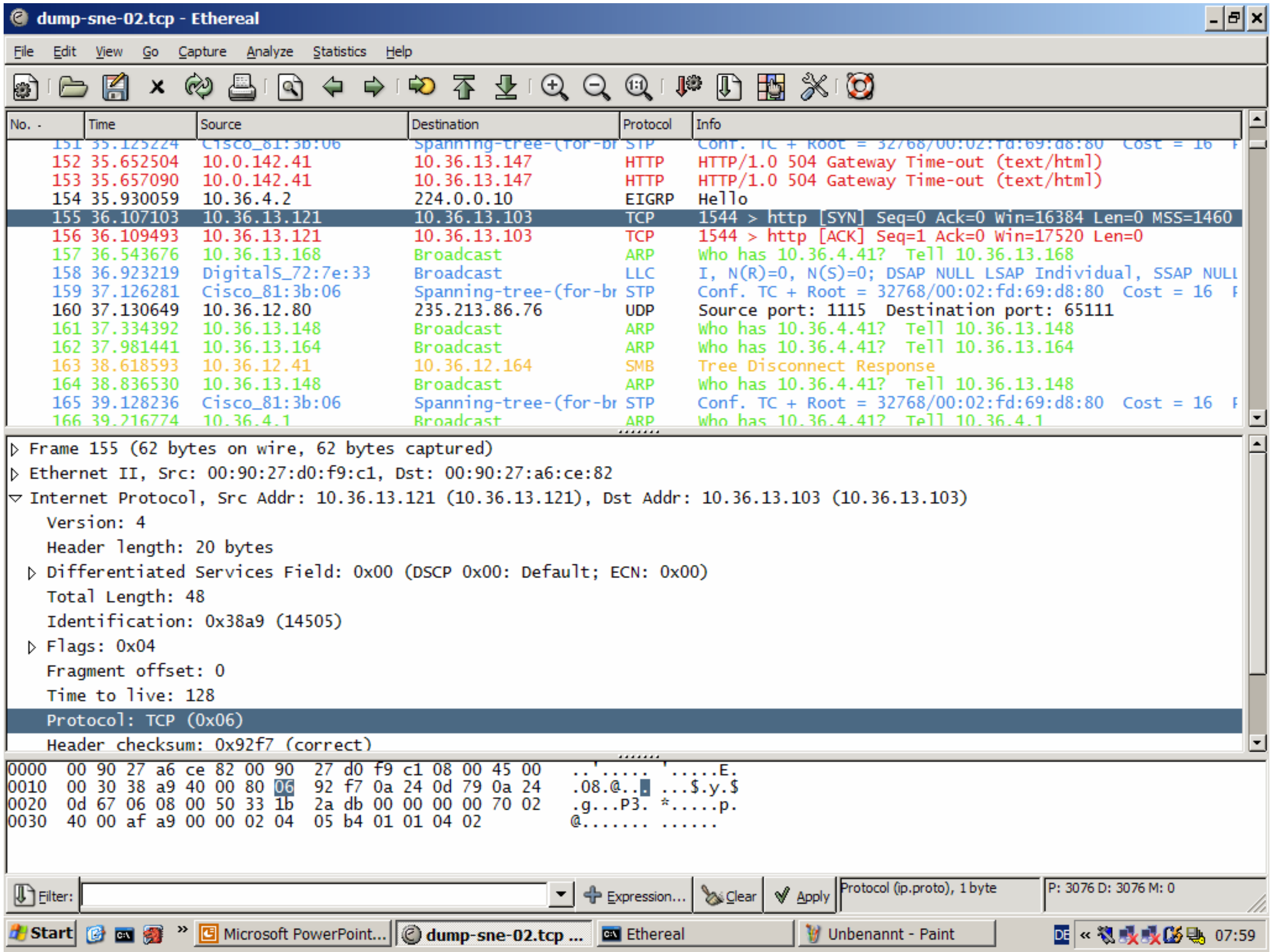
- Filter make sense if vast amounts of data are on the segment
- Example: “ip” or “not icmp” or “not tcp port 139”

- Don’t make filters too specific

- Dumps can be fed into other tools as well:
  - snort
  - dsniff
  - Ethereal

Ethereal

- List of packets
- Packet details (protocol)
- Packet dump



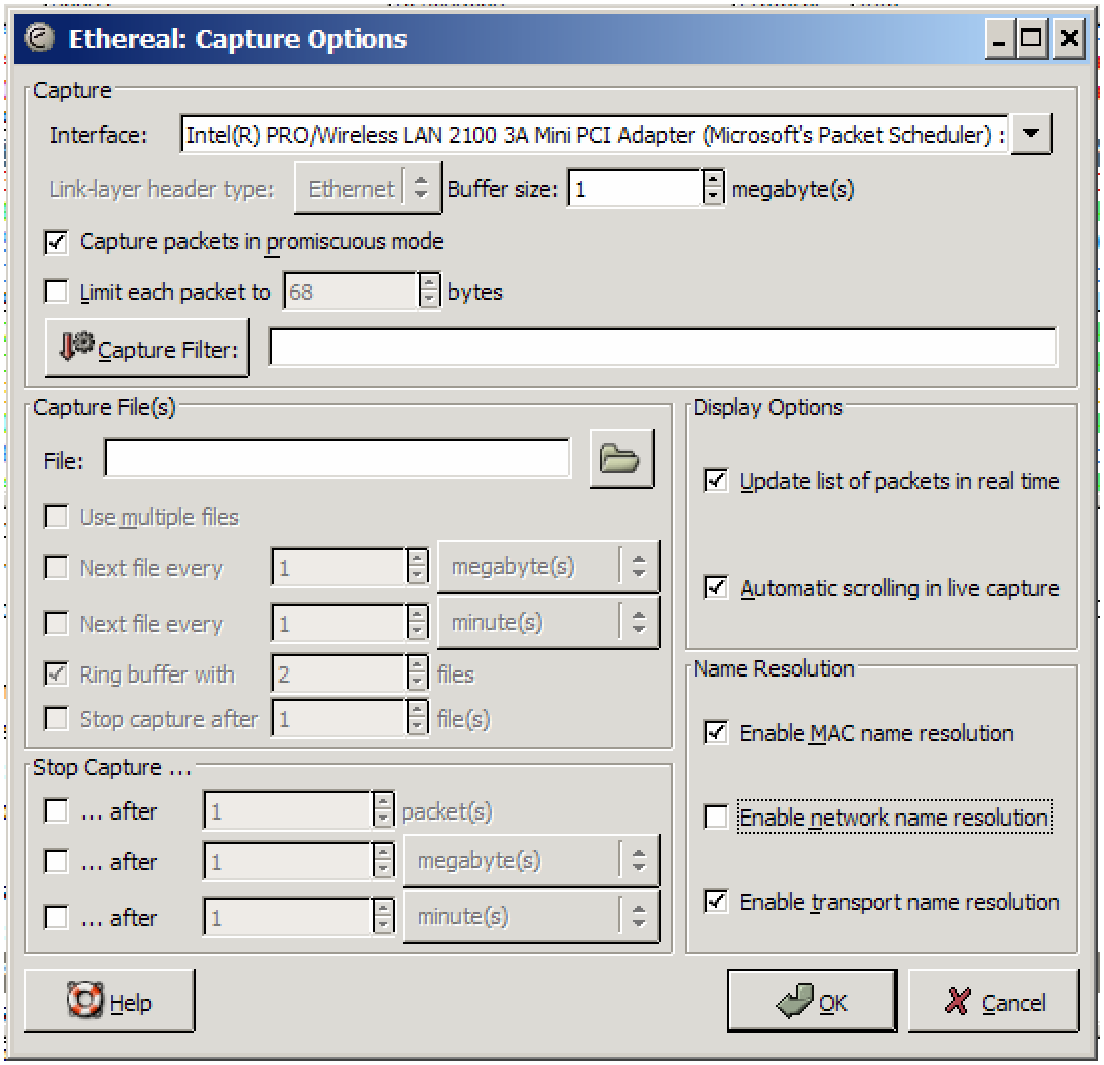
```
No. -  Time       Source              Destination            Protocol Info
151 35.125224  Cisco_81:3b:06      Spanning-tree-(for-br  STP      Conf. TC + Root = 32768/00:02:fd:69:d8:80  Cost = 16  P
152 35.652504  10.0.142.41         10.36.13.147           HTTP     HTTP/1.0 504 Gateway Time-out (text/html)
153 35.657090  10.0.142.41         10.36.13.147           HTTP     HTTP/1.0 504 Gateway Time-out (text/html)
154 35.930059  10.36.4.2           224.0.0.10             EIGRP    Hello
155 36.107103  10.36.13.121        10.36.13.103           TCP      1544 > http [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
156 36.109493  10.36.13.121        10.36.13.103           TCP      1544 > http [ACK] Seq=1 Ack=0 Win=17520 Len=0
157 36.543676  10.36.13.168        Broadcast              ARP      Who has 10.36.4.41?  Tell 10.36.13.168
158 36.923219  Digitals_72:7e:33   Broadcast              LLC      I, N(R)=0, N(S)=0; DSAP NULL LSAP Individual, SSAP NULL
159 37.126281  Cisco_81:3b:06      Spanning-tree-(for-br  STP      Conf. TC + Root = 32768/00:02:fd:69:d8:80  Cost = 16  P
160 37.130649  10.36.12.80         235.213.86.76          UDP      Source port: 1115  Destination port: 65111
161 37.334392  10.36.13.148        Broadcast              ARP      Who has 10.36.4.41?  Tell 10.36.13.148
162 37.981441  10.36.13.164        Broadcast              ARP      Who has 10.36.4.41?  Tell 10.36.13.164
163 38.618593  10.36.12.41         10.36.12.164           SMB      Tree Disconnect Response
164 38.836530  10.36.13.148        Broadcast              ARP      Who has 10.36.4.41?  Tell 10.36.13.148
165 39.128236  Cisco_81:3b:06      Spanning-tree-(for-br  STP      Conf. TC + Root = 32768/00:02:fd:69:d8:80  Cost = 16  P
166 39.216774  10.36.4.1           Broadcast              ARP      Who has 10.36.4.41?  Tell 10.36.4.1
```

```
▷ Frame 155 (62 bytes on wire, 62 bytes captured)
▷ Ethernet II, Src: 00:90:27:d0:f9:c1, Dst: 00:90:27:a6:ce:82
▽ Internet Protocol, Src Addr: 10.36.13.121 (10.36.13.121), Dst Addr: 10.36.13.103 (10.36.13.103)
    Version: 4
    Header length: 20 bytes
  ▷ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 48
    Identification: 0x38a9 (14505)
  ▷ Flags: 0x04
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (0x06)
    Header checksum: 0x92f7 (correct)
```

```
0000  00 90 27 a6 ce 82 00 90  27 d0 f9 c1 08 00 45 00   ..'..... '.....E.
0010  00 30 38 a9 40 00 80 06  92 f7 0a 24 0d 79 0a 24   .08.@... ...$.y.$
0020  0d 67 06 08 00 50 33 1b  2a db 00 00 00 00 70 02   .g...P3. *.....p.
0030  40 00 af a9 00 00 02 04  05 b4 01 01 04 02         @....... ......
```

Capturing Data

- Options resemble command line switches at tcpdump
- Select interface
- Real time display
- Decoding of addresses



Ethereal: Capture Options

Capture

Interface: Intel(R) PRO/Wireless LAN 2100 3A Mini PCI Adapter (Microsoft's Packet Scheduler) :

Link-layer header type: Ethernet Buffer size: 1 megabyte(s)

☑ Capture packets in promiscuous mode

☐ Limit each packet to 68 bytes

Capture Filter:

Capture File(s)

File:

☐ Use multiple files

☐ Next file every 1 megabyte(s)

☐ Next file every 1 minute(s)

☑ Ring buffer with 2 files

☐ Stop capture after 1 file(s)

Display Options

☑ Update list of packets in real time

☑ Automatic scrolling in live capture

Name Resolution

☑ Enable MAC name resolution

☐ Enable network name resolution

☑ Enable transport name resolution

Stop Capture ...

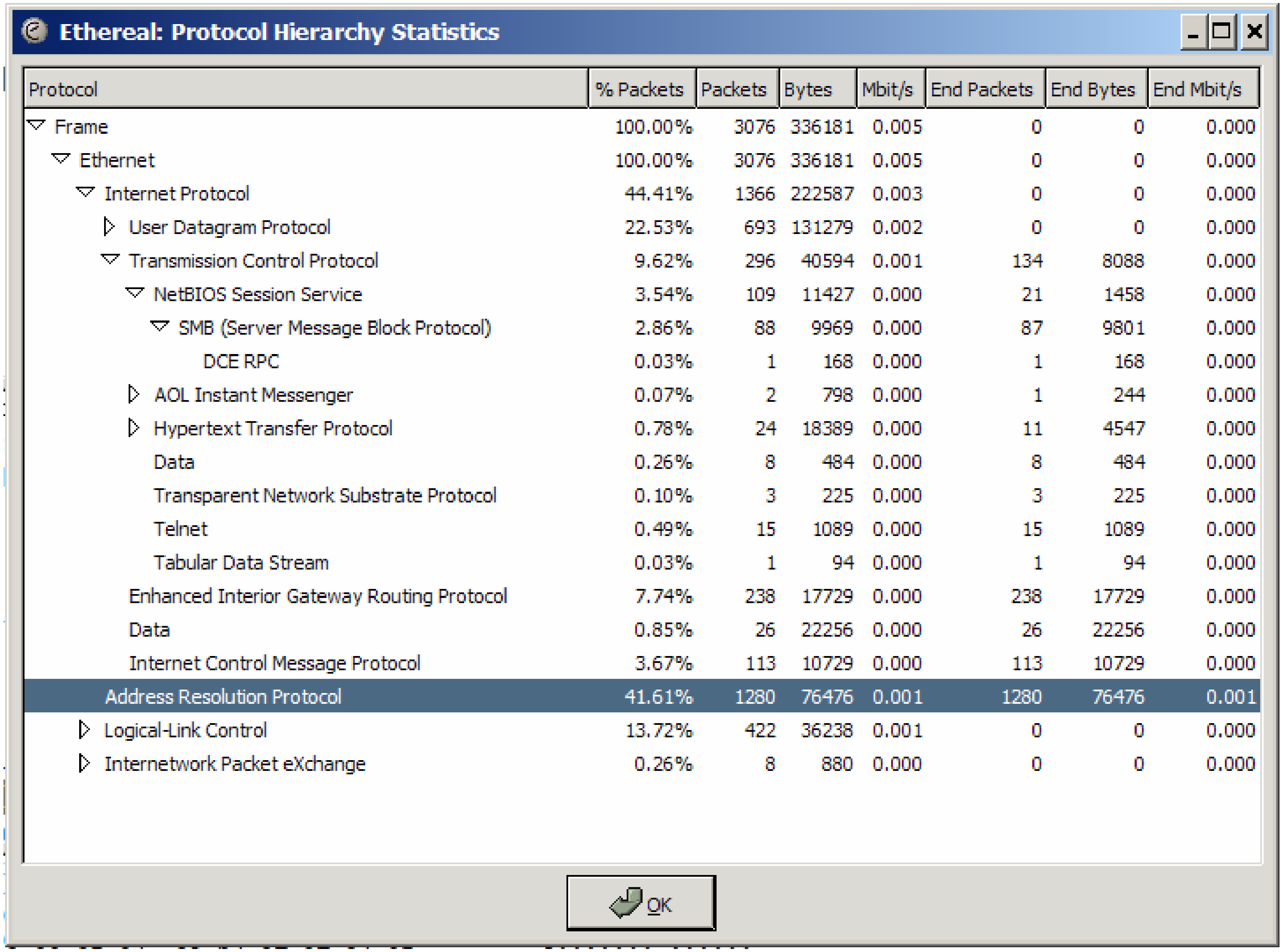
☐ ... after 1 packet(s)

☐ ... after 1 megabyte(s)

☐ ... after 1 minute(s)

Statistics

- Treeview
- Protocol distribution
- Several custom statistics



Ethereal: Protocol Hierarchy Statistics

Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
▽ Frame	100.00%	3076	336181	0.005	0	0	0.000
▽ Ethernet	100.00%	3076	336181	0.005	0	0	0.000
▽ Internet Protocol	44.41%	1366	222587	0.003	0	0	0.000
▷ User Datagram Protocol	22.53%	693	131279	0.002	0	0	0.000
▽ Transmission Control Protocol	9.62%	296	40594	0.001	134	8088	0.000
▽ NetBIOS Session Service	3.54%	109	11427	0.000	21	1458	0.000
▽ SMB (Server Message Block Protocol)	2.86%	88	9969	0.000	87	9801	0.000
DCE RPC	0.03%	1	168	0.000	1	168	0.000
▷ AOL Instant Messenger	0.07%	2	798	0.000	1	244	0.000
▷ Hypertext Transfer Protocol	0.78%	24	18389	0.000	11	4547	0.000
Data	0.26%	8	484	0.000	8	484	0.000
Transparent Network Substrate Protocol	0.10%	3	225	0.000	3	225	0.000
Telnet	0.49%	15	1089	0.000	15	1089	0.000
Tabular Data Stream	0.03%	1	94	0.000	1	94	0.000
Enhanced Interior Gateway Routing Protocol	7.74%	238	17729	0.000	238	17729	0.000
Data	0.85%	26	22256	0.000	26	22256	0.000
Internet Control Message Protocol	3.67%	113	10729	0.000	113	10729	0.000
Address Resolution Protocol	41.61%	1280	76476	0.001	1280	76476	0.001
▷ Logical-Link Control	13.72%	422	36238	0.001	0	0	0.000
▷ Internetwork Packet eXchange	0.26%	8	880	0.000	0	0	0.000

Reassembly of Data Streams

Live Demonstration

Part III:

Case studies

Case Study: Discovery of the Neighborhood

- Typical question: Where am I?
  - what IP range is used?
  - which addresses are potentially free and can be used?
  - what are the subnet masks?
  - which routers route where? Are there redirects?
- Objectives:
  - is it possible to participate in the network?
  - find out the network topology
  - investigate the network structure (trunking, VLANs, Etherchannel, …)

Case Study: Profiling the Network Usage

- Typical question: What’s going on here?
  - what kind of network is this? A university, an office, a core network segment, a hotspot?
  - what kind of network architecture and technology is used? TCP/IP? Novell? Strange Protocols?
  - what services are used? Windows desktops yelling around? Unencrypted services? Network Management?
  - which operating systems are being used?
- Objectives:
  - traffic analysis
  - identify potential past and future targets for attackers

Case Study: Eavesdropping

- Typical question: What are they doing over there?
  - capture and monitor connections of ongoing attacks
  - passive or active sniffing
  - reassemble transmission content
  - collect passphrases or authentication tokens
  - protocol analysis
- Objectives
  - understand new or unknown protocols
  - discover vulnerabilities
  - extract specific data

Case Study: Application Analysis

- Why not sniff yourself?
- Typical question: What is this new program doing, by the way?
  - covered license registrations
  - loss of privacy
  - leaking of sensitive data
  - backdoors, hidden channels
  - also: detecting and watching ongoing attacks
- “Real hackers sniff their own network 24/7”
- Watch packet TV
- Script kiddie watching

Active and Passive

- Traditional sniffing can mostly be done passivly
- Passive sniffing is hard (impossible?) to detect

- Sometimes special events need to be triggered
- Probe certain services or reactions by sending your own packets:
  - ping and other ICMP requests (flags, sequnce numbers)
  - traceroute with UDP, ICMP and TCP (TTL and other fields)
  - Netcat and watch the replies by the octett
  - nemesis, sing, Perl::Net, ...

Training Cases

- Good exercise: sniff a portscan
  - explain every single packet
  - how detects nmap active (“up”) systems?
  - how does nmap find out about the operating system?
  - is the documentation correct?
- Preparing and performing a black box security analysis
  - prepare and describe a use case
  - capture all traffic while exercising the use case
  - explain every packet
  - now, look for potential vulnerabilities

Part IV:

Wrap up and a look ahead

Limits and Other Tools

- Switches
  - ARP spoofing is sometimes necessary
  - easier: grab packets directly from monitor port (beware of network management!)
  - ettercap + Ethereal together are very poweful (beware of duplicated packets!)
- Utilities
  - dsniff simplifies collecting of “interesting data”
- VLANs and trunking
  - VLAN tagging by means of IEEE 802.1Q etc.
- Special scenario WLAN
  - additional IEEE 802.11*x* wrapper around the every frame
  - airsnort, WEPcrack etc.

Mistakes to Avoid

- MAC address cannot be changed:
  - wrong!
- Parsing text output of tcpdump:
  - you will lose valuable data, better use the filter capabilities
- Capture only packet headers (forget –s switch):
  - payload might be interesting later
- Too restrictive filters
  - to narrow in is always possible, not the other way around
- Look out only for IP and above
  - a lot of nasty stuff can be done on the link layer and with "strange protocols"

Questions,

Comments,

Discussion

Instructor

Dipl.-Inform. Nils Magnus
Senior-Consultant IT-Security
Teamleader Network Security

secunet
Security Networks AG
Osterbekstr. 90b
22083 Hamburg, Germany

Phone +49 40 69 65 99 - 13
Fax +49 40 69 65 99 - 29
Mail magnus@secunet.de
http://www.secunet.com/