I12: Capturing a packet -
from Ether and Wire to Wireshark

Jörg Mayer
<mayer@fg-networking.de>
Online version of Sharkfest talk

The slides of this talk are © 2014 Jörg Mayer

Licensed under CC-BY-SA 3.0

https://creativecommons.org/licenses/by-sa/3.0/
Introduction

Jörg Mayer

1st sniffer: Etherfind on SunOS (1992)

Ethereal user since 1998

1st patch submitted 1998

Core Developer

Dayjob:

Network Consultant (Design, Implementing, Troubleshooting)
Content

• Part 1
  
  Capturing data on the wire

  Passing the OS

  dumpcap and wireshark

• Part 2

  What's different with wireless

  OS

  Capture
Data Capture

3 Scenarios how data is captured

( ultra short version, there are whole talks just about this)

- Shared media
- Mirror ports
- Taps
Data Capture

- Shared/BUS
  - All network participants may see each other’s traffic

![Diagram showing network nodes A and B connected through L2/L3 layer with Sniffer-PC and Sniffer Software.](image-url)
Data Capture

• Tap
Data Capture

- Mirror port
Data Flow
Data Flow

Destination MAC Filter
- In Hardware
- Filters Unicast and Multicast
- Disable by activating "promiscuous mode"
Ethernet chip „surprises“

- always filters errored frames
- fcs often missing
- mac filtering (turn off via „promiscous mode“)
- vlan tagging offloading
- ip/udp/tcp checksum offloading
- generic/udp/tcp segmentation offloading
- link-pulse, autonegotiation invisible
Data Flow

- Sometimes disable additional hardware Feature VLAN offloading
- Additional surprises: IP/UDP/TCP checksum offloading and TCP segmentation

Computer

Kernel

Driver

NIC
Driver (not Windows)

- adds metadata (some of)
- timestamp
- direction
- packet size
- capture size
- encapsulation type
Data Flow

Capture Filter
Surprises:
Data reaches
TCP/IP as well
Sidetrack: NDIS

Windows „driver framework“

(ultra short version)

• Drivers and IP-Stack hook into it

• WinPcap uses NDIS version 5

• Responsible for many „features“ of windowscapture
Sidetrack: NDIS „surprises“

- No capture on Loopback, ppp, vpn interfaces
- „random“ placement in the chain of other clients
- Firewalls
- Virus checkers
- VPN
Data Flow

Compilation into BPF/NPF filters

Capture filter

Computer

libpcap/wpcap + packet.dll

BPF NPF

IPX

TCP/ IP

Driver

NIC

Kernel

Sharkfest 2014
More on packet.dll (Windows only)

Compensates for missing stuff in kernel/drivers

(ultra short version)

• Provides NPF

• Provides timestamps (and other metainfo)

Timestamps on Windows are way more inaccurate than on Unix'ish systems
Data Flow

- Programs capture filters as text
- *dumpcap*
- *libpcap/wpcap + packet.dll*
- BPF
- NPF
- IPX
- TCP/IP
- Driver
- NIC

Sharkfest 2014
Data Flow

Computer

wire
gatt
raw

SOME
shark

Control
Channel
(pipe)

Capture
Data

File

Applications

dumpcap

libpcap/wpcap + packet.dll

BPF
NPF

IPX

TCP/IP

Driver

NIC

Sharkfest 2014
dumpcap

The program that does (almost) nothing

- Asks the kernel to capture (often requires special privileges)
- Adds pcapng header to packet and writes to file
- Signals Wireshark that new data is available
Data Flow

Sharkfest 2014
Data Flow

Virtualization: VMWare/...

vSwitch: allows client NIC to use promiscuous mode
Inside Wireshark

Read filters

The packets Wireshark **knows** about

Command line syntax: `-R <read filter>`

Same syntax as display filters

Display filters

The packets Wireshark **shows**

Command line syntax: `-Y`
Inside Wireshark

Filtering is expensive (CPU, memory)

Read filters easily filter too much:

- IP fragments relevant to your protocol
- TCP packets without payload relevant to your protocol
- There are talks about what can go wrong here
Part II

Wireless
Wireless works the same

if we only

● do cooked capture

(frames look like Ethernet frames)

● don't turn on promiscous on Windows

(some Windows drivers do „interesting“ things)
... well - almost the same

cooked capture means

• replace IEEE802.11 header by Ethernet_V2 header

• we only see traffic as we would do on a switch
  
  Broadcasts, Multicasts, Unicasts to/from us

• data frames only
  
  no wireless control, management or eapol frames (4-way-handshake)

• already decrypted
Surprises with WLAN captures

- No packets captured at all
- Are you on Windows?
- Have you promiscous mode turned on?
- I only see my own traffic
- Maybe you are actually looking at cooked traffic?
- What is the L2-Header: 802.11 or Ethernet?
Surprises with WLAN captures

● No machines visible or only traffic in one direction
  ● Are you on the same channel/band that they are on?
  ● Is the „invisible“ machine a „hidden station“ (AP can see station but we can't)?
  ● Special case of hidden station: Incompatible antenna (polarization)
● I see traffic from machines not on my channel
  ● A channel is 5 MHz wide, a signal is 22(b), 20 (a/g), 20/40 (n), 20/40/80/160(ac) MHz wide. So we see neighboring traffic as well.
Surprises with WLAN captures

- Some packets are missing
- Channel hopping sniffing software (e.g. kismet)
- Rarely: AP changing channel (DFS: regulator, ACS: optimize)

- Some of ACK, RTS, CTS frames are missing
  - Some (mostly older chips) process these frames and can't forward them to the driver (reduced raw capture)
Surprises with WLAN captures

• No data frames or multicast/broadcast frames only

• The capture hardware is too old (b < g < n2, a < n5 < ac)

• The capture hardware supports not enough streams (n, ac)

• Lots of corrupted but ACK'ed frames

• Sniffer close to interference source
Surprises with WLAN captures

- Decryption of WPA/WPA2 doesn't work
  - raw capture required AND 4-way-handshake capture required AND no 802.1X
  - Wireshark: pwd vs. psk

- Network not visible but active clients
  - Hidden SSID
  - deactivated beacons
Thanks to....

• Gerald Combs for providing me with a hobby that has lasted for close to 16 years

• Janice and all the other helpers: It has been a great time

• Riverbed for sponsoring

• The Wireshark community
THANKS for listening!

Questions?