Dissecting Wireshark

A Case Study on Network Anti-Forensics

Ben Schmidt (Presenter) // @_supernothing
Lord Commander of Security Research @NarfIndustries

Paul Makowski // @myhndl
Director of World Domination @NarfIndustries

NARF INDUSTRIES

this talk

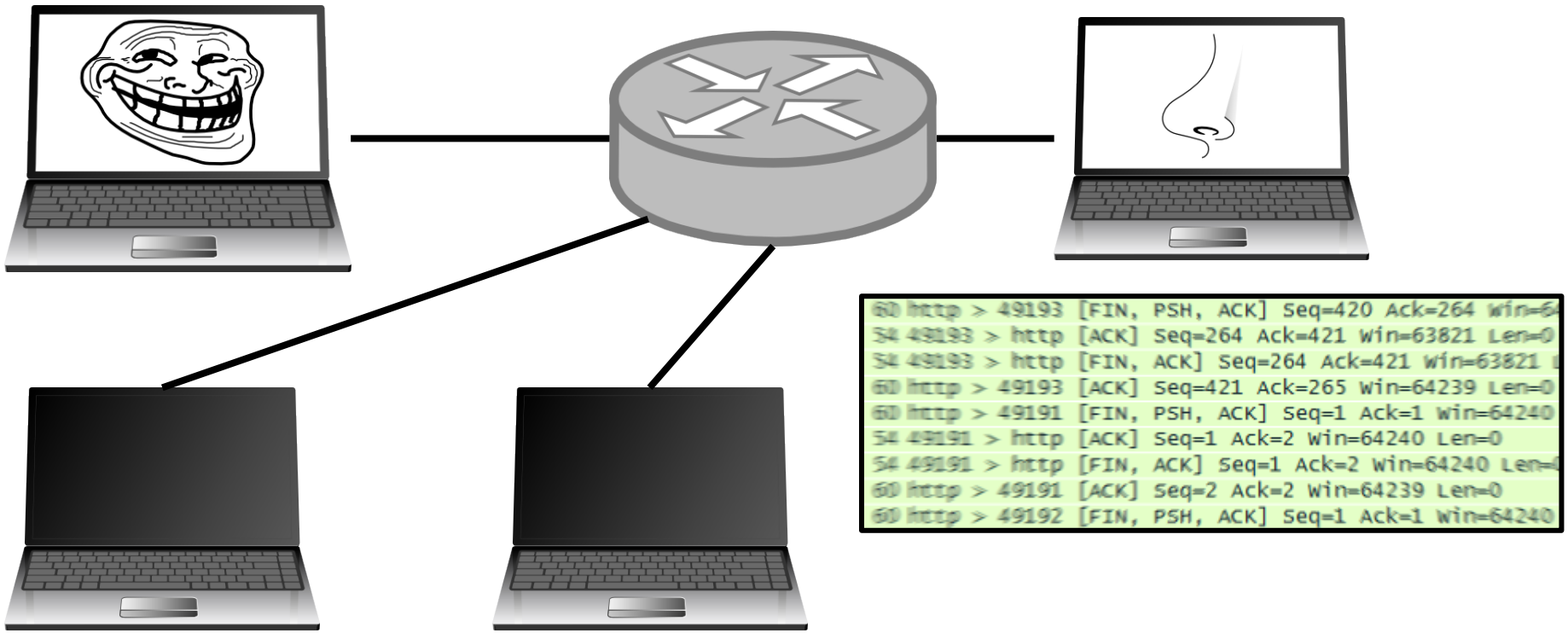
- background & why you should care
- life of a packet
- dissector overview
- packet reconstruction example
- some lame DoS & how to find your own
- RCE affecting default heuristic dissector
- mitigations & recommendations

background // why you should care

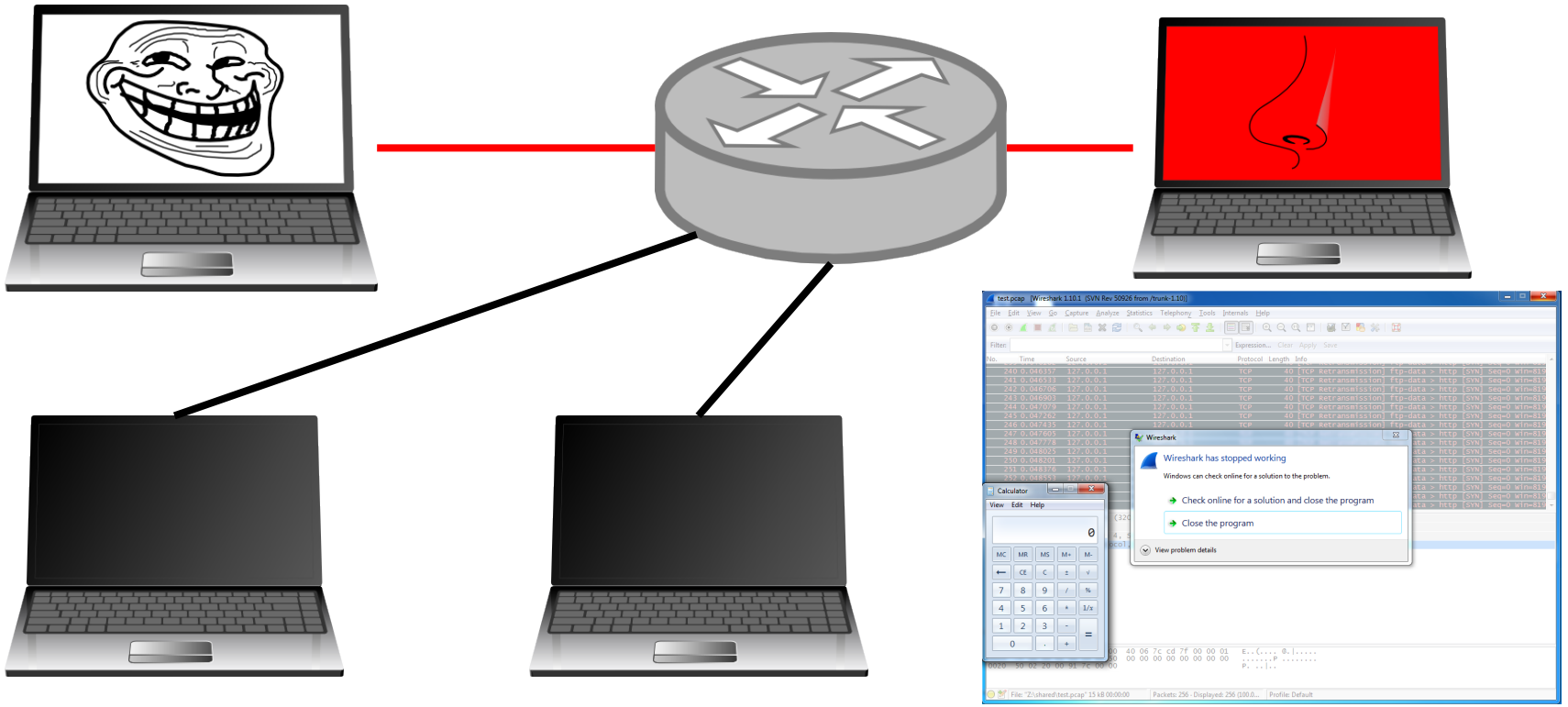
- Wireshark is **the** packet analysis tool
- used everywhere
- breaking it lets you:
  - hide traffic
  - hinder analysis
  - hack analysts
  - hurdle airgaps
- what alternatives are there?

background // Samurai F.U.D. Team

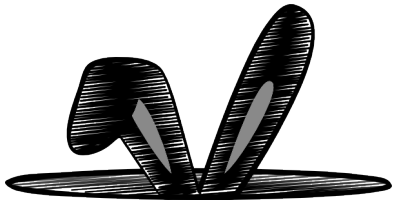
- Fear, Uncertainty and Doubt
- started at DEFCON 20
- led by Paul Makowski & Ben Schmidt
- *task:* **protect legit 'sploits & implants**
- *solution:* **pop Wireshark & related tools**

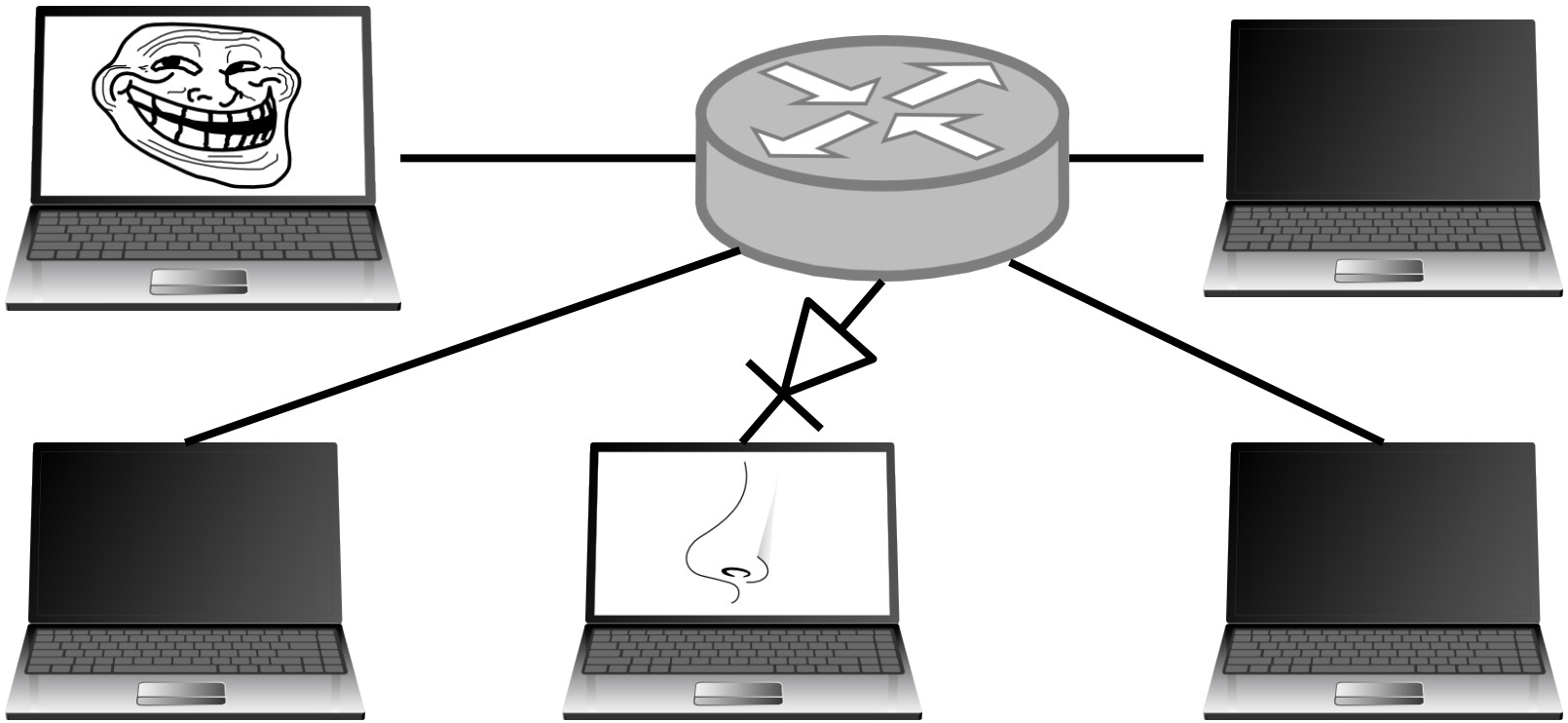


spreading F.U.D. to workstations

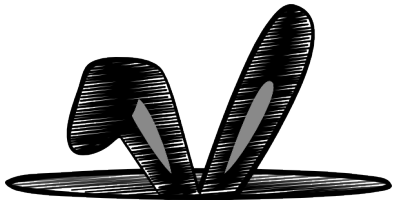


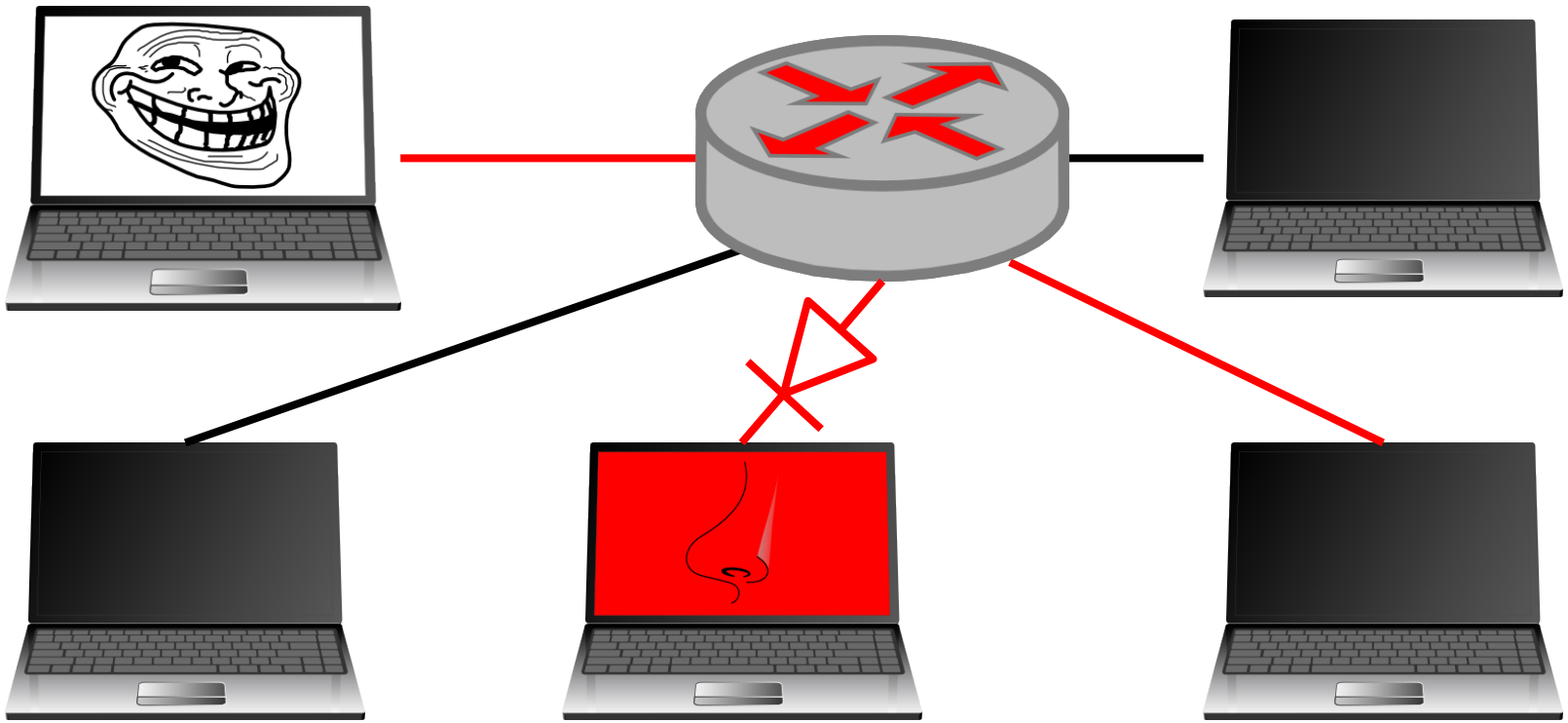
spreading F.U.D. to workstations





spreading F.U.D. to passive sniffers

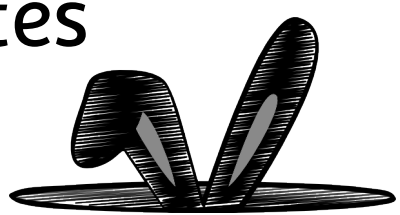




spreading F.U.D. to passive sniffers

life of a packet (1 of 2)

1. user asks [libp|WinP]cap to listen and tells it what to filter out
   a. Linux: a BPF is JITted in kernelspace
   b. Windows: filter installed in kernelspace (npf.sys)
2. packets pass monitor network interface
3. kernel passes packet to [libp|WinP]cap
4. [libp|WinP]cap ignores or duplicates packet to listening user process



life of a packet (2 of 2)

5. Wireshark reconstructs traffic from lowest to highest layer
    a. at each layer, Wireshark determines which dissector is responsible for handling the data
    b. higher-level candidates dependant on lower-level decisions
    c. dissectors tell Wireshark what data they care about... and can be pretty promiscuous

dissectors

(P)IDL

THE C PROGRAMMING LANGUAGE

ASN.1

dissectors

*“Note: **This is currently broken** (see commits 49066, 49138).”*

http://wiki.wireshark.org/Python

*“...make it harder to enable Python: change it from "--with-python" to ‘**--with-broken-python**" just to prevent people from enabling unless they really mean it (are going to work on fixing it).”*

https://anonsvn.wireshark.org/viewvc?revision=49138&view=revision

dissectors

“Although it's possible to write dissectors in Lua, Wireshark dissectors are written in C, as C is several times faster than Lua. ***Lua is ok for prototyping dissectors****, during Reverse Engineering you can use your time for finding out how things work instead of compiling and debugging your C dissector.”*

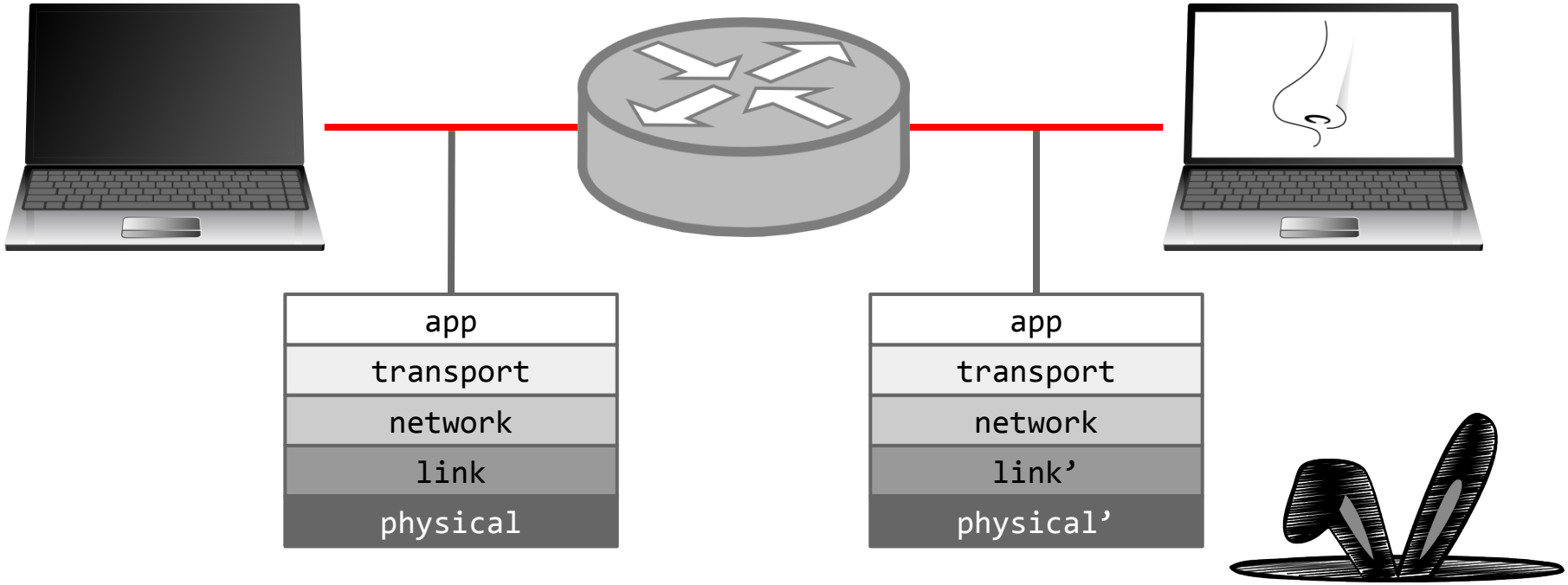
http://wiki.wireshark.org/Lua

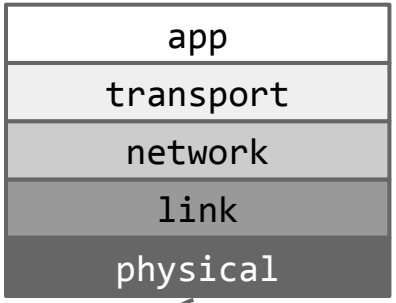
dissectors

:D

http://www.phrack.org/

example traffic reconstruction





thrown out by network card

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00  ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad  . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18  ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32  ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43  04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30  ontent-Length: 0
0060   0d 0a 44 61 74 65 3a 20 57 65 64 2c 20 30 32 20  ..Date: Wed, 02
0070   4a 75 6c 20 32 30 31 34 20 32 32 3a 33 32 3a 31  Jul 2014 22:32:1
0080   33 20 47 4d 54 0d 0a 43 6f 6e 6e 65 63 74 69 6f  3 GMT..Connectio
0090   6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 50  n: keep-alive..P
00a0   72 61 67 6d 61 3a 20 6e 6f 2d 63 61 63 68 65 0d  ragma: no-cache.
00b0   0a 45 78 70 69 72 65 73 3a 20 4d 6f 6e 2c 20 30  .Expires: Mon, 0
00c0   31 20 4a 61 6e 20 31 39 39 30 20 30 30 3a 30 30  1 Jan 1990 00:00
00d0   3a 30 30 20 47 4d 54 0d 0a 43 61 63 68 65 2d 43  :00 GMT..Cache-C
00e0   6f 6e 74 72 6f 6c 3a 20 70 72 69 76 61 74 65 2c  ontrol: private,
00f0   20 6e 6f 2d 63 61 63 68 65 2c 20 6e 6f 2d 63 61   no-cache, no-ca
0100   63 68 65 3d 53 65 74 2d 43 6f 6f 6b 69 65 2c 20  che=Set-Cookie,
0110   6e 6f 2d 73 74 6f 72 65 2c 20 70 72 6f 78 79 2d  no-store, proxy-
0120   72 65 76 61 6c 69 64 61 74 65 0d 0a 0d 0a        revalidate....
```

the raw frame

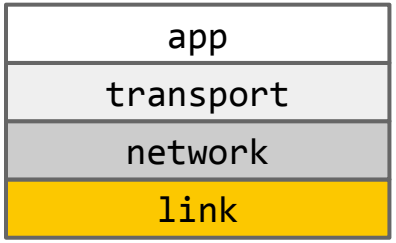
app
transport
network
link

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
```

/epan/dissectors/packet-eth.c:

```
void
proto_register_eth(void)
{
        …
        register_heur_dissector_list("eth", &heur_subdissector_list);
        …
        register_dissector("eth_withoutfcs", dissect_eth_withoutfcs, proto_eth);
}
```

link layer




```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
```

/epan/dissectors/packet-eth.c:
(dissect_eth_withoutfcs()** passes through to **dissect_eth_common())

```
static proto_tree *
dissect_eth_common(tvbuff_t *tvb, packet_info *pinfo, proto_tree *parent_tree,
    int fcs_len)
{
      …
      if (dissector_try_heuristic(heur_subdissector_list, tvb, pinfo,
            parent_tree, &hdtbl_entry, NULL))
      …
      call_dissector_with_data(ethertype_handle, tvb, pinfo, parent_tree,
            &ethertype_data);
      …
}
```

link layer

app
transport
network
link

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
```

/epan/dissectors/packet-ethertype.c:

```
void
proto_register_ethertype(void)
{
        …
        new_register_dissector("ethertype", dissect_ethertype, proto_ethertype);
        …
        ethertype_dissector_table = register_dissector_table("ethertype",
                "Ethertype", FT_UINT16, BASE_HEX);
        …
}
```

link layer

app
transport
network
link

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
```

/epan/dissectors/packet-ethertype.c:

```
static int
dissect_ethertype(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree, void
*data)
{
        …
        next_tvb = tvb_new_subset(tvb, ethertype_data->offset_after_ethertype,
                captured_length, reported_length);
        …
        dissector_found = dissector_try_uint(ethertype_dissector_table,
                ethertype_data->etype, next_tvb, pinfo, tree);
        …
}
```

link layer

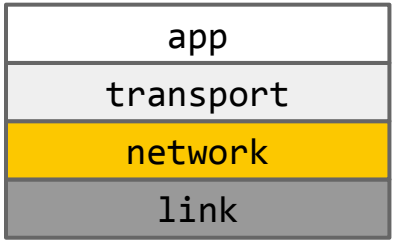
app
transport
network
link

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ..P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
```

/epan/dissectors/packet-ip.c:

```
void
proto_register_ip(void)
{
        …
        ip_dissector_table = register_dissector_table("ip.proto", "IP protocol",
                FT_UINT8, BASE_DEC);
        …
        register_dissector("ip", dissect_ip, proto_ip);
        …
}
```

network layer

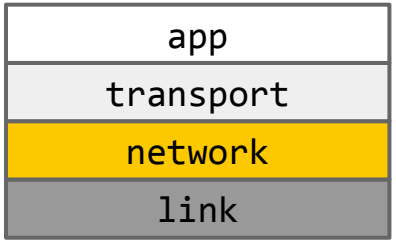



```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
```

/epan/dissectors/packet-ip.c:

```
void
proto_reg_handoff_ip(void)
{
        …
        ip_handle = find_dissector("ip");
        …
        dissector_add_uint("ethertype", ETHERTYPE_IP, ip_handle);
        …
}
```

network layer




```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ..P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
```

/epan/dissectors/packet-ip.c:

```
static void
dissect_ip(tvbuff_t *tvb, packet_info *pinfo, proto_tree *parent_tree)
{
        …
        dissector_try_uint_new(ip_dissector_table, nxt, next_tvb, pinfo,
                parent_tree, TRUE, iph)
        …
}
```

network layer

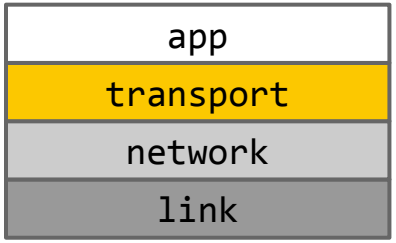
app
transport
network
link

```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
```

/epan/dissectors/packet-tcp.c:

```
void
proto_register_tcp(void)
{
        …
        register_dissector("tcp", dissect_tcp, proto_tcp);
        …
}
```

transport layer

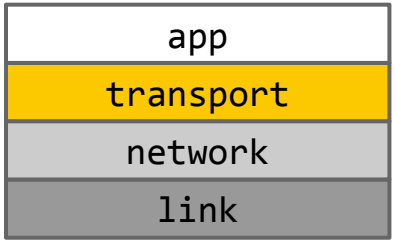



```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
```

/epan/dissectors/packet-tcp.c:

```
void
proto_reg_handoff_tcp(void)
{
        …
        tcp_handle = find_dissector("tcp");
        dissector_add_uint("ip.proto", IP_PROTO_TCP, tcp_handle);
        …
}
```

transport layer




```
0000   00 0c 29 98 30 31 00 50 56 ea 1f 28 08 00 45 00   ..).01.PV..(..E.
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
```

/epan/dissectors/packet-tcp.c:

callchain:
dissect_tcp() calls into:
    dissect_tcp_payload() calls into:
        process_tcp_payload() calls into:
            decode_tcp_ports() calls into:
                dissector_try_heuristic()

dissector_try_heuristic() iterates over the callbacks registered against the "tcp" heuristic dissector list.

HTTP registers one such callback...

transport layer

```
epan/dissectors/packet-rtps.c: heur_dissector_add("tcp", dissect_rtps_tcp, proto_rtps);
epan/dissectors/packet-ziop.c:  heur_dissector_add("tcp", dissect_ziop_heur, proto_ziop);
epan/dissectors/packet-xml.c:           heur_dissector_add("tcp", dissect_xml_heur, xml_ns.hf_tag);
epan/dissectors/packet-giop.c:  heur_dissector_add("tcp", dissect_giop_heur, proto_giop);
epan/dissectors/packet-etch.c:    heur_dissector_add("tcp", dissect_etch, proto_etch);
epan/dissectors/packet-reload-framing.c:  heur_dissector_add("tcp",  dissect_reload_framing_heur,
proto_reload_framing);
epan/dissectors/packet-spice.c:    heur_dissector_add("tcp", test_spice_protocol, proto_spice);
epan/dissectors/packet-nbd.c:    heur_dissector_add("tcp", dissect_nbd_tcp_heur, proto_nbd);
epan/dissectors/packet-dcerpc.c:    heur_dissector_add("tcp", dissect_dcerpc_cn_bs, proto_dcerpc);
epan/dissectors/packet-tds.c:    heur_dissector_add("tcp", dissect_tds_tcp_heur, proto_tds);
epan/dissectors/packet-dplay.c:    heur_dissector_add("tcp", heur_dissect_dplay, proto_dplay);
epan/dissectors/packet-dnp.c:    heur_dissector_add("tcp", dissect_dnp3_tcp, proto_dnp3);
epan/dissectors/packet-icep.c:        heur_dissector_add("tcp", dissect_icep_tcp, proto_icep);
epan/dissectors/packet-paltalk.c:           heur_dissector_add("tcp", dissect_paltalk, proto_paltalk);
epan/dissectors/packet-starteam.c:  heur_dissector_add("tcp", dissect_starteam_heur, proto_starteam);
epan/dissectors/packet-xcsl.c:    heur_dissector_add("tcp", dissect_xcsl_tcp_heur, hfi_xcsl->id);
epan/dissectors/packet-cimd.c:  heur_dissector_add("tcp", dissect_cimd_heur, proto_cimd);
epan/dissectors/packet-rpc.c:    heur_dissector_add("tcp", dissect_rpc_tcp_heur, proto_rpc);
epan/dissectors/packet-openwire.c:    heur_dissector_add("tcp", dissect_openwire_heur, proto_openwire);
epan/dissectors/packet-tali.c:  heur_dissector_add("tcp", dissect_tali_heur, hfi_tali->id);
epan/dissectors/packet-reload.c:  heur_dissector_add("tcp", dissect_reload_heur, proto_reload);
epan/dissectors/packet-q931.c:   heur_dissector_add("tcp", dissect_q931_tpkt_heur, proto_q931);
epan/dissectors/packet-fix.c:    heur_dissector_add("tcp", dissect_fix_heur, proto_fix);
epan/dissectors/packet-drda.c:    heur_dissector_add("tcp", dissect_drda_heur, proto_drda);
epan/dissectors/packet-iwarp-mpa.c:         heur_dissector_add("tcp", dissect_iwarp_mpa, proto_iwarp_mpa);
epan/dissectors/packet-vnc.c:               heur_dissector_add("tcp", test_vnc_protocol, proto_vnc);
epan/dissectors/packet-yhoo.c:   heur_dissector_add("tcp", dissect_yhoo, proto_yhoo);
epan/dissectors/packet-ndmp.c:   heur_dissector_add("tcp", dissect_ndmp_heur, proto_ndmp);
epan/dissectors/packet-fcip.c:    heur_dissector_add("tcp", dissect_fcip_heur, proto_fcip);
epan/dissectors/packet-spdy.c:  heur_dissector_add("tcp", dissect_spdy_heur, proto_spdy);
epan/dissectors/packet-iscsi.c:    heur_dissector_add("tcp", dissect_iscsi_heur, proto_iscsi);
epan/dissectors/packet-ipsec-tcp.c:                   heur_dissector_add("tcp", dissect_tcpencap_heur,
proto_tcpencap);
epan/dissectors/packet-fmtp.c:    heur_dissector_add("tcp", dissect_fmtp, proto_fmtp);
epan/dissectors/packet-lbttcp.c:        heur_dissector_add("tcp", test_lbttcp_packet, proto_lbttcp);
epan/dissectors/packet-mq.c:    heur_dissector_add("tcp",     dissect_mq_heur_tcp, proto_mq);
epan/dissectors/packet-lanforge.c:    heur_dissector_add("tcp", dissect_lanforge, proto_lanforge);
epan/dissectors/packet-bfcp.c:             heur_dissector_add("tcp", dissect_bfcp_heur, proto_bfcp);
epan/dissectors/packet-jxta.c:            heur_dissector_add("tcp", dissect_jxta_TCP_heur, proto_jxta);
epan/dissectors/packet-skype.c:  heur_dissector_add("tcp", dissect_skype_heur, proto_skype);
```

```
epan/dissectors/packet-classicstun.c:    heur_dissector_add("tcp", dissect_classicstun_heur,
proto_classicstun);
epan/dissectors/packet-tuxedo.c: heur_dissector_add("tcp", dissect_tuxedo_heur, proto_tuxedo);
epan/dissectors/packet-adwin-config.c:      heur_dissector_add("tcp", dissect_adwin_config_tcp,
proto_adwin_config);
epan/dissectors/packet-bittorrent.c:   heur_dissector_add("tcp", test_bittorrent_packet,
proto_bittorrent);
epan/dissectors/packet-ctdb.c:   heur_dissector_add("tcp", dissect_ctdb, proto_ctdb);
epan/dissectors/packet-xmcp.c:    heur_dissector_add("tcp", dissect_xmcp_heur, proto_xmcp);
epan/dissectors/packet-ucp.c:    heur_dissector_add("tcp", dissect_ucp_heur, proto_ucp);
epan/dissectors/packet-openflow.c:        heur_dissector_add("tcp", dissect_openflow_heur,
proto_openflow);
epan/dissectors/packet-ifcp.c:    heur_dissector_add("tcp", dissect_ifcp_heur, proto_ifcp);
epan/dissectors/packet-msrp.c:    heur_dissector_add("tcp", dissect_msrp_heur, proto_msrp);
epan/dissectors/packet-lbmpdmtcp.c:        heur_dissector_add("tcp", test_lbmpdm_tcp_packet,
lbmpdm_tcp_protocol_handle);
epan/dissectors/packet-pvfs2.c:  heur_dissector_add("tcp", dissect_pvfs_heur, proto_pvfs);
epan/dissectors/packet-h1.c:  heur_dissector_add("tcp", dissect_h1, proto_h1);
epan/dissectors/packet-http.c:   heur_dissector_add("tcp", dissect_http_heur_tcp, proto_http);
epan/dissectors/packet-smpp.c:    heur_dissector_add("tcp", dissect_smpp_heur, proto_smpp);
epan/dissectors/packet-ymsg.c:   heur_dissector_add("tcp", dissect_ymsg, proto_ymsg);
epan/dissectors/packet-dcm.c:        heur_dissector_add("tcp", dissect_dcm_heuristic, proto_dcm);
epan/dissectors/packet-sip.c:        heur_dissector_add("tcp", dissect_sip_tcp_heur, proto_sip);
doc/README.heuristic:    heur_dissector_add("tcp", dissect_PROTOABBREV_heur, proto_PROTOABBREV);
```

how many of these are you legitimately running on your network?

app
transport
network
link

```
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
0060   0d 0a 44 61 74 65 3a 20 57 65 64 2c 20 30 32 20   ..Date: Wed, 02
0070   4a 75 6c 20 32 30 31 34 20 32 32 3a 33 32 3a 31   Jul 2014 22:32:1
0080   33 20 47 4d 54 0d 0a 43 6f 6e 6e 65 63 74 69 6f   3 GMT..Connectio
0090   6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 50   n: keep-alive..P
00a0   72 61 67 6d 61 3a 20 6e 6f 2d 63 61 63 68 65 0d   ragma: no-cache.
00b0   0a 45 78 70 69 72 65 73 3a 20 4d 6f 6e 2c 20 30   .Expires: Mon, 0
00c0   31 20 4a 61 6e 20 31 39 39 30 20 30 30 3a 30 30   1 Jan 1990 00:00
00d0   3a 30 30 20 47 4d 54 0d 0a 43 61 63 68 65 2d 43   :00 GMT..Cache-C
00e0   6f 6e 74 72 6f 6c 3a 20 70 72 69 76 61 74 65 2c   ontrol: private,
00f0   20 6e 6f 2d 63 61 63 68 65 2c 20 6e 6f 2d 63 61    no-cache, no-ca
0100   63 68 65 3d 53 65 74 2d 43 6f 6f 6b 69 65 2c 20   che=Set-Cookie,
0110   6e 6f 2d 73 74 6f 72 65 2c 20 70 72 6f 78 79 2d   no-store, proxy-
0120   72 65 76 61 6c 69 64 61 74 65 0d 0a 0d 0a         revalidate....
```

application layer

app
transport
network
link

```
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
```

/epan/dissectors/packet-http.c:

```
void
proto_reg_handoff_message_http(void)
{
        …
        heur_dissector_add("tcp", dissect_http_heur_tcp, proto_http);
        …
}
```

application layer

app
transport
network
link

```
0010   01 20 ff fd 00 00 80 06 17 31 de ad be ef de ad   . .......1..my..
0020   be ef 00 50 c0 1b ef 65 26 24 59 8d f2 f6 50 18   ...P...e&$Y...P.
0030   fa f0 e2 17 00 00 48 54 54 50 2f 31 2e 31 20 32   ......HTTP/1.1 2
0040   30 34 20 4e 6f 20 43 6f 6e 74 65 6e 74 0d 0a 43   04 No Content..C
0050   6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 30   ontent-Length: 0
```

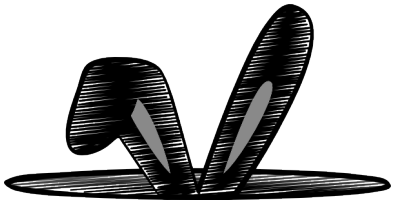
/epan/dissectors/packet-http.c:

```
static gboolean
dissect_http_heur_tcp(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree, void
*data)
{
      …
      if((tvb_strncaseeql(tvb, linelen-8, "HTTP/1.1", 8) == 0)||(tvb_strncaseeql
      (tvb, 0, "HTTP/1.1", 8) == 0)){
                  …
                  dissect_http(tvb, pinfo, tree, data);
                  return TRUE;
            }
      …
}
```

application layer

lame DoS 0days #1–12

/ dozen



lame DoS 0days #1–12

- AKA revenge of the decompression bomb
- `grep -r “tvb_child_uncompress” .`
- at least 12 dissectors affected
  - some of these are heuristic (!)
- limited to 65535 bytes in some cases
  - solutions:
    - attack protocol doing TCP session reassembly
    - send more packets

lame DoS 0day // demo

- `packet-http.c`
- `packet-gadugadu.c`
- you get the idea...

less-lame DoS 0days #13-???

- find loops controlled by user length
  - `tvb_get_ntohl/letohl` is usually a good sign
- make sure offset isn't incremented
  - or can be controlled (`offset += user_data`)
- ensure there aren't any hidden checks
- most Defcon 20 bugs were this type
- slightly less lame than prior bugs

less-lame DoS // MongoDB

- infinite loop in **packet-mongo.c**, <= v1.8.1
- found with:

```
grep -r -B8 -e ‘for.\?(‘ -e ‘while.\?(‘ . | less
```

- **packet-mongo.c** registers interest in TCP port **27017**
- if you can send TCP traffic on **27017** past a listener: profit

less-lame DoS // MongoDB

```
/epan/dissectors/packet-mongo.c:

(multiple functions)
while(offset < tvb_reported_length(tvb)) {
      offset += dissect_bson_document(tvb, pinfo, offset, tree, hf_mongo_document, 1);
}
```

...what if `dissect_bson_document()` return 0?

```
/epan/dissectors/packet-mongo.c:
static int
dissect_bson_document(tvbuff_t *tvb, packet_info *pinfo, guint offset, proto_tree *tree,
int hf_mongo_doc, int nest_level)
{
      ...
      document_length = tvb_get_letohl(tvb, offset);
      ...
      return document_length;
}
```

bug #??? // want more?

- https://www.wireshark.org/download/automated/captures/
  - check back early and often
  - usually at least one current
- https://bugs.wireshark.org/bugzilla/
  - search for “crash” and/or “hang”
  - mirror all the capture files that have been uploaded
- get more pcaps and use Wireshark’s fuzzing framework

super 1337 RCE // code

/epan/dissectors/packet-rtps.c @ rtps_util_add_typecode():

```
guint32 size[MAX_ARRAY_DIMENSION]; /* Max dimensions */
dim_max = NEXT_guint32(tvb, offset, little_endian);
offset += 4;
for (i = 0; i < MAX_ARRAY_DIMENSION; ++i)
    size[i] = 0;
for (i = 0; i < dim_max; ++i) {
    size[i] = NEXT_guint32(tvb, offset, little_endian);
    offset += 4;
}
// :D :D :D :D :D :D :D
```

exploit mitigations // Windows

DEP is on :(

svchost.exe		1,180 K	3,820 K	2240 6.1.7600.16385	DEP (permanent)	System
TrustedInstaller.exe		2,080 K	6,612 K	3700 6.1.7601.17514	DEP (permanent)	System
lsass.exe		2,724 K	6,304 K	540 6.1.7601.18443	DEP (permanent)	System
lsm.exe		1,308 K	2,708 K	548 6.1.7601.17514	DEP (permanent)	System
csrss.exe	0.15	19,664 K	4,648 K	432 6.1.7600.16385	DEP (permanent)	System
winlogon.exe		1,732 K	4,304 K	488 6.1.7601.18409	DEP (permanent)	System
explorer.exe	0.02	36,232 K	44,972 K	2092 6.1.7601.17514	DEP (permanent)	Medium
vmtoolsd.exe	0.05	9,668 K	15,392 K	2356 9.6.2.31837	DEP (permanent)	Medium
SnippingTool.exe	< 0.01	2,940 K	9,480 K	3848 6.1.7600.16385	DEP (permanent)	Medium
GoogleCrashHandler.exe		1,140 K	544 K	2268 1.3.24.15	DEP (permanent)	System
Wireshark.exe	0.79	84,588 K	64,076 K	3732 1.10.8.2	DEP (permanent)	High
procexp.exe	1.55	12,872 K	19,608 K	2884 16.2.0.0	DEP (permanent)	High

D:

Name	Description	Company Name	Path	ASLR
Packet.dll	packet.dll (Vista) Dynamic Link Libr...	Riverbed Technology, Inc.	C:\Windows\System32\Packet.dll	
wpcap.dll	wpcap.dll Dynamic Link Library - b...	Riverbed Technology, Inc.	C:\Windows\System32\wpcap.dll	
Wireshark.exe	Wireshark	The Wireshark developer ...	C:\Program Files\Wireshark\Wireshark.exe	ASLR
SensApi.dll	SENS Connectivity API DLL	Microsoft Corporation	C:\Windows\System32\SensApi.dll	ASLR
wimaxmacphy.dll	wimaxmacphy dissector	The Wireshark developer ...	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR
wimaxasncp.dll	wimaxasncp dissector	The Wireshark developer ...	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR
wimax.dll	m2m dissector	Intel Corporation	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR

exploit mitigations // Windows

but no ASLR on WinPcap modules :) :)

svchost.exe		1,180 K	3,820 K	2240 6.1.7600.16385	DEP (permanent)	System
TrustedInstaller.exe		2,080 K	6,612 K	3700 6.1.7601.17514	DEP (permanent)	System
lsass.exe		2,724 K	6,304 K	540 6.1.7601.18443	DEP (permanent)	System
lsm.exe		1,308 K	2,708 K	548 6.1.7601.17514	DEP (permanent)	System
csrss.exe	0.15	19,664 K	4,648 K	432 6.1.7600.16385	DEP (permanent)	System
winlogon.exe		1,732 K	4,304 K	488 6.1.7601.18409	DEP (permanent)	System
explorer.exe	0.02	36,232 K	44,972 K	2092 6.1.7601.17514	DEP (permanent)	Medium
vmtoolsd.exe	0.05	9,668 K	15,392 K	2356 9.6.2.31837	DEP (permanent)	Medium
SnippingTool.exe	< 0.01	2,940 K	9,480 K	3848 6.1.7600.16385	DEP (permanent)	Medium
GoogleCrashHandler.exe		1,140 K	544 K	2268 1.3.24.15	DEP (permanent)	System
Wireshark.exe	0.79	84,588 K	64,076 K	3732 1.10.8.2	DEP (permanent)	High
procexp.exe	1.55	12,872 K	19,608 K	2884 16.2.0.0	DEP (permanent)	High

Name	Description	Company Name	Path	ASLR
Packet.dll	packet.dll (Vista) Dynamic Link Libr...	Riverbed Technology, Inc.	C:\Windows\System32\Packet.dll	:D
wpcap.dll	wpcap.dll Dynamic Link Library - b...	Riverbed Technology, Inc.	C:\Windows\System32\wpcap.dll	
Wireshark.exe	Wireshark	The Wireshark developer ...	C:\Program Files\Wireshark\Wireshark.exe	ASLR
SensApi.dll	SENS Connectivity API DLL	Microsoft Corporation	C:\Windows\System32\SensApi.dll	ASLR
wimaxmacphy.dll	wimaxmacphy dissector	The Wireshark developer ...	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR
wimaxasncp.dll	wimaxasncp dissector	The Wireshark developer ...	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR
wimax.dll	m2m dissector	Intel Corporation	C:\Program Files\Wireshark\plugins\1.10.8\wi...	ASLR

exploit mitigations // Windows

/GS enabled :(

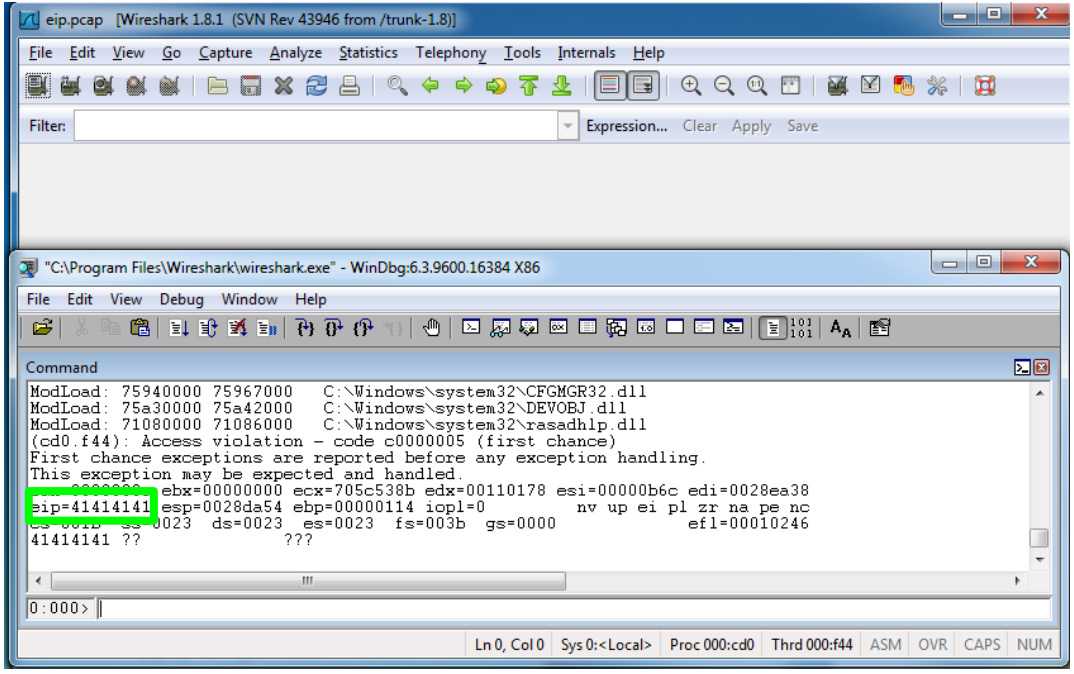
```
6a06ff4d 8b4df0          mov     ecx,dword ptr [ebp-10h]
6a06ff50 33cd            xor     ecx,ebp
6a06ff52 e8a9f79400      call    libwireshark!__security_check_cookie (6a9bf700)
```

...but variable reordering wasn't triggered :) :) :)

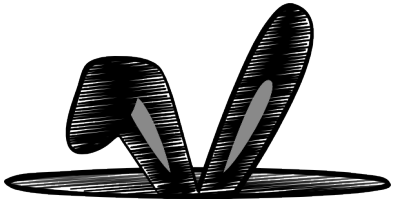
```
6a06fad8 e86356d0ff      call    libwireshark!tvb_get_ntohl (69d75140)
6a06fadd 83c408          add     esp,8
6a06fae0 898594feffff    mov     dword ptr [ebp-16Ch],eax
6a06fae6 8b45c0          mov     eax,dword ptr [ebp-40h] ss:0023:0061cfd0=0000000d
6a06fae9 8b8d94feffff    mov     ecx,dword ptr [ebp-16Ch]
6a06faef 898c850cffffff  mov     dword ptr [ebp+eax*4-0F4h],ecx
6a06faf6 8b5510          mov     edx,dword ptr [ebp+10h]
6a06faf9 83c204          add     edx,4
```

super 1337 RCE // exploitation

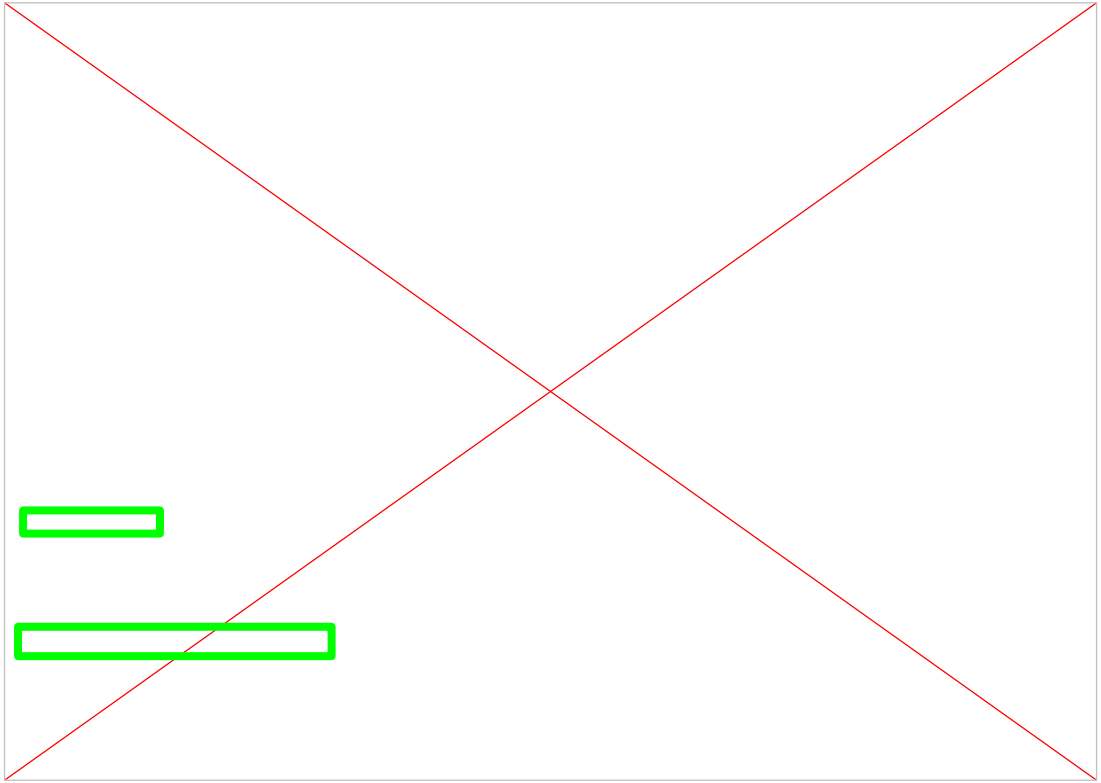
Overwrite `dim_size` and `i` with the right values...



:D



super 1337 RCE // wat



super 1337 RCE // exploitation

- let's recap: we've got
  - ASLR defeat
  - EIP & ESP
  - what else could we possibly need?
- attack different versions
  - vuln goes back to v1.03(!), released in Sept 2008
- stack space...later versions aren't nice.
- solution: we have to go deeper

super 1337 RCE // exploitation

we can recurse before we crash!

```
/epan/dissectors/packet-rtps.c @ rtps_util_add_typecode():
case RTI_CDR_TK_ARRAY: {
    …
    /* Recursive decode seq typecode */
    /*offset += */rtps_util_add_typecode(tree, tvb, offset,
        little_endian, indent_level, is_pointer,
        bitfield, is_key, offset_begin, name, -1, size,
        ndds_40_hack);
    …
}
```

super 1337 RCE // exploitation

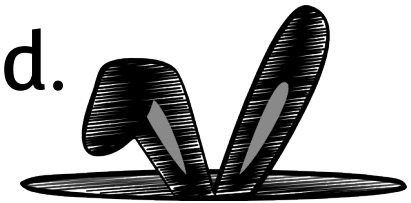
- benefits of recursing are many
  - fix up stack frames
  - perform differently with different stack layouts
  - gain more stack space to work with
  - support multiple versions in single packet
- downside: annoying to write and test
- let's start with frame differences...

super 1337 RCE // exploitation

- v1.10 changed stack layout from v1.8
- `i` (and other vars) at different offsets
- `RA` & `SP` at different offsets
  - v1.8 has slightly more stack space
- strategy:
  - do v1.10 overflow first, then recurse
  - do v1.8 overflow second, and return
  - if no code exec yet, we'll return again

super 1337 RCE // exploitation

- works good in theory
- `tvb` pointer stored after stored `RA`
  - greatly reduced stack space :(
- recurse three times just for v1.10
  - once to copy shellcode to stack
  - once to set up an add `ESP,8` pivot
  - once to copy ROP chain above pivot
- chaining together with v1.8 is hard.

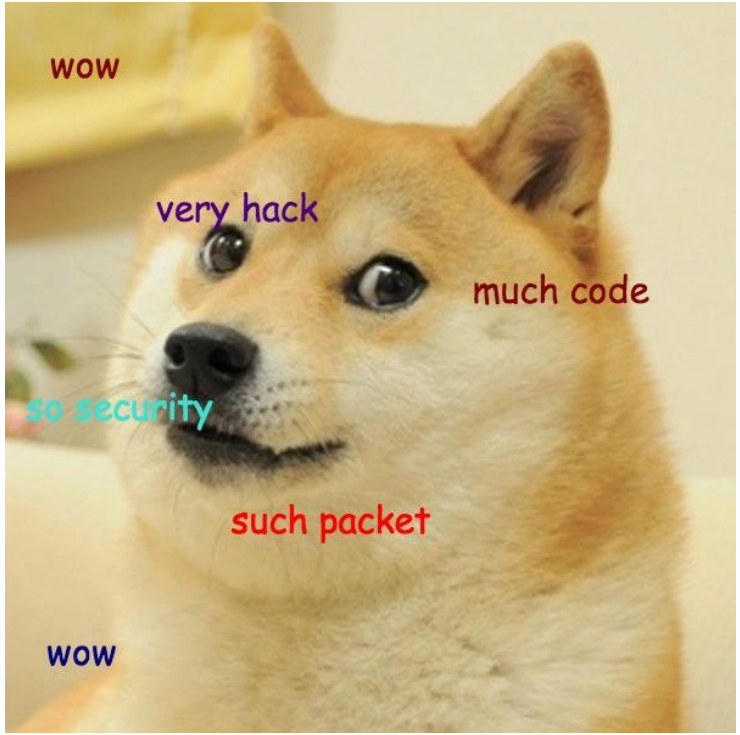


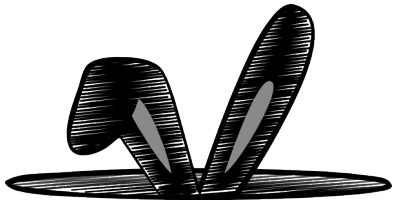
super 1337 RCE // exploitation

v1.10.1	v1.8.9
Shared Preamble	
First vulnerable frame - desync occurs	
Copy in shellcode	Skip frame by long copy
Set up stack pivot	Skip frame by long copy
Copy ROP	Copy in shellcode
Encounters i=0 and returns	Set up stack pivot
N/A	Copy ROP and return

super 1337 RCE // demo

- v1.8
- v1.10





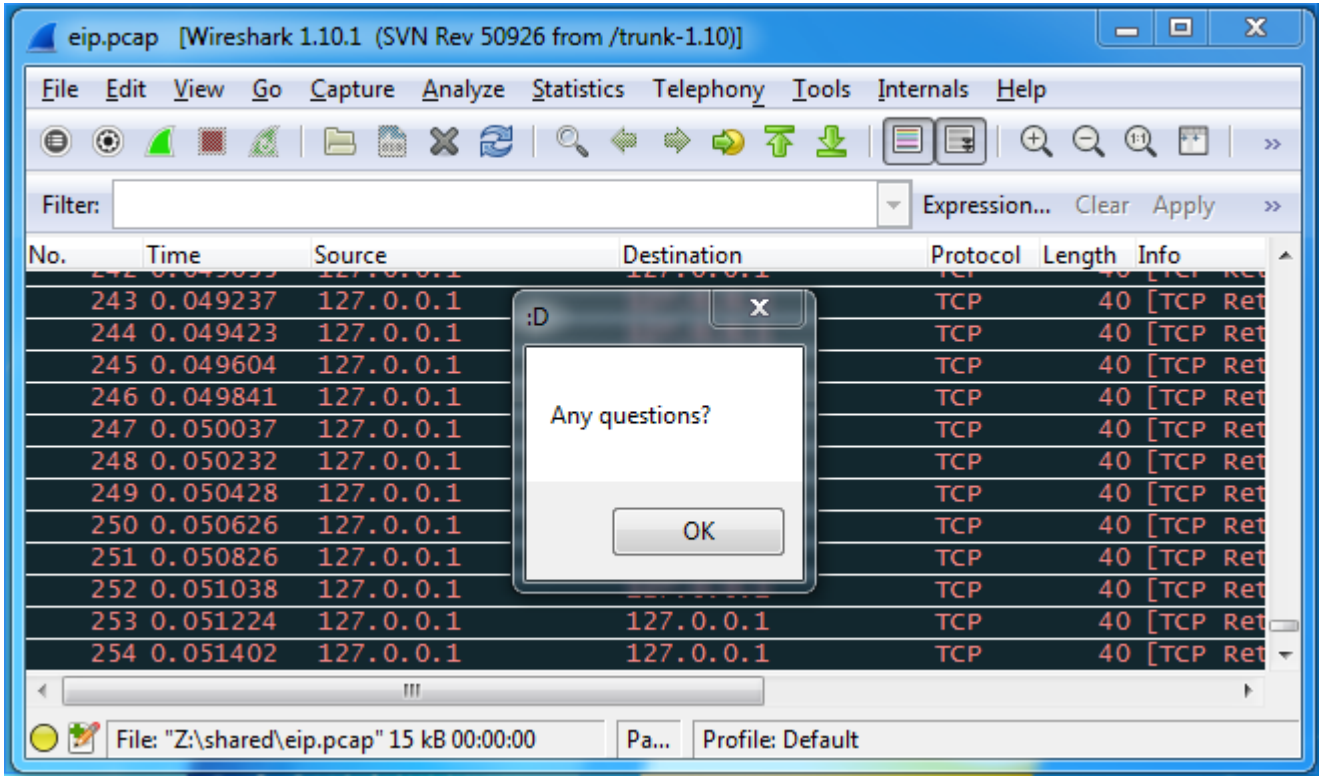
mitigations

- for users
  - disable dissectors you don't need
  - follow recommendations on reducing privilege: http://wiki.wireshark.org/CaptureSetup/CapturePrivileges
- for devs
  - `seccomp()` / `seccomp_bpf()`
  - actual privilege separation (separate processes)
  - better killing of long-running dissectors
  - more hardened memory-management functions
  - more official/automated code review
  - don't handle jumping to `0x41414141`?

thanks

- Samurai
- Wireshark devs
- thesaurus.com

questions


eip.pcap [Wireshark 1.10.1 (SVN Rev 50926 from /trunk-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply
No. Time Source Destination Protocol Length Info
243 0.049237 127.0.0.1 TCP 40 [TCP Ret
244 0.049423 127.0.0.1 TCP 40 [TCP Ret
245 0.049604 127.0.0.1 TCP 40 [TCP Ret
246 0.049841 127.0.0.1 TCP 40 [TCP Ret
247 0.050037 127.0.0.1 TCP 40 [TCP Ret
248 0.050232 127.0.0.1 TCP 40 [TCP Ret
249 0.050428 127.0.0.1 TCP 40 [TCP Ret
250 0.050626 127.0.0.1 TCP 40 [TCP Ret
251 0.050826 127.0.0.1 TCP 40 [TCP Ret
252 0.051038 127.0.0.1 TCP 40 [TCP Ret
253 0.051224 127.0.0.1 127.0.0.1 TCP 40 [TCP Ret
254 0.051402 127.0.0.1 127.0.0.1 TCP 40 [TCP Ret
:D
Any questions?
OK
File: "Z:\shared\eip.pcap" 15 kB 00:00:00
Pa...
Profile: Default