# Network Security 2

The Internet Of Things is the nightmare of pervasive embedded insecurity made real. - Taylor Swift

### Physical/Link-Layer Threats: Eavesdropping

#### Computer Science 161 Spring 2019

- Also termed sniffing
- For subnets using broadcast technologies (e.g., WiFi, some types of Ethernet), get it for "free"
  - Each attached system's NIC (= Network Interface Card) can capture any communication on the subnet
  - Some handy tools for doing so
    - tcpdump (low-level ASCII printout)

# TCPDump

Computer Science 161 Spring 2019	Popa & Weaver
demo 2 % tcpdump -r all.trace2	
reading from file all.trace2, link-type EN10MB (Ethernet)	
21:39:37.772367 IP 10.0.1.9.60627 > 10.0.1.255.canon_bjnp2: UDP, length 16	
21:39:37.772565 IP 10.0.1.9.62137 > all_systems.mcast.net.canon_bjnp2: UDP, length 16	
21:39:39.923030 IP 10.0.1.9.17500 > broadcasthost.17500: UDP, length 130	
21:39:39.923305 IP 10.0.1.9.17500 > 10.0.1.255.17500: UDP, length 130	
21:39:42.286770 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [S], seq 2	
523449627, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 429017455 ecr 0,sack	
OK,eol], length 0	
21:39:42.309138 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [S.], seq	
3585654832, ack 2523449628, win 14480, options [mss 1460,sackOK,TS val 1765826995 ecr 42	
9017455,nop,wscale 9], length 0	
21:39:42.309263 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [.], ack 1	
, win 65535, options [nop,nop,TS val 429017456 ecr 1765826995], length 0	
21:39:42.309796 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [P.], seq	
1:525, ack 1, win 65535, options [nop,nop,TS val 429017456 ecr 1765826995], length 524	
21:39:42.326314 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [.], ack 5	
25, win 31, options [nop,nop,TS val 1765827012 ecr 429017456], length 0	
21:39:42.398814 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [P.], seq	
1:535, ack 525, win 31, options [nop,nop,TS val 1765827083 ecr 429017456], length 534	3
21:39:42.398946 IP 10.0.1.13.61901 > star=01=02=pao1.facebook.com.http: Flaas [.]. ack 5	3

### Physical/Link-Layer Threats: Eavesdropping

#### Computer Science 161 Spring 2019

- Also termed sniffing
- For subnets using broadcast technologies (e.g., WiFi, some types of Ethernet), get it for "free"
  - Each attached system's NIC (= Network Interface Card) can capture any communication on the subnet
  - Some handy tools for doing so
    - tcpdump (low-level ASCII printout)
    - Wireshark (higher-level printing)

#### Wireshark: GUI for Packet Capture/Exam.

		<b>(</b> )		X all.tra	ce2 [Wireshark	x 1.6.2 J	
	<u>F</u> ile <u>E</u>	dit <u>V</u> iew <u>G</u> o <u>C</u> a	apture <u>A</u> nalyze <u>S</u> tatistics T	elephony <u>T</u> ools <u>I</u> nternals <u>H</u>	elp		
	ð, íð		🗅 🛃 🗶 📚 📇	9, 수 🔿 🏵 🛃		ਦ, ਦ, @,    , № ங ‰   💢	
er Science 161 Spri	Filter:			Expression Cle	ar Apply		Popa & Weaver
	No.	Time	Source	Destination	Protocol Len	gth Info	
		1 0.000000	10.0.1.9	10.0.1.255	BJNP	58 Printer Command: Unknown code (2)	
		2 0.000198	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)	
		3 2.150663	10.0.1.9	255.255.255.255		172 Dropbox LAN sync Discovery Protocol	
		4 2.150938	10.0.1.9	10.0.1.255		172 Dropbox LAN sync Discovery Protocol	
		5 4.514403	10.0.1.13	31.13.75.23	TCP	78 61901 > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 TSval=4290	
		6 4.536771	31.13.75.23	10.0.1.13	TCP	74 http > 61901 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK	
		7 4.536896	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=429017456 T	
		8 4.537429	10.0.1.13	31.13.75.23	HTTP	590 GET / HTTP/1.1	
		9 4.553947	31.13.75.23	10.0.1.13	TCP	66 http > 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827012	
		10 4.626447	31.13.75.23	10.0.1.13	нттр	600 HTTP/1.1 302 Found	
		11 4.626579	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=525 Ack=535 Win=524280 Len=0 TSval=4290174	
		12 7.065664	10.0.1.9	10.0.1.255	BJNP	58 Printer Command: Unknown code (2)	
		13 7.065846	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)	
	Þ ⊤ran		)l Protocol, Src Port: ht	23 (31.13.75.23), Dst: 10 tp (80), Dst Port: 61901			
	0010 0020 0030	02 4a 67 be 00 01 0d 00 50 f1 00 1f f4 2f 00	00 00 25 00 fe aa 41 08 00 58 06 83 9f 1f 0d 4b cd d5 b8 c0 31 96 68 cb 00 01 01 08 0a 69 40 62 50 2f 31 2e 31 20 33 30	17 0a 00 .JgX 28 80 18P 1.h 0b 19 92/i@	K .( b	×	5

#### Wireshark: GUI for Packet Capture/Exam.

Veaver

6 4.536771         31.13.75.23         10.0.1.13         TCP         74 http > 61901         [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 Sc           7 4.536896         10.0.1.13         31.13.75.23         TCP         66 61901 > http         [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=429017450           8 4.537429         10.0.1.13         31.13.75.23         HTTP         590 GET / HTTP/1.1           9 4.553947         31.13.75.23         10.0.1.13         TCP         66 http > 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=17658270           10 4.626447         31.13.75.23         10.0.1.13         HTTP         600 HTTP/1.1 302 Found						ll.trace2 [Wiresha	rk 1.6.2 ]
S1 Star         Filter         Expression         Cler         Apply           No.         Time         Source         Destination         Protocol         Length         Info           1 0.0000000         10.0.1.9         10.0.1.255         BJNP         S8 Printer Command: Unknown code (2)           3 2.150663         10.0.1.9         255.255.255         DB-LSP-D         172 Dropbox LAN sync Discovery Protocol           4 4.2150938         10.0.1.13         31.13.75.23         TCP         74 http > 61901 [SNN, ACK] Sequed Ackel Min-65535 Lone O MSS-1460 MSS=1450 MSS=1460 MSS=1450 MSSS=1550 MSS=1550 MSS=150 MSS=150 MSSS=1550 MSS=150 MSS=1550 MSS=15	1	ile <u>E</u> di	t <u>V</u> iew <u>G</u> o <u>C</u> a	apture <u>A</u> nalyze <u>S</u> t	itistics Telephony <u>T</u> ools Internal	s <u>H</u> elp	
Hiter:         Expression User Appy           No.         Time         Source         Destination         Protocol         Length         Info           10.000000         10.0.1.9         10.0.1.255         BJNP         58 Printer Command: Unknown code (2)           32.150663         10.0.1.9         224.0.0.1         BJNP         58 Printer Command: Unknown code (2)           42.150938         10.0.1.9         252.252.255         DB-LSP-D         172 Dropbox LAN sync Discovery Protocol           5.451403         10.0.1.13         31.13.75.23         TCP         78 05010 > http [StN] Seep0 MineS555 Leno MSS=1460 MS=8 TSval-4.4           6.4538771         31.13.75.23         10.0.1.13         31.13.75.23         TCP         66 0100 > http [AtK] Seep1 Ack =1 Wine5255 Leno MSS=1460 MS=8 TSval-42200745           8.4537429         10.0.1.13         31.13.75.23         TCP         66 0100 > http [AtK] Seep1 Ack=35 Wine15672 Leno TSval-42200745           10.4626447         31.13.75.23         10.0.1.13         31.13.75.23         TCP         66 0100 > http [AtK] Seep1 Ack=55 Wine15672 Leno TSval-4200745           11.4626579         10.0.1.13         31.13.75.23         TCP         66 0100 > http [Atk] Wine525 Ack=535 Wine15672 Leno TSval-4200745           12.7055864         10.0.1.23         31.13.75.23         TCP         66 0100 > http [A		) (j	斜 🎱 📦	🖿 🛃 🗶 🕯	] 占   🔍 🔶 🗼 😔 👍		🗨 🔍 🔍 📅   🎬 🖹 🎦 ‰   💢
10.000000       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         32.150683       10.0.1.9       255.255.255       DBLSP-D       172 Dropbox LAN sync Discovery Protocol         42.150938       10.0.1.9       255.255.255       DBLSP-D       172 Dropbox LAN sync Discovery Protocol         43.15003       10.0.1.9       10.0.1.255       DBLSP-D       172 Dropbox LAN sync Discovery Protocol         44.150938       10.0.1.13       31.13.75.23       TCP       78 61901 > http [SYN] Seq.0 Win=65335 Lem=0 MSS=1460 MS=8 TSval=4.4         64.383771       31.13.75.23       10.0.1.13       TCP       74 http > 61901 [SYN, ACK] Seq.0 Ack=1 Win=1480 Lem=0 MSS=1460 S         74.358886       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Lem=0 TSval=4290745         84.553947       31.13.75.23       10.0.1.13       TCP       66 fttp > 61901 [ACK] Seq=1 Ack=1 Win=524280 Lem=0 TSval=4290745         10.4.628447       31.13.75.23       10.0.1.13       TCP       66 fttp > 110/1/1.13       Seq.14200 MS=2         12.7.06584       10.0.1.25       BJNP       Se Printer Command: Unknown code (2)       Seq.275 Ack=535 Win=524280 Lem=0 TSval=4280         13.7.052846       10.0.1.25       BJNP       Se Printer Command: Unknown code (2)       Seq.275 Ack=535 Win=524280 Lem=0 TSval=42	Spr <sub>F</sub>	ilter:			▼ Expression	. Clear Apply	
10.000000       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         20.000198       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         32.150663       10.0.1.9       255.255.255       DE-LSP-D       172 Dropbox LAN sync Discovery Protocol         42.150938       10.0.1.9       10.0.1.255       DE-LSP-D       172 Dropbox LAN sync Discovery Protocol         54.51403       31.13.75.23       10.0.1.13       31.13.75.23       TCP       78 61901 > http [SYN] Seq@ Okcl=1 win=14880 Len=0 MSS=1460 S         7.4.558987       31.13.75.23       10.0.1.13       31.13.75.23       TCP       66 61901 > http [Ack] Seq=1 Ack=1 Win=524280 Len=0 TSval=42901745         8.4.535947       31.13.75.23       10.0.1.13       TCP       66 f1901 > http [Ack] Seq=1 Ack=1 Win=524280 Len=0 TSval=1676827         10.4.25247       31.13.75.23       10.0.1.13       HTTP       500 GET / HTTP/1.1         9.4.555947       31.13.75.23       10.0.1.13       HTTP       66 f1901 > http [Ack] Seq=525 Ack=555 Win=524280 Len=0 TSval=167827         114.6262579       10.0.1.25       BJNP       58 Printer Commad: Unknown code (2)       137.065846         12.7.065644       10.0.1.2       01.0.1.25       BJNP       58 Printer Commad: Unknown code (2)         137.065846		lo.	Time	Source	Destination	Protocol Le	ength   Info
2 0.000198       10.0.1.9       224.0.0.1       BJNP       S8 Printer Command: Utknown code (2)         3 2.150653       10.0.1.9       10.0.1.255       D8-LSP-D       172 Dropbox LAN sync Discovery Protocol         4 2.150938       10.0.1.13       31.13.75.23       TCP       78 61301 > http (Seq) KAN Sync Discovery Protocol         5 4.514403       10.0.1.13       31.13.75.23       TCP       78 61301 > http (Seq) Ack.1 Wan=53450 Len=0 MSS=1460 KS= 1460 S         7 4.536896       10.0.1.13       31.13.75.23       TCP       76 61301 > http (Seq) Ack.1 Wan=524280 Len=0 TSval=42201745         8 4.537429       10.0.1.13       31.13.75.23       TCP       66 61501 > http (AX) Seq=1 Ack=1 Wan=524280 Len=0 TSval=42201745         9 4.553947       31.13.75.23       10.0.1.13       TCP       76 61501 > http (AX) Seq=1 Ack=525 Win=15872 Len=0 TSval=42201745         10 4.625479       10.0.1.13       31.13.75.23       TCP       66 61501 > http (AX) Seq=2 Ack=535 Win=5372 Len=0 TSval=42201745         12 7.065564       10.0.1.13       31.13.75.23       TCP       66 61501 > http (AX) Seq=2 Ack=55 Win=5372 Len=0 TSval=4280         12 7.065564       10.0.1.13       31.13.75.23       TCP       66 61501 > http (AX) Seq=252 Ack=535 Win=5372         12 7.065564       10.0.1.13       31.13.75.23       TCP       66 615001 > http (AX) Seq=24.455       Wi				10.0.1.9	10.0.1.255		-
4 2.150938       10.0.1.9       10.0.1.25       DB-LSP-D       172 Drophox LAN sync Discovery Protocol         5 4.514403       10.0.1.13       31.13.75.23       TCP       78 61901 > http [STN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 Tsval=44.         6 4.536771       31.13.75.23       10.0.1.13       TCP       78 61901 > http [ACK] Seq=0 Ack=1 Win=524280 Len=0 TSval=42901745         7 4.536896       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=42901745         9 4.553947       31.13.75.23       10.0.1.13       TCP       66 61901 > http [ACK] Seq=1 Ack=52 Win=15872 Len=0 TSval=42901745         10 4.626547       31.13.75.23       10.0.1.13       TCP       66 61901 > http [ACK] Seq=1 Ack=52 Win=15872 Len=0 TSval=4290         12 7.065644       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         15 Former LI, Src: Apple fe:a410 (00:25:00:fe:aa:41), Dst: Apple 0.41:eb:00 (etc:e:8f:41:eb:00)       >       >         9 Exberner LI, Src: Apple fe:a400 (bits), SoO bytes captured (4800 bits), SoO bytes on wire (4800 bits), SoO bytes captured (190, Dst Port: 61901 (61901)       Source port: http (80)       >         9 Exberner LI, Src: Apple fe:a41 (b0:25:00:fe:aa:41], Dst: Apple 41:eb:00 (etc:e:8f:41:eb:00)       >			2 0.000198	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)
5 4.51403       10.0.1.13       31.13.75.23       TCP       78 61901 > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 TSVal=4         6 4.536771       31.13.75.23       10.0.1.13       TCP       74 http > 61901 [SYN, ACK] Seq=0 Ack=1 Win=24480 Len=0 MSS=1460 S         7 4.536896       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=1 Ack=1 Win=54280 Len=0 TSVal=42901745         8 4.537429       10.0.1.13       31.13.75.23       HTP       590 GET / HTTP/1.1         9 4.553647       31.13.75.23       10.0.1.13       TCP       66 http > 61901 [ACK] Seq=1 Ack=525 Win=5872 Len=0 TSval=4290174         10 4.626447       31.13.75.23       10.0.1.13       TCP       66 http > http [ACK] Seq=1 Ack=525 Win=524280 Len=0 TSval=4290         11 4.626579       10.0.1.13       31.13.75.23       TCP       66 fst901 > http [ACK] Seq=1 Ack=525 Win=524280 Len=0 TSval=4290         12 7.065846       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         Y Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901)       Stream index: 0]       Sequence number: 1       (relative sequence number)         Reader length: 32 bytes       P Flags: 018 (PSH, ACK)       Window size scaling factor: S12]       (Vindow			3 2.150663	10.0.1.9	255.255.255.255	DB-LSP-D	172 Dropbox LAN sync Discovery Protocol
6 4.536771       31.13.75.23       10.0.1.13       TCP       74 http > 61901 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 S         7 4.536896       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=429017450         9 4.553947       31.13.75.23       10.0.1.13       TCP       66 http > 61901 > http [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827         10 4.626447       31.13.75.23       10.0.1.13       TCP       66 http > 61901 > http [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827         11 4.626547       10.0.1.13       31.13.75.23       10.0.1.13       HTP       590 GET / HTTP/1.1       302 Formal State Stat			4 2.150938	10.0.1.9	10.0.1.255	DB-LSP-D	172 Dropbox LAN sync Discovery Protocol
<pre>7 4.536966 10.0.1.13 31.13.75.23 TCP 66 61901 &gt; http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=42901745 8 4.537429 10.0.1.13 31.13.75.23 10.0.1.13 TCP 66 http &gt; 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827 10 4.626447 31.13.75.23 10.0.1.13 TCP 66 http &gt; 61901 [ACK] Seq=1 Ack=525 Win=524280 Len=0 TSval=1765827 10 4.626447 31.13.75.23 10.0.1.13 HTTP 600 HTTP/1.1 302 Found 11 4.626579 10.0.1.13 31.13.75.23 TCP 66 61900 &gt; http [ACK] Seq=252 Ack=535 Win=524280 Len=0 TSval=4290 12 7.065664 10.0.1.9 10.0.1.255 BJNP 58 Printer Command: Unknown code (2) 13 7.065846 10.0.1.9 224.0.0.1 BJNP 58 Printer Command: Unknown code (2) 13 7.065846 10.0.1.9 224.0.0.1 BJNP 58 Printer Command: Unknown code (2) 7 Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits) 7 Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits) 7 Transmission Control Protocol, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13) 8 Ethernet II, Src: Apple_fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00) 9 Internet Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13) 7 Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901) 7 Sequence number: 1 (relative sequence number) 7 Next Sequence number: 525 (relative sequence number) 7 Header length: 32 bytes 9 Flags: 0x18 (PSH, ACK) 7 Window size value: 31 7 [Calculated window size: 15872] 7 [Window size value: 31 7 [Calculated window size: 15872] 7 [Window size scaling factor: 512] 9 Checksum: 0xf42f [validation disabled] 7 Transmission 0xf42f [validation disabled</pre>			5 4.514403	10.0.1.13	31.13.75.23	TCP	78 61901 > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 TSval=429
7 4,538966       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=42901745;         8 4.537429       10.0.1.13       31.13.75.23       10.0.1.13       TCP       66 http > 61901 [ACK] Seq=1 Ack=525 Win=5872 Len=0 TSval=42901745;         10 4.628477       31.13.75.23       10.0.1.13       TCP       66 http > 61901 [ACK] Seq=1 Ack=525 Win=5872 Len=0 TSval=4290         11 4.628579       10.0.1.13       31.13.75.23       TCP       66 61901 > http [ACK] Seq=252 Ack=535 Win=524280 Len=0 TSval=4290         12 7.065664       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         ************************************			6 4.536771	31.13.75.23	10.0.1.13	TCP	74 http > 61901 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SA
9 4.553947       31.13.75.23       10.0.1.13       TCP       66 http > 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827/ 10 4.628447         11 4.628579       10.0.1.13       31.13.75.23       TCP       66 http > 61901 > HTTP 1.1 302 Found         12 7.065664       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         ************************************			7 4.536896	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=429017456
<pre>10 4.628447 31.13.75.23 10.0.1.13 HTTP 600 HTTP/1.1 302 Found 11 4.628579 10.0.1.13 31.13.75.23 TCP 66 61901 &gt; http [ACK] Seq=525 Ack=535 Win=524280 Len=0 TSval=4290 12 7.065664 10.0.1.9 10.0.1.255 BJNP 58 Printer Command: Unknown code (2) 13 7.065846 10.0.1.9 224.0.0.1 BJNP 58 Printer Command: Unknown code (2) P Ethernet II, Src: Apple fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00) &gt; Internet Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13) Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901), Seq: 1, Ack: 525, Len: 534 Source port: http (80) Destination port: 61901 (61901) [Stream index: 0] Sequence number: 1 (relative sequence number) [Next sequence number: 525 (relative sequence number)] Acknowledgement number: 525 (relative ack number) Header length: 32 bytes &gt; Flags: 0x18 (PSH, ACK) Window size value: 31 [Calculated window size: 15872] [Window size value: 31 [Calculated window size i 15872] [Window size value: 31 [Calculated window size scaling factor: 512] &gt; Checksum: 0xf42f [validation disabled] </pre>			8 4.537429	10.0.1.13	31.13.75.23	HTTP	590 GET / HTTP/1.1
11       4.626579       10.0.1.13       31.13.75.23       TCP       66       66       61901 > http       [ACK] Seq=525       Ack=535       Win=524280       Len=0       TSval=4290         12       7.065664       10.0.1.9       10.0.1.255       BJNP       58       Printer Command: Unknown code (2)         13       7.065846       10.0.1.9       224.0.0.1       BJNP       58       Printer Command: Unknown code (2)         *       Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)       58       Printer Command: Unknown code (2)         *       Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)       58       Printer Command: Unknown code (2)         *       Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)       *       *       *         *       Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)       *       *       *         *       Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901)       *       *       *       *         *       Sequence number: 1       (relative sequence number)       *       *       *       *         *       Sequence number: 32       (relative ack number)       *       *       *       *         *       Fra		1	9 4.553947	31.13.75.23	10.0.1.13	TCP	66 http > 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=17658270
12 7.065664       10.0.1.9       10.0.1.255       BJNP       58 Printer Command: Unknown code (2)         13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)         •       Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)       •       •         •       Ethernet II, Src: Apple_fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00)       •         •       Internet Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13)       •         •       Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901), Seq: 1, Ack: 525, Len: 534       Source port: http (80)         Destination port: 61901 (61901)       [Stream index: 0]       •       Sequence number: 1 (relative sequence number)         [Next sequence number: 535 (relative sequence number)]       Acknowledgement number: 535 (relative ack number)       •         Header length: 32 bytes       •       Flags: 0x18 (PSH, ACK)       •         Window size value: 31       [Calculated window size: 15872]       •       •         [Window size scaling factor: 512]       •       •       •         •       Checksum: 0xf2f [validation disabled]       •       •         •       •       •       •       •       •         0010       02 46 75 Bo 00		1	9 4.626447	31.13.75.23	10.0.1.13	HTTP	600 HTTP/1.1 302 Found
13 7.065846       10.0.1.9       224.0.0.1       BJNP       58 Printer Command: Unknown code (2)            Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits)           Fethernet II, Src: Apple_fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00)             Fitner Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13)             Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901), Seq: 1, Ack: 525, Len: 534             Source port: http (80)              Destination port: 61901 (61901)              [Stream index: 0]              Sequence number: 1 (relative sequence number)              [Next sequence number: 525 (relative ack number)              Header length: 32 bytes             Flags: Ox18 (PSH, ACK)             Window size value: 31             [Calculated window size: 15872]             [Window size scaling factor: 512]             Cate Sf 41 eb 00 00 025 00 fe aa 41 08 00 45 20             Outo 02 44 67 86 41 eb 00 00 00 25 00 fe aa 41 08 00 45 20             Outo 02 44 67 96 00 00 58 06 83 96 11 fod 4b 17 0a 00		1	1 4.626579	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=525 Ack=535 Win=524280 Len=0 TSval=42901
<pre>&gt; Frame 10: 600 bytes on wire (4800 bits), 600 bytes captured (4800 bits) &gt; Ethernet II, Src: Apple_fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00) &gt; Internet Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13) &gt; Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901), Seq: 1, Ack: 525, Len: 534 Source port: http (80) Destination port: 61901 (61901) [Stream index: 0] Sequence number: 1 (relative sequence number) [Next sequence number: 535 (relative sequence number)] Acknowledgement number: 525 (relative ack number) Header length: 32 bytes &gt; Flags: 0x18 (PSH, ACK) Window size value: 31 [Calculated window size: 15872] [Window size scaling factor: 512] &gt; Checksum: 0xf42f [validation disabled] A%AE 0000 e4 ce 8f 41 eb 00 00 25 00 fe aa 41 08 00 45 20 0010 02 44 ce 7b e0 00 00 25 00 fe aa 41 08 00 45 20 A%AE</pre>		1	27.065664	10.0.1.9	10.0.1.255	BJNP	58 Printer Command: Unknown code (2)
<pre>     Ethernet II, Src: Apple_fe:aa:41 (00:25:00:fe:aa:41), Dst: Apple_41:eb:00 (e4:ce:8f:41:eb:00)     Internet Protocol Version 4, Src: 31.13.75.23 (31.13.75.23), Dst: 10.0.1.13 (10.0.1.13)     Transmission Control Protocol, Src Port: http (80), Dst Port: 61901 (61901), Seq: 1, Ack: 525, Len: 534     Source port: http (80)     Destination port: 61901 (61901)     [Stream index: 0]     Sequence number: 1 (relative sequence number)     [Next sequence number: 535 (relative sequence number)]     Acknowledgement number: 525 (relative ack number)     Header length: 32 bytes     Flags: 0x18 (PSH, ACK)     Window size value: 31     [Calculated window size: 15872]     [Window size scaling factor: 512]     b Checksum: 0xf42f [validation disabled]     [Outon 02 4a 67 be 00 00 25 00 fe aa 41 08 00 45 20     [A%AE     [A%AE     [A%AE </pre>		13	37.065846	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)
0000 e4 ce 8f 41 eb 00 00 25 00 fe aa 41 08 00 45 20A%AE 0010 02 4a 67 be 00 00 58 06 83 9f 1f 0d 4b 17 0a 00JgXK		Transr Sour [Str Sequ [Ne: Ackr Head ∳ Flag Wind [Ca [Wind	nission Contro ree port: http ination port: ream index: 0 uence number: tt sequence nu nowledgement n der length: 32 gs: 0x18 (PSH, dow size value culated windo ndow size scal	l Protocol, Src ( (80) 61901 (61901) 1 (relative s mber: 535 (re bumber: 525 (r 2 bytes ACK) :: 31 w size: 15872] .ing factor: 512]	Port: http (80), Dst Port: 61 equence number) .ative sequence number)] elative ack number)		
0030 00 1f f4 2f 00 00 01 01 08 0a 69 40 62 0b 19 92,/i@b 0040 49 70 48 54 54 50 2f 31 2e 31 20 33 30 32 20 46 IpHTTP/1 .1 302 F ● Frame (frame), 600 bytes Packets: 13 Displayed: 13 Marked: 0 Load time: 0:00.109 Profile: Default	0 0 0 0	000 e4 010 02 020 01 030 00 040 49	ce 8f 41 eb 4a 67 be 00 0d 00 50 f1 1f f4 2f 00 70 48 54 54	00 00 25 00 fe a 00 58 06 83 9f 1 cd d5 b8 c0 31 9 00 01 01 08 0a 6 50 2f 31 2e 31 2	a 41 08 00 45 20A% f 0d 4b 17 0a 00 .JgX. 6 68 cb 28 80 18P 9 40 62 0b 19 92/ 0 33 30 32 20 46 IpHTTP/1	K .l.h.( i@b .l 302 ⊨	

#### Wireshark: GUI for Packet Capture/Exam.

	0	<u> </u>			ce2 [Wireshark	1.6.2 ]	
		<u>E</u> dit <u>∨</u> iew <u>G</u> o <u>C</u> a		Telephony <u>T</u> ools <u>I</u> nternals <u>H</u>			
			🖻 🛃 🗶 📚 📇	🔍 🔶 📦 🌳 ዥ 🛂		오, 오, 맨, 쨈   🎬 🗹 🍢 ‰   💢	
er Science 161 Spr	Filter:			🔻 Expression Cle	ear Apply		Popa & Wo
	No.	Time	Source	Destination	Protocol Leng	ath Info	
		1 0.000000	10.0.1.9	10.0.1.255	BJNP	58 Printer Command: Unknown code (2)	
		2 0.000198	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)	
		3 2.150663	10.0.1.9	255.255.255.255	DB-LSP-D	172 Dropbox LAN sync Discovery Protocol	
		4 2.150938	10.0.1.9	10.0.1.255	DB-LSP-D	172 Dropbox LAN sync Discovery Protocol	
		5 4.514403	10.0.1.13	31.13.75.23	TCP	78 61901 > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8 TSval=4290	
		6 4.536771	31.13.75.23	10.0.1.13	TCP	74 http > 61901 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK	
		7 4.536896	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=1 Ack=1 Win=524280 Len=0 TSval=429017456 T	
		8 4.537429	10.0.1.13	31.13.75.23	HTTP	590 GET / HTTP/1.1	
		9 4.553947	31.13.75.23	10.0.1.13	TCP	66 http > 61901 [ACK] Seq=1 Ack=525 Win=15872 Len=0 TSval=1765827012	
		10 4.626447	31.13.75.23	10.0.1.13	HTTP	600 HTTP/1.1 302 Found	
		11 4.626579	10.0.1.13	31.13.75.23	TCP	66 61901 > http [ACK] Seq=525 Ack=535 Win=524280 Len=0 TSval=4290174	
		12 7.065664	10.0.1.9	10.0.1.255	BJNP	58 Printer Command: Unknown code (2)	
		13 7.065846	10.0.1.9	224.0.0.1	BJNP	58 Printer Command: Unknown code (2)	
	<ul> <li>▶ Trar</li> <li>▶ Hype</li> <li>▶ Hi</li> <li>↓ Co</li> <li>Se</li> <li>Se</li> <li>Co</li> <li>Co</li> <li>Co</li> <li>Co</li> <li>Co</li> </ul>	nsmission Contro ertext Transfer (TP/1.1 302 Four ocation: https:/ BP: CP="Facebook at-Cookie: high at-Cookie: wd=de ontent-Type: tes (FB-Debug: Os+s)	<pre>pl Protocol, Src Port: ht Protocol dd/r\n //www.facebook.com/\r\n t does not have a P3P pol Contrast=deleted; expires eleted; expires=Thu, 01-2 tt/html; charset=utf-8\r\. ArTHbmLqsy+ArGAuQyqZYR42 0 2013 05:39:42 GMT\r\n alive\r\n</pre>	lan-1970 00:00:01 GMT; pai n	(61901), Seq 0://fb.me/p3p 01 GMT; path=	: 1, Ack: 525, Len: 534	
		•	00 00 25  00 fe aa 41 08	00 45 20A%A	E		

### Physical/Link-Layer Threats: Eavesdropping

#### Computer Science 161 Spring 2019

- Also termed sniffing
- For subnets using broadcast technologies (e.g., WiFi, some types of Ethernet), get it for "free"
  - Each attached system's NIC (= Network Interface Card) can capture any communication on the subnet
  - Some handy tools for doing so
    - tcpdump (low-level ASCII printout)
    - Wireshark (higher-level printing)
    - bro (scriptable real-time network analysis; see bro.org)
- You can also "tap" (mirror) a link or configure a "mirror port"

### One Of Nick's Favorite Toys: DualComm DCGS-2005

#### Computer Science 161 Spring 2019

- A \$200, 5-port Ethernet switch...
  - With some bonus features
- Built in port "mirror"
  - All traffic to and from port 1 is copied to port 5
- Powered through a USB connection
  - So no need for an extra power supply
- Power-Over-Ethernet passthrough
  - Port 2 can send power to port 1 so you can tap IP phones...





#### **Operation Ivy Bells**

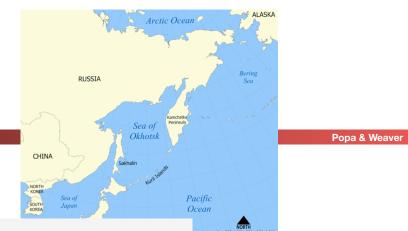
#### By Matthew Carle Military.com

Computer Science 161 Spri

At the beginning of the 1970's, divers from the speciallyequipped submarine, USS Halibut (SSN 587), left their decompression chamber to start a bold and dangerous mission, code named "Ivy Bells".



The Regulus guided missile submarine, USS Halibut (SSN 587) which carried out Operation Ivy Bells.

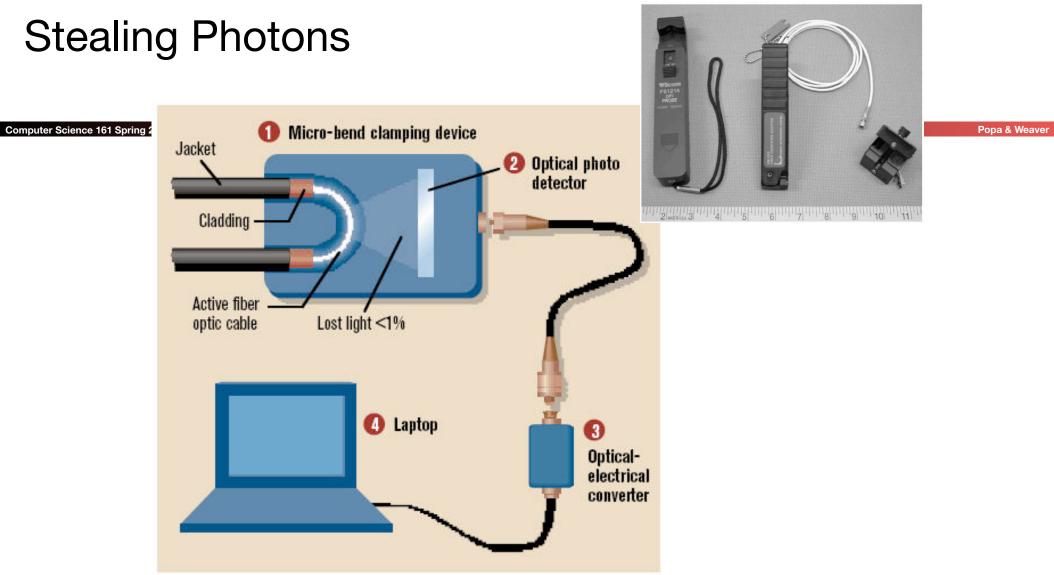


In an effort to alter the balance of Cold War, these men scoured the <u>ocean floor for a five-inch diameter cable</u> carry secret Soviet communications between military bases.

The divers found the cable and installed a <u>20-foot long listening</u> <u>device</u> on the cable. designed to attach to the cable without piercing the casing, the device <u>recorded all communications</u> that occurred. If the cable malfunctioned and the Soviets raised it for repair, the bug, by design, would fall to the bottom of the ocean. <u>Each month Navy divers retrieved the recordings</u> and installed a new set of tapes.

Upon their return to the United States, intelligence agents from the NSA analyzed the recordings and tried to decipher any encrypted information. The Soviets apparently were <u>confident</u> in the <u>security of their communications lines</u>, as a surprising amount of sensitive information traveled through the lines without encryption.

prison. The original tap that was discovered by the Soviets is now on exhibit at the KGB museum in Moscow.



#### П

### The Rogue AP...

#### Computer Science 161 Spring 2019

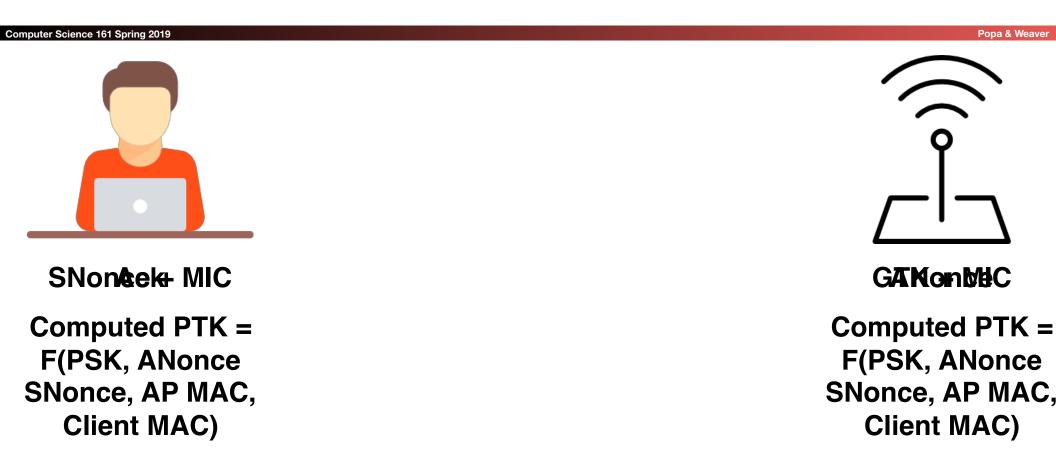
- Your phone/computer keeps broadcasting "Is network X available"?
  - If there is no password, why not just say "Yeah, I'm here!!!"
- Your phone happily connects...
  - To the attacker's internet connection
- The attacker as a man-in-the-middle...
- Can now extract pretty much all non-encrypted communication data...
- "Hey web-browser, spit up effectively all cookies that are sent on non-TLS connections..."

# Wireless Ethernet Security Option: WPA2 Pre Shared Key

Computer Science 161 Spring 2019

- This is what is used these days when the WiFi is "password protected"
  - The access point and the client have the same pre-shared key (called the PSK key)
  - Goal is to create a shared key called the PTK (Pairwise Transient Key)
- This key is derived from a combination of both the password and the SSID (network name)
  - PSK = PBKDF2(SHA1, passphrase, ssid, 4096, 256)
- Use of PBKDF
  - The SSID as salt ensures that the same password on different network names is different
  - The iteration count assures that it is *slow*
    - Any attempt to brute force the passphrase should take a lot of time per guess

### The WPA 4-way Handshake



Icons made by Freepik and Iconic from www.flaticon.com CC 3.0 BY14

### Remarks

- This is **only** secure if an eavesdropper doesn't know the pre shared key
  - Otherwise an eavesdropper who sees the handshake can perform the same computations to get the transport key
  - However, by default, network cards don't do this:
     This is a "do not disturb sign" security. It will keep the maid from entering your hotel room but won't stop a burglar
- Oh, and given ANonce, SNonce, MIC(SNonce), can attempt an offline brute-force attack
  - And since people don't chose good passwords, it will almost certainly succeed: People have built single systems that can try ~8M passwords/second!
  - And can execute a "*deauthentication attack*" to cause the client to disconnect and then reconnect: Running another handshake
- The MIC is really a MAC, but as MAC also refers to the MAC address, they use MIC in the description
- The GTK is for broadcast
  - So the AP doesn't have to rebroadcast things, but usually does anyway

### Rogue APs and WPA2-PSK...

- You can still do a rogue AP!
  - Just answer with a random ANonce...
  - That gets you back the SNonce and MIC(SNonce)
    - Which uses as a key for the MIC = F(PSK, ANonce, SNonce, AP MAC, Client MAC)
- So just do a brute-force dictionary attack on PSK
  - Since PSK = PBKDF2(SHA1, pw, ssid, 4096, 256)
    - So 8192 SHA-1 invocations... Yawn.
  - Verify the MIC to validate whether the guess was correct
- Because lets face it, people don't chose very good passwords...
  - Anyone want to build a full hardware stack version to do this for next DEFCON?
    - Using a Xilinx PYNQ board? Dual core ARM Linux w a 13k logic cell FPGA

### Actually Making it Secure: WPA Enterprise

#### Computer Science 161 Spring 2019

- When you set up Airbears 2, it asks you to accept a public key certificate
  - This is the public key of the *authentication* server, not the access point
- Now before the 4-way handshake:
  - Your computer first handshakes with the authentication server
    - This is secure using public key cryptography
  - Your computer then authenticates to this server
    - With your username and password
- The server now generates a unique key that it both tells your computer and tells the base station
  - So the 4 way handshake is now secure since its a unique PSK

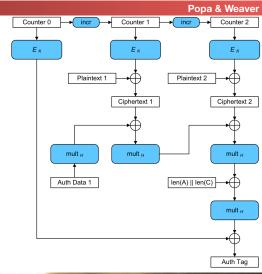
### The Latest Hotness: KRACK attack...

#### Computer Science 161 Spring 2019

- To actually encrypt the individual packets: IV of a packet is {Agreed IV || packet counter}
  - Thus for each packet you only need to send the packet counter (48 bits) rather than the full IV (128b)
- Multiple different modes
  - One common one is CCM (Counter with CBC-MAC)
    - MAC the data with CBC-MAC Then encrypt with CTR mode
  - The highest performance is GCM (Galois/Counter Mode)
- But if you thought CTR mode was bad on IV reuse...
  - GCM is worse: A couple of reused IVs can reveal enough information to forge the authentication!
- Discovered a year ago, fairly quickly patch, but...

### GCM...

- GCM is like CTR mode with a twist...
  - The confidentiality is pure CTR mode
  - The "Galois" part is a hash of the cipher text
    - The only secret part being the "Auth Data"
- Reuse the IV, what happens?
  - Not only do you have CTR mode loss of confidentiality...
  - But if you do it enough, you lose confidentiality on the Auth Data...
  - So you lose the integrity that GCM supposedly provided!





### And Packets Get "Lost"

- Even a wired network will "drop packets"
  - A message is sent but simply never delivered
- Its far worse on wireless
  - A gazillion things can go wrong, including other transmitters
    - And noise like a microwave oven!
- So you have to design for packets to be rebroadcast...
- In the WPA handshake, what do you do when you receive the 3rd packet?
  - Initialize the key you use for encrypting the packets
  - Set the packet counter to 0

### And A Replay Attack...

#### Computer Science 161 Spring 2019

- What if the attacker listens for the third step in the handshake...
  - And then repeats it?
- Why, the client is supposed to reinitialize the key and agreed IV...
  - Which on many implementations, *also resets the packet counter*...
  - Oh, and Linux (and Android 6) is worse... It reinitializes the key to zero!
- So what does that mean?

### Attack Scenario...

- Attacker is close to target
- Attacker captures the 3rd step in the handshake
- Attacker repeatedly replays this to the client
- Client now repeats IVs for encryption...
- Other modes. Annoyance: the damage is minor
- CCM-mode: Attacker can now decrypt in practice thanks to IV reuse
- GCM-mode...
- Attacker can now decrypt *and forge packets*: Reusing the IV also reveals the MAC-secret!

# Mitigations...

- Like all attacks on WiFi, it requires a "close" attacker...
  - 100m to a km or two...
- If you use WPA2-PSK, aka a "WiFi Password", who cares?
  - Unless your WiFi password sounds like a cat hawking up a hairball, you don't have enough entropy to resist a brute-force attacks
- If you use WPA2-Enterprise, this *may* matter...
  - But lets face it, there are so many more critical things to patch first...
  - And why are you treating the WiFi as trusted anyway?

### Announcements!

- Midterm 2 rescheduled: April 9th, 8-10pm
- DSP accommodations
  - If you have one, we assume you still have one
- Other accommodations
- We assume that you no longer need them because of the changed date
- New Accommodations
  - You *must* request by March 6th!

### But Broadcast Protocols Make It Worse...

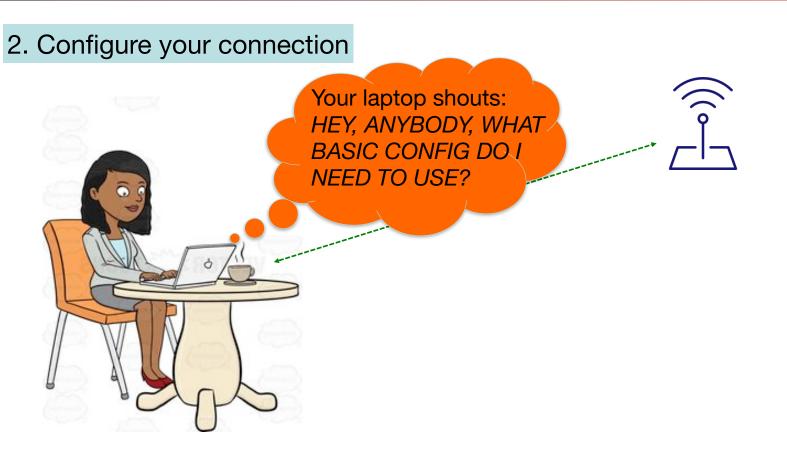
#### Computer Science 161 Spring 2019

- By default, both DHCP and ARP broadcast requests
  - Sent to **all** systems on the local area network
- DHCP: Dynamic Host Control Protocol
  - Used to configure all the important network information
    - Including the DNS server: If the attacker controls the DNS server they have complete ability to intercept all traffic!
    - Including the Gateway which is where on the LAN a computer sends to: If the attacker controls the gateway
- ARP: Address Resolution Protocol
  - "Hey world, what is the Ethernet MAC address of IP X"
  - Used to find both the Gateway's MAC address and other systems on the LAN

### Broadcast Protocols And The LAN

- A rogue device on the LAN can respond to these
  - As long as it arrives first, the attacker wins
- DHCP: Give "bad" gateway...
  - Can directly intercept all traffic to the Internet
- DHCP: Give "proper" gateway but a bad DNS server...
  - Now can intercept all desired traffic by just giving bad DNS response
- ARP: Give "bad" answer for ARP requests to gateway...
  - Can directly intercept all traffic to the Internet

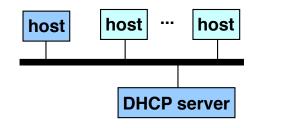




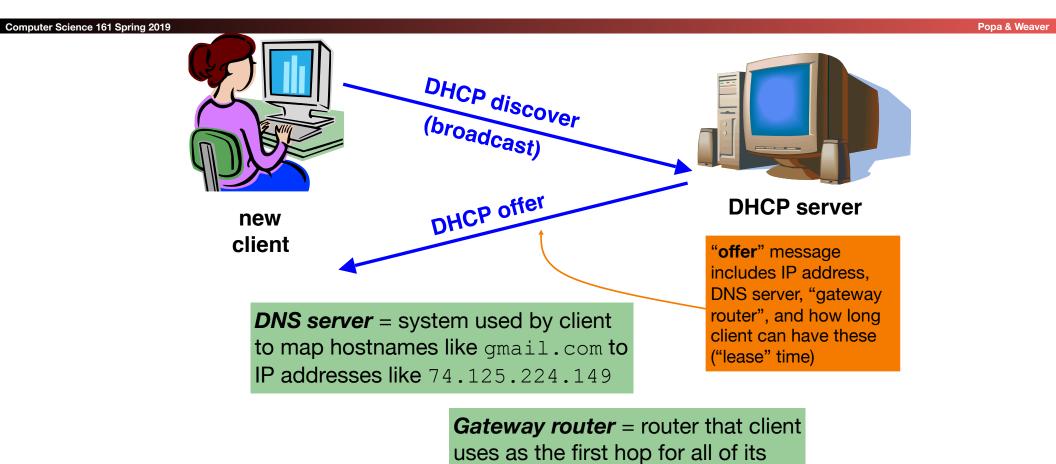
### Internet Bootstrapping: DHCP

#### Computer Science 161 Spring 2019

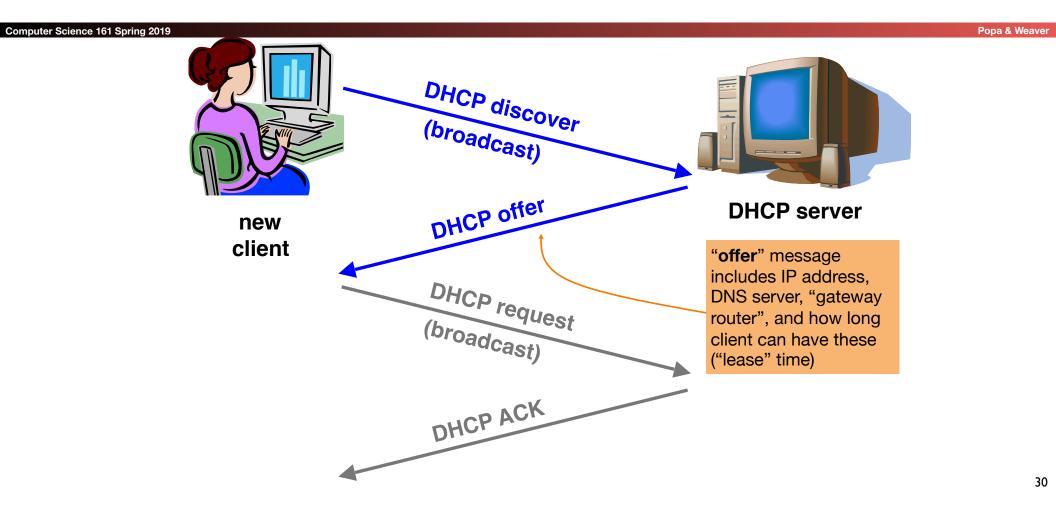
- New host doesn't have an IP address yet
  - So, host doesn't know what source address to use
- Host doesn't know who to ask for an IP address
  - So, host doesn't know what destination address to use
- (Note, host does have a separate WiFi address)
- Solution: shout to "discover" server that can help
  - Broadcast a server-discovery message (layer 2)
  - Server(s) sends a reply offering an address

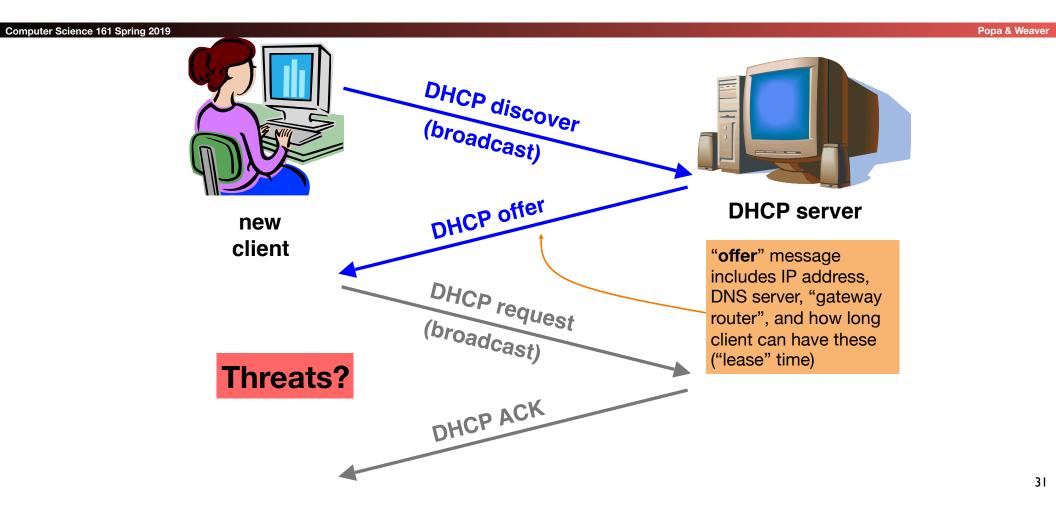


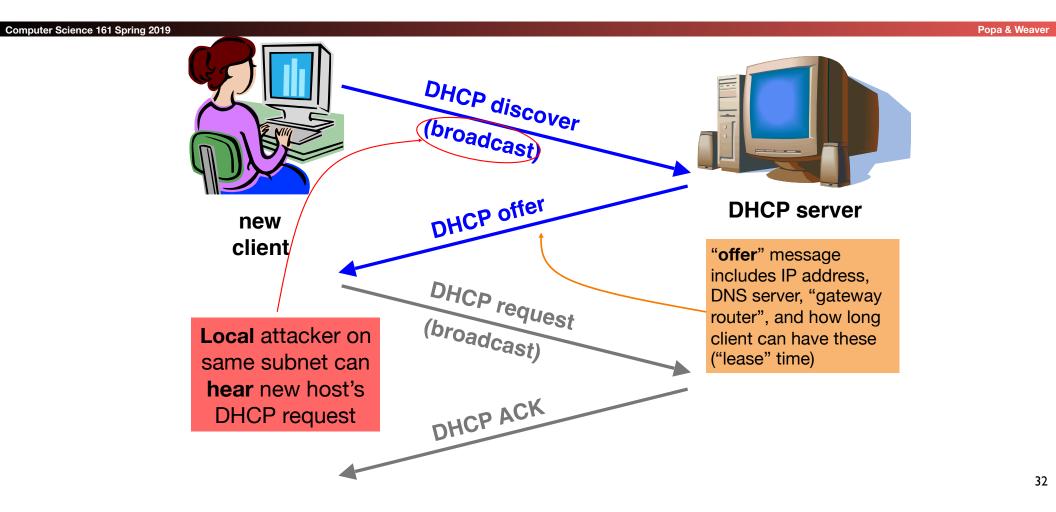
DHCP = Dynamic Host Configuration Protocol

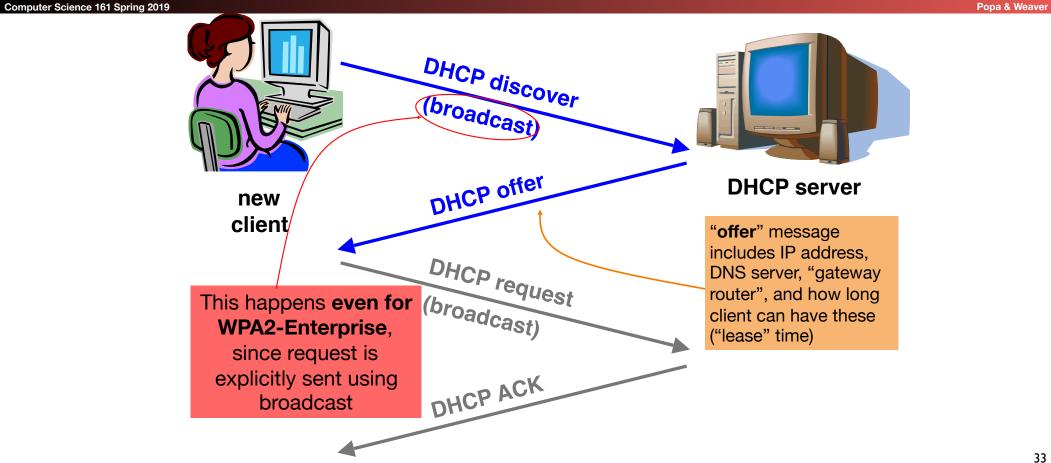


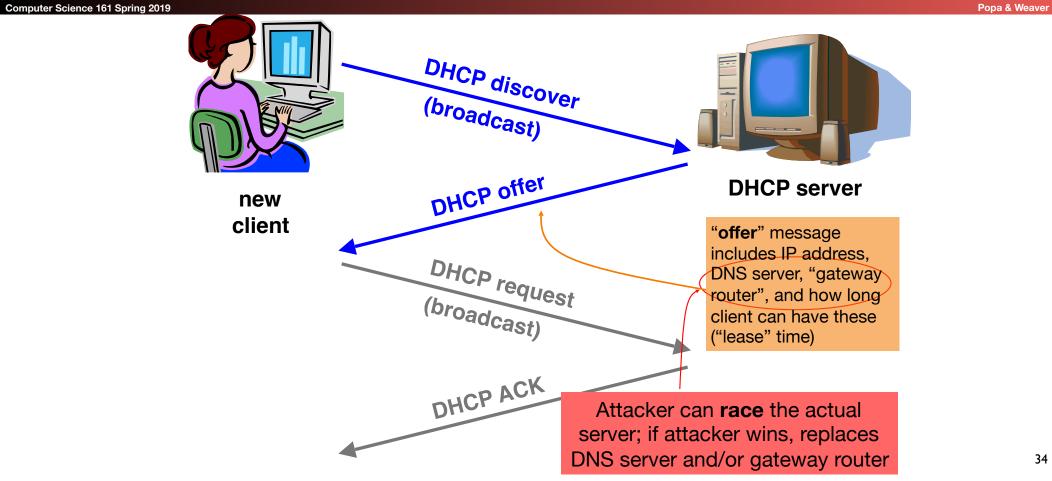
Internet traffic to remote hosts











# **DHCP** Threats

#### Computer Science 161 Spring 2019

- Substitute a fake DNS server
  - Redirect any of a host's lookups to a machine of attacker's choice (e.g., gmail.com = 6.6.6.6)
- Substitute a fake gateway router
  - Intercept all of a host's off-subnet traffic
  - Relay contents back and forth between host and remote server
    - Modify however attacker chooses
  - This is one type of invisible Man In The Middle (MITM)
    - Victim host generally has no way of knowing it's happening!
    - (Can't necessarily alarm on peculiarity of receiving multiple DHCP replies, since that can happen benignly)
- How can we fix this?

*Hard*, because we lack a *trust anchor* 

# **DHCP** Conclusion

#### Computer Science 161 Spring 2019

- DHCP threats highlight:
  - Broadcast protocols inherently at risk of local attacker spoofing
    - Attacker knows exactly when to try it ...
    - ... and can see the victim's messages
  - When initializing, systems are particularly vulnerable because they can lack a trusted foundation to build upon
  - Tension between *wiring in trust* vs. *flexibility and convenience*
  - MITM attacks insidious because no indicators they're occurring

# So How Do We Secure the LAN?

#### Computer Science 161 Spring 2019

- Option 1: We don't
  - Just assume we can keep bad people out
  - This is how most people run their networks: "Hard on the outside with a goey chewy caramel center"

#### • Option 1 variant:

The LAN is a festering pool just like the rest of the Internet, so treat everything as hostile always all the time!

Option 2: smart switching and active monitoring

### The Switch

#### Computer Science 161 Spring 2019

- Hubs are very inefficient:
  - By broadcasting traffic to all recipients this greatly limits the aggregate network bandwidth
- Instead, most Ethernet uses switches
  - The switch keeps track of which MAC address is seen where
- When a packet comes in:
  - If there is no entry in the MAC cache, broadcast it to all ports
  - If there is an entry, send it just to that port
- Result is vastly improved bandwidth
  - All ports can send or receive at the same time

# Smarter Switches: Clean Up the Broadcast Domain

- Modern high-end switches can do even more
- A large amount of potential packet processing on items of interest
- Basic idea: constrain the broadcast domain
  - Either filter requests so they only go to specific ports
  - Limits other systems from listening
  - Or filter replies
    - Limits other systems from replying
- Locking down the LAN is very important practical security
  - This is *real* defense in depth: Don't want 'root on random box, pwn whole network'
  - This removes "pivots" the attacker can try to extend a small foothold into complete network ownership
- This is why an Enterprise switch may cost \$1000s yet provide no more real bandwidth than a \$100 Linksys.

### Smarter Switches: Virtual Local Area Networks (VLANs)

Computer Science 161 Spring 2019

- Our big expensive switch can connect a lot of things together
  - But really, many are in *different* trust domains:
    - Guest wireless
    - Employee wireless
    - Production desktops
    - File Servers
    - etc...
- Want to isolate the different networks from each other
  - Without actually buying separate switches

### **VLANs**

- An ethernet port can exist in one of two modes:
  - Either on a single VLAN
  - On a trunk containing multiple specified VLANs
- All network traffic in a given VLAN stays only within that VLAN
  - The switch makes sure that this occurs
- When moving to/from a trunk the VLAN tag is added or removed
  - But still enforces that a given trunk can only read/write to specific VLANs
- VLAN tag is *automatically* added internally when appropriate to constrain internal traffic

# Putting It Together: If I Was In Charge of UC networking...

- I'd isolate networks into 3+ distinct classes
  - The plague pits (AirBears, Dorms, etc)
  - The mildly infected pits (Research)
  - Administration

Computer Science 161 Spring 2019

- Administration would be locked down
  - Separate VLANs
  - Restricted DHCP/system access
  - Isolated from the rest of campus

# Addressing on the Layers On The Internet

#### Computer Science 161 Spring 2019

Popa & Weaver

- Ethernet:
  - Address is 6B MAC address, Identifies a machine on the local LAN

#### • IP:

• Address is a 4B (IPv4) or 16B (IPv6) address, Identifies a system on the Internet

#### • TCP/UDP:

- Address is a 2B port number, Identifies a particular listening server/process/activity on the system
  - Both the client and server have to have a port associated with the communication
- Ports 0-1024 are for privileged services
  - Must be root to accept incoming connections on these ports
  - Any thing can do an outbound request to such a port
- Port 1025+ are for anybody
  - And high ports are often used ephemerally

### UDP: Datagrams on the Internet

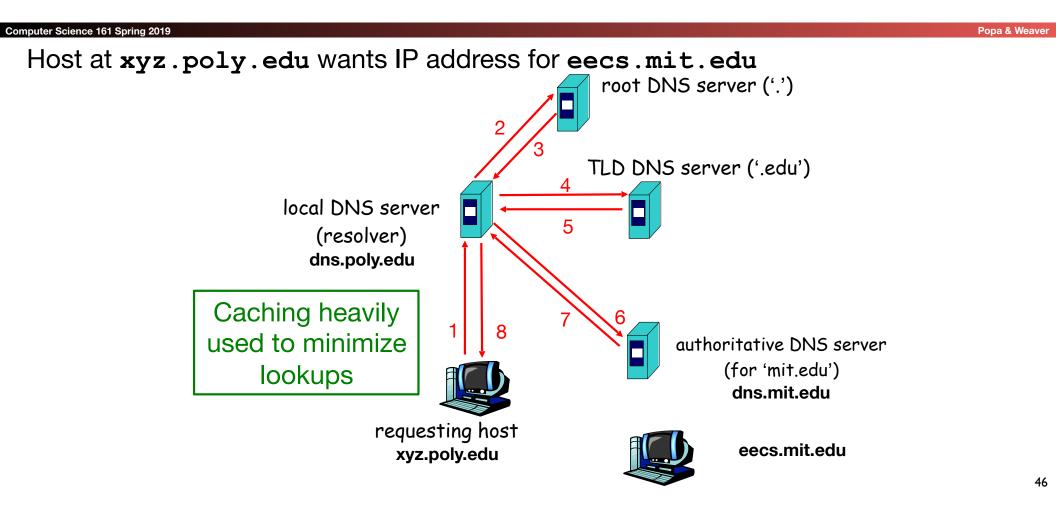
Computer Science 161 Spring 2019

- UDP is a protocol built on the Internet Protocol (IP)
- It is an "unreliable, datagram protocol"
  - Messages may or may not be delivered, in any order
  - Messages can be larger than a single packet
    - IP will fragment these into multiple packets (mostly)
- Programs create a socket to send and receive messages
  - Just create a datagram socket for an ephemeral port
  - Bind the socket to a particular port to receive traffic on a specified port
  - Basic recipe for Python: <u>https://wiki.python.org/moin/UdpCommunication</u>

# **DNS** Overview

- DNS translates www.google.com to 74.125.25.99
  - Turns a human abstraction into an IP address
  - Can also contain other data
- It's a performance-critical distributed database.
- DNS security is critical for the web. (Same-origin policy *assumes* DNS is secure.)
  - Analogy: If you don't know the answer to a question, ask a friend for help (who
    may in turn refer you to a friend of theirs, and so on).
- Based on a notion of hierarchical trust:
  - You trust . for everything, com. for any com, google.com. for everything google...

### DNS Lookups via a Resolver



### Security risk #1: malicious DNS server

```
Popa & Weaver
```

- Of course, if *any* of the DNS servers queried are malicious, they can lie to us and fool us about the answer to our DNS query
- (In fact, they used to be able to fool us about the answer to other queries, too. We'll come back to that.)

### Security risk #2: on-path eavesdropper

#### Computer Science 161 Spring 2019

- If attacker can eavesdrop on our traffic... we're hosed.
- Why? We'll see why.

### Security risk #3: off-path attacker

```
Popa & Weaver
```

- If attacker can't eavesdrop on our traffic, can he inject spoofed DNS responses?
- This case is especially interesting, so we'll look at it in detail.

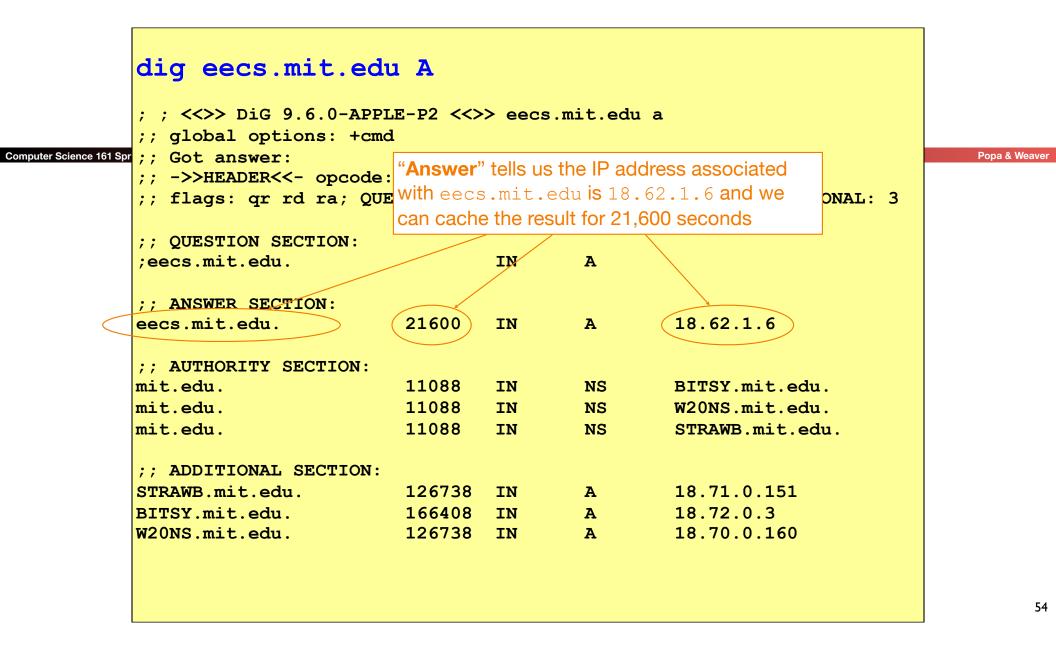
### **DNS** Threats

- DNS: path-critical for just about everything we do
  - Maps hostnames ↔ IP addresses
  - Design only scales if we can minimize lookup traffic
    - #1 way to do so: caching
    - #2 way to do so: return not only answers to queries, but additional info that will likely be needed shortly
      - The "glue records"
- What if attacker eavesdrops on our DNS queries?
  - Then similar to DHCP, ARP, AirPwn etc, can spoof responses
- Consider attackers who can't eavesdrop but still aim to manipulate us via how the protocol functions
- Directly interacting w/ DNS: dig program on Unix
  - Allows querying of DNS system
  - Dumps each field in DNS responses

	dig eecs.mit.edu			-	to look up IP address cs.mit.edu via DNS	
	; ; <<>> DiG 9.6.0-APPL	E-P2 <<>	> eecs.	mit.edu	a	
	;; global options: +cmd					
Computer Science 161 Sp	;; Got answer:					Popa & Weaver
	;; ->>HEADER<<- opcode:	QUERY,	status:	NOERRO	R, id: 19901	
	;; flags: qr rd ra; QUE	RY: 1, A	NSWER:	1, AUTH	ORITY: 3, ADDITIONAL: 3	
	A OUESELON SECETON.					
	;; QUESTION SECTION: ;eecs.mit.edu.		IN	A		
	, eecs.mitt.edu.		TIN	А		
	;; ANSWER SECTION:					
	eecs.mit.edu.	21600	IN	A	18.62.1.6	
	MUMUODINY CECUION.					
	;; AUTHORITY SECTION: mit.edu.	11088	IN	NS	BITSY.mit.edu.	
	mit.edu.	11088	IN	NS		
	mit.edu.	11088	IN	NS	STRAWB.mit.edu.	
	;; ADDITIONAL SECTION:					
	STRAWB.mit.edu.	126738	IN	А	18.71.0.151	
	BITSY.mit.edu.	166408			18.72.0.3	
	W20NS.mit.edu.	126738	IN	A	18.70.0.160	
						51
						51

dig eecs.mit.ed	u A		
; ; <<>> DiG 9.6.0-APPI ;; global options: +cmo ce 161 Spr ;; Got answer: ;; ->>HEADER<<- opcode ;; flags: qr rd ra; QUE	d : QUERY, status	NOERR	
;; <u>QUESTION SECTION:</u> ;eecs.mit.edu.	IN	A	
;; ANSWER SECTION: eecs.mit.edu.	21600 IN	A	18.62.1.6
;; AUTHORITY SECTION: mit.edu.	11088 IN	NS	BITSY.mit.edu.
mit.edu. mit.edu.	11088 IN 11088 IN	NS NS	W20NS.mit.edu. STRAWB.mit.edu.
;; ADDITIONAL SE The que	estion we asked the	server	
STRAWB.mit.edu. BITSY.mit.edu. W20NS.mit.edu.	126738 IN 166408 IN 126738 IN		
WZUNS.MIT.edu.	120/30 IN	A	18.70.0.100

<pre>; ; &lt;&lt;&gt;&gt; DiG 9.6.0-APPI ;; global options: +cmd  r;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode:</pre>	d : QUERY,	status	: NOERRO	DR, id: 19901	
<pre>;; flags: qr rd ra; QUE ;; QUESTION SECTION: ;eecs.mit.edu.</pre>	2RY: 1, A	ANSWER: IN	1, AUTH A	ORITY: 3, ADDITIONAL:	3
;; ANSWER SECTION: eecs.mit.edu.				<b>n identifier</b> that enables , in this case) to match up	
;; AUTHORITY SECTION: mit.edu.				ginal request	
mit.edu. mit.edu.	11088 11088			W20NS.mit.edu. STRAWB.mit.edu.	
;; ADDITIONAL SECTION: STRAWB.mit.edu.	126738 166408	IN IN	A A	18.71.0.151 18.72.0.3	
	100000	IN	7	18.70.0.160	



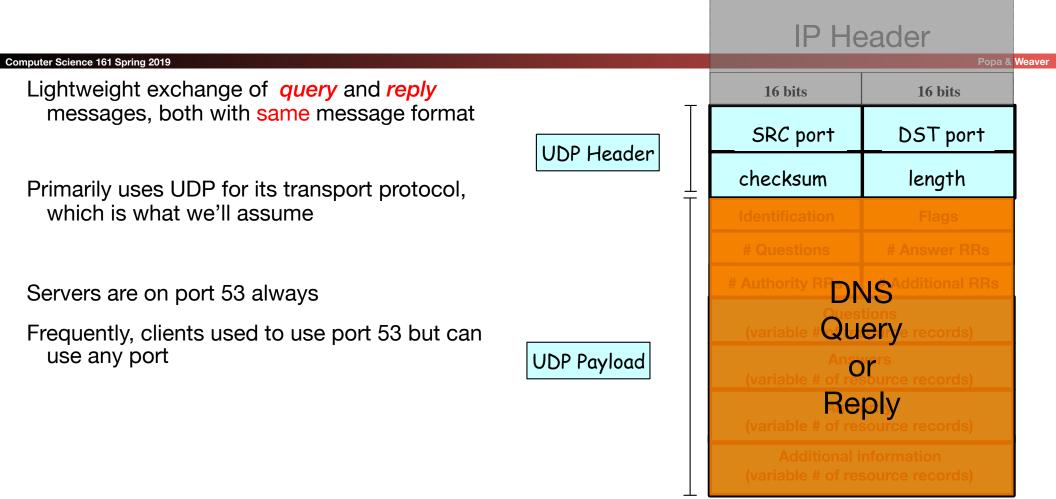
; ; <<>> DiG 9.6.0-APP ;; global options: +cm		>> eecs	.mit.edu	a
;; Got answer: ;; ->>HEADER<<- opcode ;; flags: qr rd ra; QU				R, id: 19901 ORITY: 3, ADDITIONAL:
;; QUESTION SECTION: ;eecs.mit.edu.		IN	A	
;; ANSWER SECTION: eecs.mit.edu.	21600	IN	A	18.62.1.6
;; AUTHORITY SECTION: mit.edu.	11088	IN	NS	BITSY.mit.edu.
mitt.eau.	his include:	s, left-to-	right, a Dl	Record (RR) like NS name, a time-
	o-live, a far a type (A he			
BITSY.mit.edu. W20NS.mit.edu.	166408 126738		A A	18.72.0.3 18.70.0.160

	; ; <<>> DIG 9.6.0-APE					_	
uter Science 161 Sp	<pre>;; global options: +cm ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode ;; flags: qr rd ra; QU</pre>	the answer	. Each er ("NS")	RR gives t for name	e servers responsible for the hostname of a different s in mit.edu. We should 8 seconds.	3	
	;; QUESTION SECTION: ;eecs.mit.edu.				npty, then the resolver's the original query to one of		
	;; ANSWER SECTION: eecs.mit.edu.	these name					
	;; AUTHORITY SECTION:	$\frown$					
	mit.edu.	11088	IN	NS	BITSY.mit.edu.		
	mit.edu.	11088	IN	NS	W20NS.mit.edu.		
	<pre>mit.edu. ;; ADDITIONAL SECTION:</pre>	11088	IN	NS	STRAWB.mit.edu.		
	STRAWB.mit.edu.	126738	IN	A	18.71.0.151		
	BITSY.mit.edu.	166408	IN	A	18.72.0.3		
	W20NS.mit.edu.	126738	IN	A	18.70.0.160		

п

<pre>;; QUESTION SECTION: ;eecs.mit.edu. ;; ANSWER SECTION eecs.mit.edu. ;; AUTHORITY SECT mit.edu. mit.edu. ;; ADDITIONAL SECTION: STRAWB.mit.edu. BITSY.mit.edu. ;; ADDITIONAL SECTION: STRAWB.mit.edu. BITSY.mit.edu. ;; ADDITIONAL SECTION: STRAWB.mit.edu. ;; ADDITIONAL SECTION: ;; ADDITIONAL SECT</pre>	<pre>; ; &lt;&lt;&gt;&gt; DiG 9.6.0 ;; global options: ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- op ;; flags: qr rd ra</pre>	+cmd ocode: QUERY, st	atus: NOERF	
'; ANSWER SECTION         eecs.mit.edu.         '; AUTHORITY SECT         making separate lookups for it, or helps with bootstrapping.         Here, it tells us the IP addresses for the hostnames of the         nit.edu.         11088         IN         NS         BITSY.mit.edu.         11088         IN         NS         STRAWB.mit.edu.         126738         IN         AUTHORITIONAL	;; QUESTION SECTION	N:		
;; ANSWER SECTION       making separate lookups for it, or helps with bootstrapping.         eecs.mit.edu.       Here, it tells us the IP addresses for the hostnames of the name servers. We add these to our cache.         mit.edu.       11088       IN       NS       BITSY.mit.edu.         mit.edu.       11088       IN       NS       BITSY.mit.edu.         mit.edu.       11088       IN       NS       STRAWB.mit.edu.         ;; ADDITIONAL SECTION:       126738       IN       A       18.71.0.151	;eecs.mit.edu.	"Additional" provid	des extra infor	mation to save us from
eecs.mit.edu.Here, it tells us the IP addresses for the hostnames of the name servers. We add these to our cache.mit.edu.11088mit.edu.11088mit.edu.11088mit.edu.11088strawb.mit.edu.11088strawb.mit.edu.126738strawb.mit.edu.18.71.0.151		· · · · · ·		
Here, it tells us the IP addresses for the hostnames of the;; AUTHORITY SECT:name servers. We add these to our cache.mit.edu.11088INNSmit.edu.11088INNSmit.edu.11088INNSstrawb.mit.edu.11088INNSstrawb.mit.edu.11088INNSstrawb.mit.edu.126738INA18.71.0.15118.71.0.151			-	
mit.edu.11088INNSBITSY.mit.edu.mit.edu.11088INNSW20NS.mit.edu.mit.edu.11088INNSSTRAWB.mit.edu.;; ADDITIONAL SECTION:STRAWB.mit.edu.126738INA		Here, it tells us the	IP addresses	for the hostnames of the
mit.edu.11088INNSW20NS.mit.edu.mit.edu.11088INNSSTRAWB.mit.edu.;; ADDITIONAL SECTION:STRAWB.mit.edu.126738INASTRAWB.mit.edu.126738INA18.71.0.151	;; AUTHORITY SECT:	name servers. We	add these to d	our cache.
mit.edu. 11088 IN NS STRAWB.mit.edu. ;; ADDITIONAL SECTION: STRAWB.mit.edu. 126738 IN A 18.71.0.151	mit.edu.	11088 1	IN NS	BITSY.mit.edu.
;; ADDITIONAL SECTION: STRAWB.mit.edu. 126738 IN A 18.71.0.151	mit.edu.	11088 1	IN NS	W20NS.mit.edu.
STRAWB.mit.edu. 126738 IN A 18.71.0.151	mit.edu.	11088 1	IN NS	STRAWB.mit.edu.
STRAWB.mit.edu. 126738 IN A 18.71.0.151	ADDITIONAL SECT	TON		
				18 71 0 151
W20NS.mit.edu. 126738 IN A 18.70.0.160	DIIOI IMICICAA.			

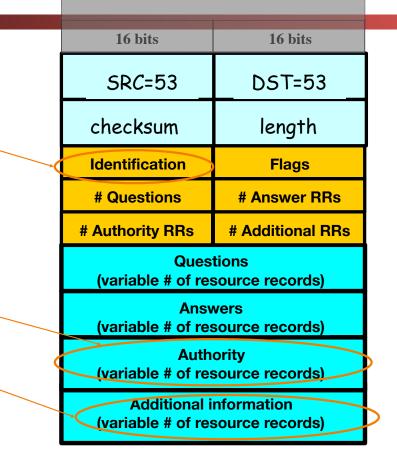
### **DNS Protocol**



#### **IP** Header

#### Message header:

- Identification: 16 bit # for query, reply to query uses same #
- Along with repeating the Question and providing Answer(s), replies can include "Authority" (name server responsible for answer) and "Additional" (info client is likely to look up soon anyway)
- Each Resource Record has a Time To Live (in seconds) for caching (not shown)



	dig eecs.mit.edu	A					
Computer Science 161 Spi	<pre>; ; &lt;&lt;&gt;&gt; DiG 9.6.0-APPLE- ;; global options: +cmd ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode: ()</pre>					01	Popa & Weaver
	;; flags: qr rd ra; QUERY	r: W	hat if th	e mit.e	du server		
	;; QUESTION SECTION: ;eecs.mit.edu.				? Could , say, all		
	;; ANSWER SECTION: eecs.mit.edu. 2		our we erkeley.e		•	. 6	
		Se 11088 11088		NS NS	BITSY.m W20NS.m		
	<pre>mit.edu. 1 ;; ADDITIONAL SECTION:</pre>	11088	IN	NS	STRAWB.	mit.edu.	
	BITSY.mit.edu.	12673 16640 12673	8 IN	A A A	18.71.0 18.72.0 18.70.0	.3	
					10.70.0		
							60

п

	u A				
; ; <<>> DiG 9.6.0-APP		> eecs	.mit.edu	ıa	
;; global options: +cm nce161 Spr;; Got answer:	a				
;; ->>HEADER<<- opcode	· OIIFRY	et=t11e	· NOFPRO	P id: 19901	
;; flags: gr rd ra; QU	~ /				
,, 110gb: q1 10 10, g0.			1, 11011		
;; QUESTION SECTION:				the state of	
;eecs.mit.edu.				w in the	
	orig	jinal D	NS des	ign	
;; ANSWER SECTION:	216 (sin	co five	ъ <b>с</b> )	-	
eecs.mit.edu.	216(5)		50)	. 6	
;; AUTHORITY SECTION:	11000	<b>T</b> 11	270		
mit.edu.	11088	IN	NS	BITSY.mit.edu.	
mit.edu.		IN	NS	W20NS.mit.edu.	
mit.edu.	11088	IN	NS	STRAWB.mit.edu.	
;; ADDITIONAL SECTION:					
STRAWB.mit.edu.	126738	IN	A	18.71.0.151	
BITSY.mit.edu.	166408	IN	A	18.72.0.3	
W20NS.mit.edu.	126738	IN	А	18.70.0.160	

<pre>dig eecs.mit.e ; ; &lt;&lt;&gt;&gt; DiG 9.6.0-A ;; global options: + cience 161 Spr ;; Got answer:</pre>	APPLE-P2 <<> cmd			
;; ->>HEADER<<- opco				OR, id: 19901 CORITY: 3, ADDITIONAL: 3
<pre>;; QUESTION SECTION: ;eecs.mit.edu. ;; ANSWER SECTION:</pre>	Wha			he mit.edu server o us instead?
eecs.mit.edu.	21600	IN	A	18.62.1.6
;; AUTHORITY SECTION mit.edu. mit.edu. mit.edu.	11088 11088	IN IN IN	NS NS <mark>NS</mark>	
;; ADDITIONAL SECTIO				
www.berkeley.edu.				18.6.6.6
	1 ( ( 1 0 0	IN	A	18.72.0.3
BITSY.mit.edu. W20NS.mit.edu.	166408 126738		A	18.70.0.160

dig eecs.mit.e	edu A			
<pre>; ; &lt;&lt;&gt;&gt; DiG 9.6.0-Z ;; global options: + prScience 161 Spr ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opco ;; flags: qr rd ra;</pre>	⊦cmd ode: QUERY, status	: NOERI		Popa & W
;; QUESTION SECTION:		7		
;eecs.mit.edu.	IN	A		
;; ANSWER SECTION: eecs.mit.edu.	-	lu <b>to an</b> l	IP address under MIT's	
;; AUTHORITY SECTION	•		any IP address they s.)	
mit.edu.	11088 IN	NS	BITSY.mit.edu.	
mit.edu.	11088 IN	NS	W20NS.mit.edu.	
mit.edu.	11088 IN	NS	www.berkeley.edu.	
;; ADDITIONAL SECTIO	)N:			
www.berkeley.edu.	100000 IN	Α	18.6.6.6	
BITSY.mit.edu.	166408 IN	A	18.72.0.3	
W20NS.mit.edu.	126738 IN	A	18.70.0.160	

<pre>dig eecs.mit.edu ; ; &lt;&lt;&gt;&gt; DiG 9.6.0-APPI ;; global options: +cmd Spr ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode:</pre>	LE-P2 <<>> eecs.		
			IORITY: 3, ADDITIONAL: 3
;; QUESTION SECTION: ;eecs.mit.edu.	IN	A	
;; ANSWER SECTION: eecs.mit.edu.	In this case they mapping last a lo just as easily ma	ong time.	They could 6
;; AUTHORITY SECTION:	of seconds.		
mit.edu.	11088 IN	NS	BITSY.mit.edu.
mit.edu.	11088 IN	NS	W20NS.mit.edu.
mit.edu.	11088 IN	NS	www.berkeley.edu.
;; ADDITIONAL SECTION:			
www.berkeley.edu.	(100000) IN	A	18.6.6.6
BITSY.mit.edu.	166408 IN	А	18.72.0.3
BIISI.MIC. edu.		А	18.70.0.160

	dig eecs.mit.	edu A			
er Science 161 Sp	<pre>; ; &lt;&lt;&gt;&gt; DiG 9.6.0 ;; global options: ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opc ;; flags: qr rd ra;</pre>	+cmd ode: QUERY,	status	s: NOERRO	
	;; QUESTION SECTION ;eecs.mit.edu.		IN	А	,
	;; ANSWER SECTION: eecs.mit.edu.	How do v	ve fi>	such (	cache poisoning?
	;; AUTHORITY SECTIO	и:			
	mit.edu.	11088	IN	NS	BITSY.mit.edu.
	mit.edu.	11088	IN	NS	W20NS.mit.edu.
	mit.edu.	30	IN	NS	www.berkeley.edu.
	;; ADDITIONAL SECTI	ON:			
	www.berkeley.edu.	30	IN	A	18.6.6.6
		166409	TN	А	18.72.0.3
	BITSY.mit.edu.	100408			

Computer Science 161 Spr		9.6.0-API cions: +cr				Popa & Weaver	
	;; flags: qr	_	E.g., look additiona				
	;; QUESTION S ;eecs.mit.edu ;; ANSWER SEC eecs.mit.edu.	I. CTION:	them to us	No extra risk in accepting these since server could return hem to us directly in an <b>Answer</b> anyway. This is called " <b>Bailiwick</b> checking"			
	;; AUTHORITY	SECTION:					
	mit.edu. mit.edu. mit.edu.		11088 11088 <mark>11088</mark>	IN IN IN	bail·i·wick /ˈbālə,wik/ -⊛		
	;; ADDITIONAL SECTION www.berkeley.edu BITSY.mit.edu. W20NS.mit.edu.		100000 - IN       noun         166408 IN       1. one's sphere of operations or particul "you never give the presentations—the sphere of operations or particul "you never give the presentations—the sphere of operation of a bailie or the district or jurisdiction of a bailie or the sphere of operation of a bailie or the sphere of operations or particul "you never give the presentations—the sphere of operation of a bailie or the sphere of operation of a bailie of the sphere of operatic definition of a bailie of the sphere of		at's my bailiwick"		
						00	

### DNS Resource Records and RRSETs

#### Computer Science 161 Spring 2019

- DNS records (Resource Records) can be one of various types
  - Name TYPE Value
    - · Also a "time to live" field: how long in seconds this entry can be cached for
  - Addressing:
    - A: IPv4 addresses
    - AAAA: IPv6 addresses
    - CNAME: aliases, "Name X should be name Y"
    - MX: "the mailserver for this name is Y"
  - DNS related:
    - NS: "The authority server you should contact is named Y"
    - SOA: "The operator of this domain is Y"
  - Other:
    - text records, cryptographic information, etc....
- Groups of records of the same type form RRSETs:
  - E.g. all the nameservers for a given domain.

### The Many Moving Pieces In a DNS Lookup of <u>www.isc.org</u>

#### Computer Science 161 Spring 2019



#### ? A www.isc.org

User's ISP's ? A www.isc.org

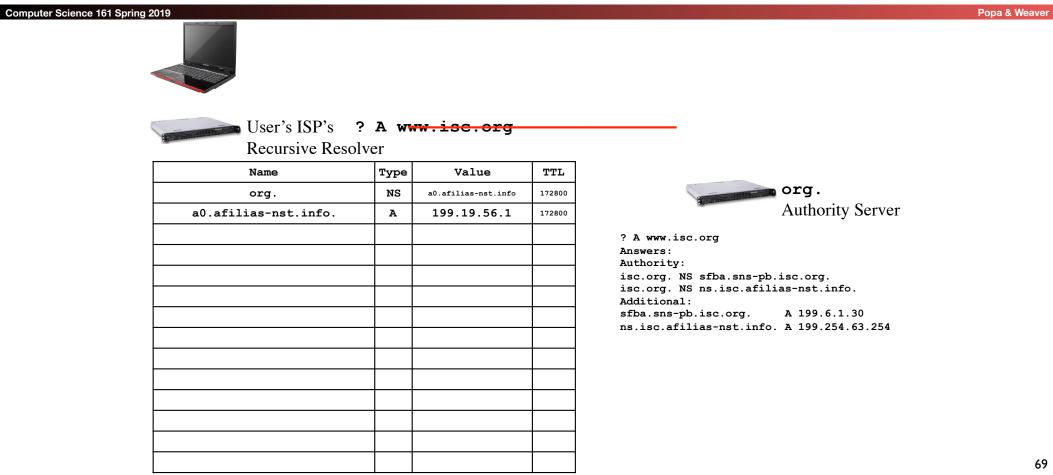
**Recursive Resolver** 

Name	Туре	Value	TTL



Authority Server Answers: (the "root") Authority: org. NS a0.afilias-nst.info Additional: a0.afilias-nst.info A 199.19.56.1

### The Many Moving Pieces In a DNS Lookup of www.isc.org



### The Many Moving Pieces In a DNS Lookup of <u>www.isc.org</u>

Computer Science 161 Spring 2019





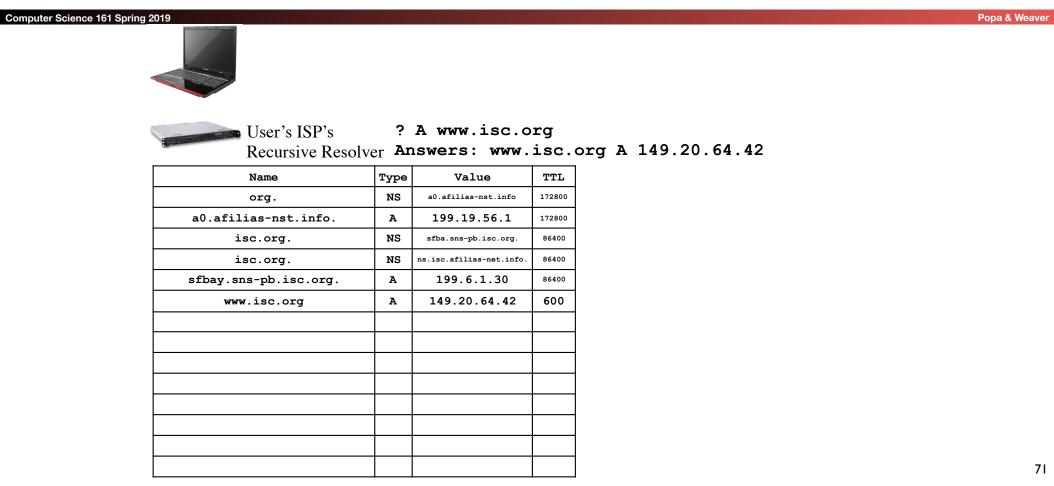
User's ISP's ? A www.isc.org

Recursive Resolver

Name	Туре	Value	TTL
org.	NS	a0.afilias-nst.info	172800
a0.afilias-nst.info.	A	199.19.56.1	172800
isc.org.	NS	sfba.sns-pb.isc.org.	86400
isc.org.	NS	ns.isc.afilias-net.info.	86400
sfbay.sns-pb.isc.org.	A	199.6.1.30	86400



### The Many Moving Pieces In a DNS Lookup of www.isc.org



# Stepping Through This With **dig**

#### Computer Science 161 Spring 2019

- Some flags of note:
  - +norecurse: Ask directly like a recursive resolver does
  - +trace: Act like a recursive resolver without a cache

```
nweaver% dig +norecurse slashdot.org @a.root-servers.net
; <<>> DiG 9.8.3-P1 <<>> +norecurse slashdot.org @a.root-servers.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26444
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 12
;; QUESTION SECTION:
;slashdot.org.
                                IN
                                         Α
;; AUTHORITY SECTION:
org.
                        172800 IN
                                         NS
                                                 a0.org.afilias-nst.info.
. . .
;; ADDITIONAL SECTION:
                                                 199.19.56.1
a0.org.afilias-nst.info. 172800 IN
                                         Α
```

## So in dig parlance

#### Computer Science 161 Spring 2019

- So if you want to recreate the lookups conducted by the recursive resolver:
  - dig +norecurse www.isc.org @a.root-servers.net
  - dig +norecurse www.isc.org @199.19.56.1
  - dig +norecurse www.isc.org @199.6.1.30

## Security risk #1: malicious DNS server

```
Popa & Weave
```

- Of course, if *any* of the DNS servers queried are malicious, they can lie to us and fool us about the answer to our DNS query...
- and they used to be able to fool us about the answer to other queries, too, using *cache poisoning*. Now fixed (phew).

## Security risk #2: on-path eavesdropper

### Computer Science 161 Spring 2019

- If attacker can eavesdrop on our traffic... we're hosed.
- Why?

## Security risk #2: on-path eavesdropper

### Computer Science 161 Spring 2019

- If attacker can eavesdrop on our traffic... we're hosed.
- Why? They can see the query and the 16-bit transaction identifier, and race to send a spoofed response to our query.
  - China does this operationally:
    - Note: You may need to use the IPv4 address of www.tsinghua.edu.cn
  - dig www.benign.com @www.tsinghua.edu.cn
  - dig www.facebook.com @www.tsinghua.edu.cn

## Security risk #3: off-path attacker

```
Popa & Weaver
```

- If attacker can't eavesdrop on our traffic, can he inject spoofed DNS responses?
- Answer: It used to be possible, via *blind spoofing*.
   We've since deployed mitigations that makes this harder (but not totally impossible).

# Blind spoofing

#### Computer Science 161 Spring 2019

- Say we look up mail.google.com; how can an off-path attacker feed us a bogus A answer before the legitimate server replies?
- How can such a remote attacker even know we are looking up
   Suppose, e.g., we visit a web page under their control:

	16 bits	16 bits				
	SRC=53	DST=53				
	checksum	length	1			
	Identification	Flags				
	# Questions	# Answer RRs				
	# Authority RRs	# Additional RRs				
	Questions (variable # of resource records)					
$\langle$	Answers (variable # of resource records)					
	Authority (variable # of resource records)					
	Additional information (variable # of resource records)					

...<img src="http://mail.google.com" ...> ...

# Blind spoofing

	16 bits	16 bits	
Computer Science 161 Spring 2019	SRC=53	DST=53	Popa & Wea
<ul> <li>Say we look up</li> </ul>	checksum	length	
mail.google.com; how can	Identification	Flags	
an off-path attacker feed us a	# Questions	# Answer RRs	
bogu <u>s</u> A answer before the	# Authority RRs	# Additional RRs	
<ul> <li>Iegitin This HTML snippet causes browser to try to fetch an</li> <li>How mail.google.com. To ceven browser first has to look umail address associated with the Suppose, e.g., we visit a web page under their control:</li> </ul>	image from that, our the IP hat name.	ations source records) wers source records) hority source records) information source records)	

...<img src="http://mail.google.com" ...> ...

aver

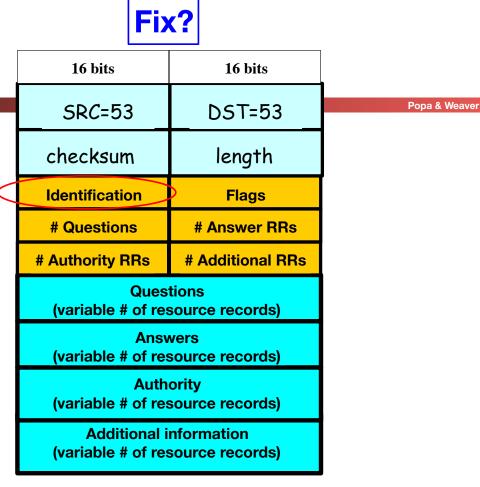
# Blind spoofing

#### Computer Science 161 Spring 2019

Once they know we're looking it up, they just have to guess the Identification field and reply before legit server.

How hard is that?

Originally, identification field incremented by 1 for each request. How does attacker guess it?



<img src="http://badguy.com" ...> They observe ID k here
<img src="http://mail.google.com" ...>So this will be k+1

80

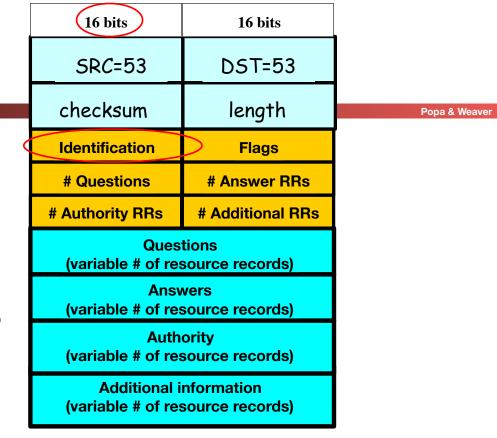
## DNS Blind Spoofing, cont.

#### Computer Science 161 Spring 2019

Once we randomize the Identification, attacker has a 1/65536 chance of guessing it correctly. Are we pretty much safe?

Attacker can send lots of replies, not just one ...

However: once reply from legit server arrives (with correct Identification), it's **cached** and no more opportunity to poison it. Victim is innoculated!



Unless attacker can send 1000s of replies before legit arrives, we're likely safe – phew! **?** 

# Enter Kaminski... Glue Attacks

Computer Science 161 Spring 2019					Popa & Weaver
<ul> <li>Dan Kaminski noticed</li> </ul>	nweaver% dig +norecurse	slashdot	c.org @a	root-se:	rvers.net
something strange, however	<pre>; &lt;&lt;&gt;&gt; DiG 9.8.3-P1 &lt;&lt;&gt;&gt; +norecurse slashdot.org @a.root-servers.net ;; global options: +cmd ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 26444</pre>				
<ul> <li>Most DNS servers would cache the in-bailiwick glue</li> </ul>	<pre>;; flags: qr; QUERY: 1, ;; QUESTION SECTION: ;slashdot.org.</pre>	ANSWER:	0, AUTH	HORITY: 6, A	, ADDITIONAL: 12
<ul> <li>And then <i>promote</i> the glue</li> </ul>	;; AUTHORITY SECTION:				
<ul> <li>And will also <i>update</i> entries based on glue</li> </ul>		172800	IN	NS	a0.org.afilias-nst.info
<ul> <li>So if you first did this</li> </ul>	;; ADDITIONAL SECTION: a0.org.afilias-nst.info. 	172800	IN	А	199.19.56.1
<ul> <li>IOOKUP</li> <li>And then went to a0.org.afilias-nst.info</li> </ul>	;; Query time: 128 msec ;; SERVER: 198.41.0.4#53 ;; WHEN: Tue Apr 16 09:4 ;; MSG SIZE rcvd: 432	-	-		

• there would be no other lookup!

# The Kaminski Attack In Practice

- Rather than trying to poison www.google.com...
- Instead try to poison a.google.com...
   And state that "www.google.com" is an authority
   And state that "www.google.com A 133.7.133.7"
  - If you succeed, great!
- But if you fail, just try again with b.google.com!
  - Turns "Race once per timeout" to "race until win"
- So now the attacker may still have to send lots of packets
  - In the 10s of thousands
- The attacker can keep trying until success

# Defending Against Kaminski: Up the Entropy

### Computer Science 161 Spring 2019

- Also randomize the UDP source port
  - Adds 16 bits of entropy
- Observe that most DNS servers just copy the request directly
  - Rather than create a new reply
- So caMeLcase the NamE ranDomly
  - Adds only a few bits of entropy however, but it does help

# Defend Against Kaminski: Validate Glue

### Computer Science 161 Spring 2019

- Don't blindly accept glue records...
  - Well, you *have* to accept them for the purposes of resolving a name
- But if you are going to cache the glue record...
- Either only use it for the context of a DNS lookup
- No more promotion
- Or explicitly validate it with another fetch
- Unbound implemented this, bind did not
  - Largely a *political* decision: bind's developers are heavily committed to DNSSEC (next week's topic)

# Oh, and Profiting from Rogue DNS

- Suppose you take over a lot of home routers...
  - How do you make money with it?
- Simple: Change their DNS server settings
  - Make it point to yours instead of the ISPs
- Now redirect all advertising
  - And instead serve up ads for "Vimax" pills...

