



CSE 127: Computer Security
Network Security

Kirill Levchenko

November 28, 2017

Network Security

- ❖ **Original TCP/IP design:** Trusted network and hosts
 - Hosts and networks administered by mutually trusted parties
- ❖ **15 years ago:** Hosts can't be trusted
 - Anyone can connect to public Internet
 - Untrusted insiders on internal networks
- ❖ **Today:** Network can't be trusted either
 - Wifi has taken the last shred of trust we had in network

Trust that ...

- ❖ Network protocols used only as intended
 - Packet headers filled out correctly
 - Rate limiting of costly operations
- ❖ Hosts controlled by trusted administrators
 - Control of access to network by untrusted parties
 - Correct information reported by hosts
 - Protocols implemented correctly

Attacker Models

- ❖ **Man in the middle:** can see, block, and modify traffic
 - Attacker controls wifi access point
- ❖ **Passive:** Eavesdrop on traffic
 - Attacker has passive tap or recorded traces
- ❖ **Off-path:** attacker can inject traffic into network
 - Anyone with access to network

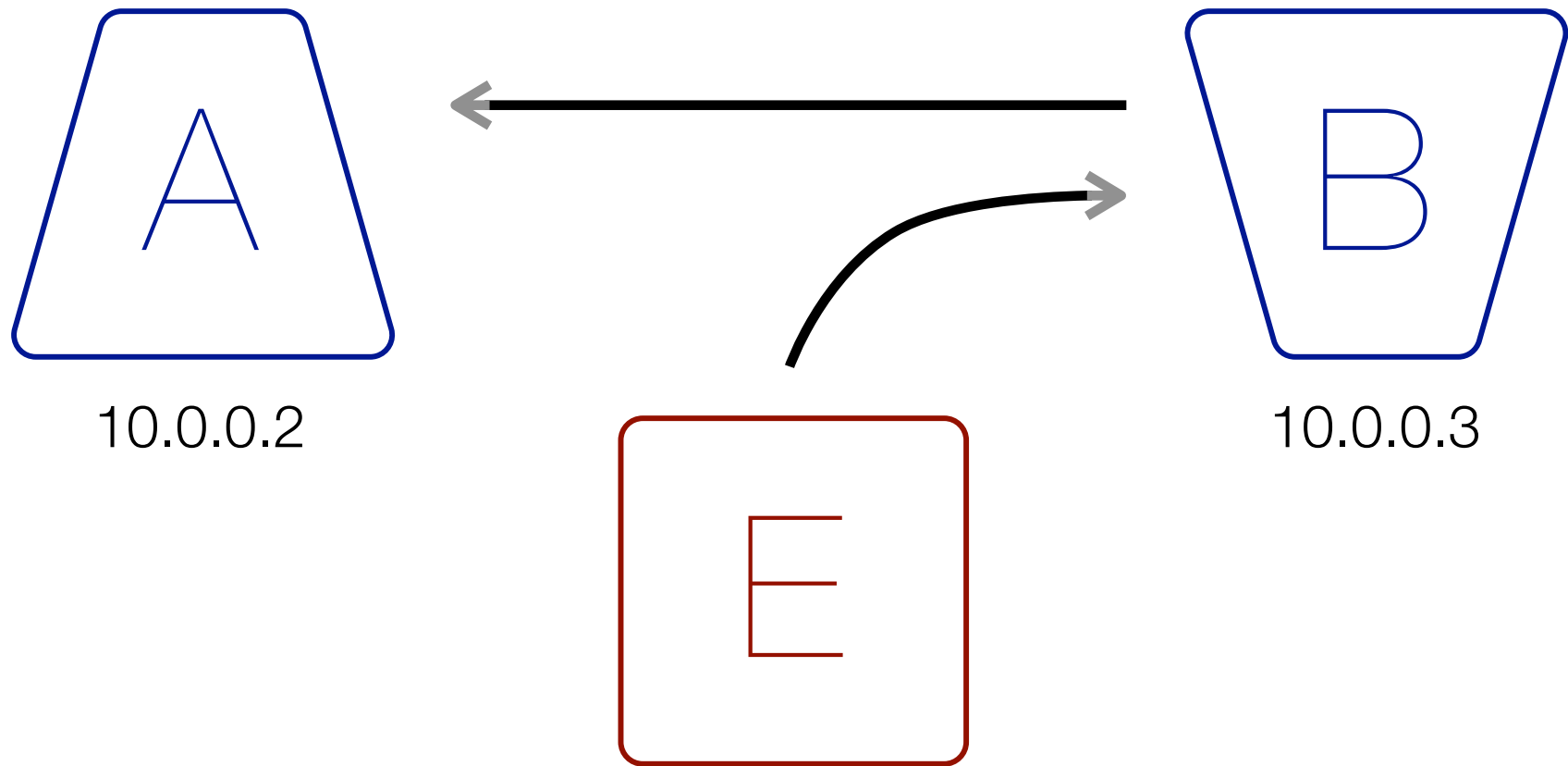
Secrecy

- ❖ Who can see the packets you send?
 - Network (routers, switches, access points, etc.)
 - Unprotected WiFi network: everyone
 - WPA2 Personal (PSK): everyone on same network
 - Non-switched Ethernet: everyone on same network
 - Switched Ethernet: maybe everyone on same network

No Authentication

- ❖ TCP/IP offers no authentication of packets
 - Source address in IP header set by sender
- ❖ Attacker with direct access to network (including MitM) can *spoof* source address
 - Spoof: forge, set to arbitrary value
- ❖ Connectionless protocols (UDP) especially vulnerable

TCP Connection Spoofing

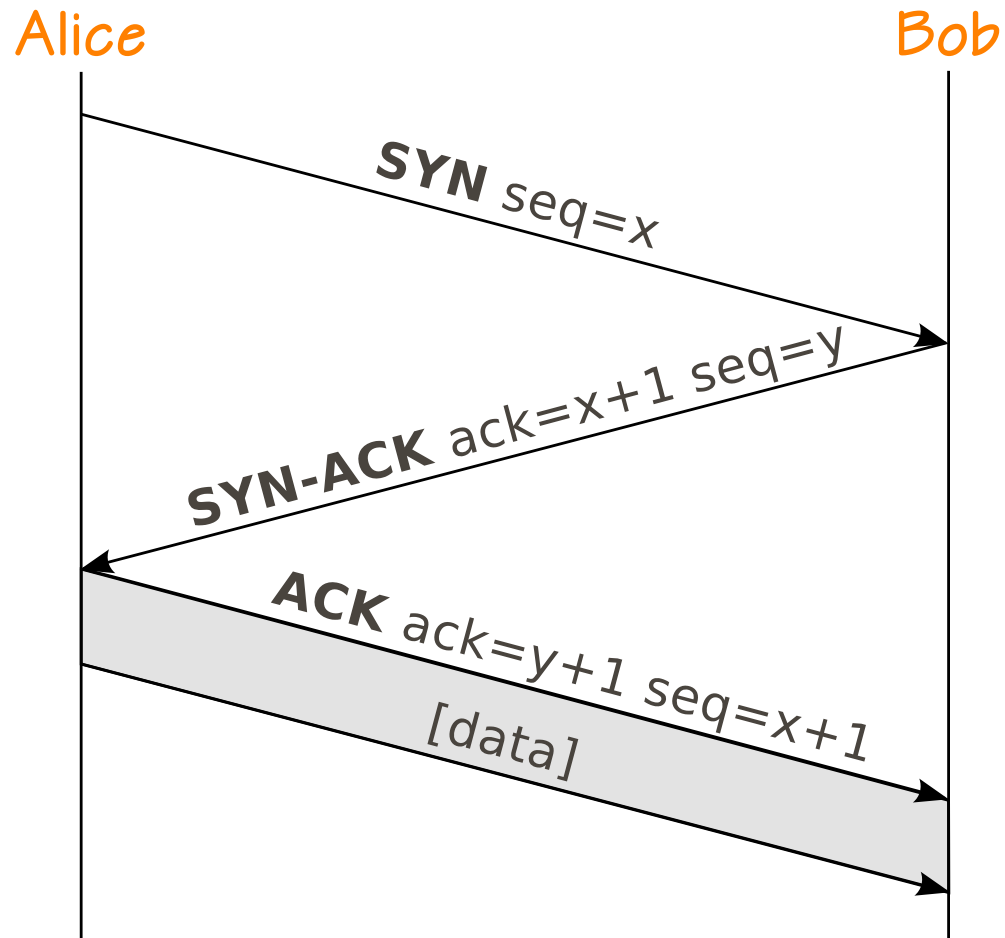


What prevents Eve from impersonating Alice to Bob?
(Assume Alice has direct network access)

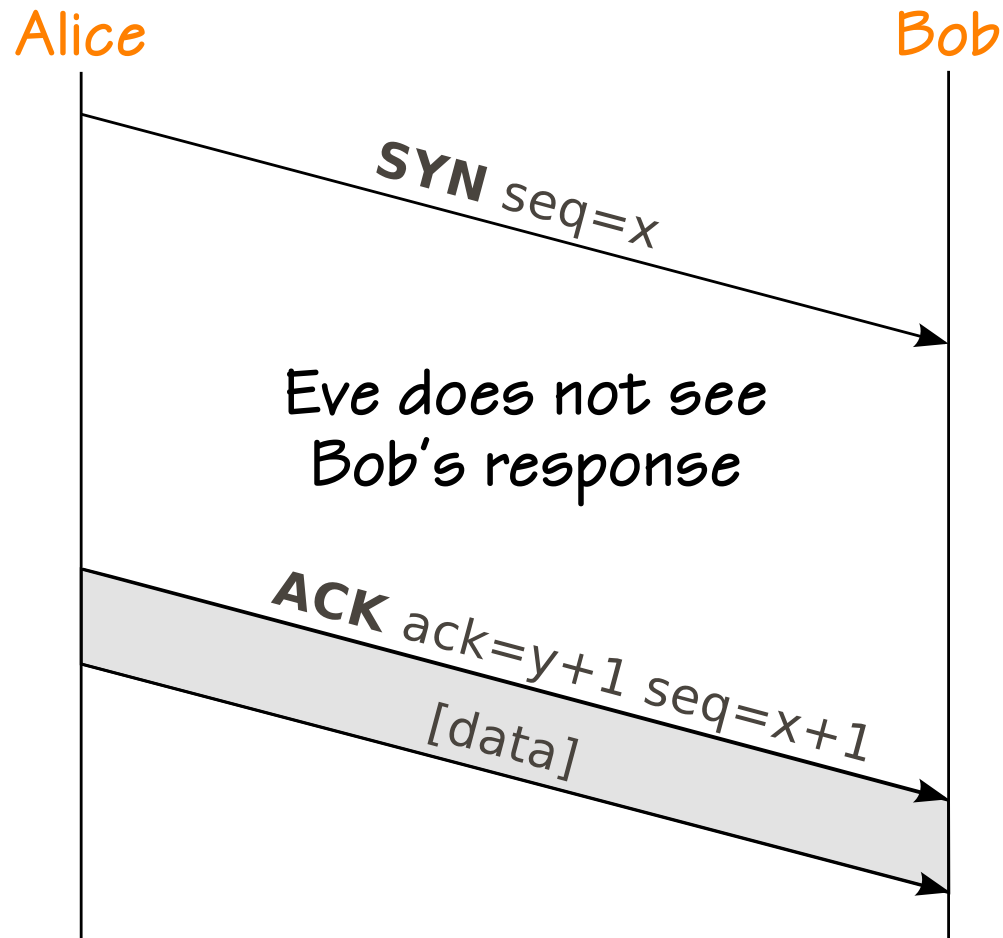
TCP Connection Spoofing

- ❖ Eve needs to complete the TCP three-way handshake between “Alice” and Bob
- ❖ Eve can't see traffic between Alice and Bob
 - “TCP off-path attack”

Three-Way Handshake

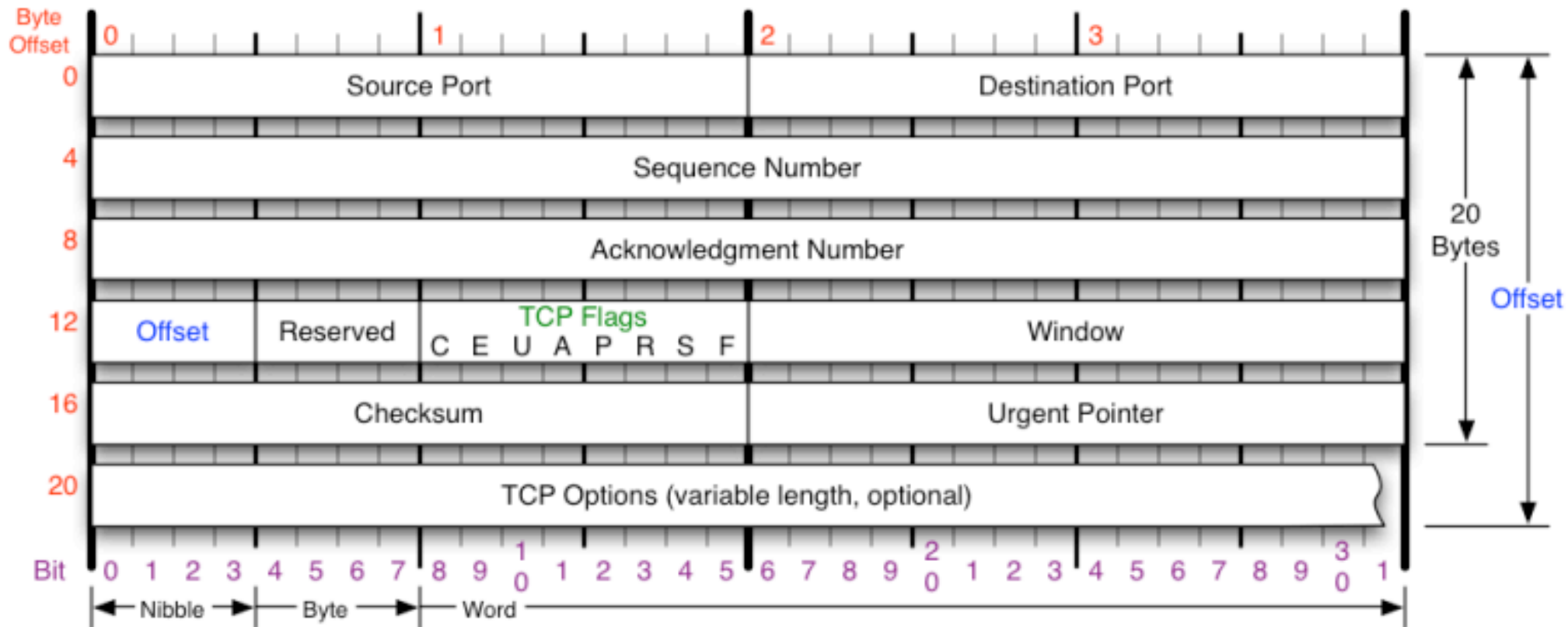


Three-Way Handshake



TCP Connection Spoofing

- ❖ Eve needs to complete the TCP three-way handshake between “Alice” and Bob
- ❖ Eve can't see traffic between Alice and Bob
 - “TCP off-path attack”
- ❖ Eve needs to guess initial sequence number y in order to correctly ACK Bob's SYN



TCP Flags

C E U A P R S F

Congestion Window

C 0x80 Reduced (CWR)

E 0x40 ECN Echo (ECE)

U 0x20 Urgent

A 0x10 Ack

P 0x08 Push

R 0x04 Reset

S 0x02 Syn

F 0x01 Fin

Congestion Notification

ECN (Explicit Congestion Notification). See RFC 3168 for full details, valid states below.

Packet State	DSB	ECN bits
Syn	00	11
Syn-Ack	00	01
Ack	01	00
No Congestion	01	00
No Congestion	10	00
Congestion	11	00
Receiver Response	11	01
Sender Response	11	11

TCP Options

- 0 End of Options List
- 1 No Operation (NOP, Pad)
- 2 Maximum segment size
- 3 Window Scale
- 4 Selective ACK ok
- 8 Timestamp

Checksum

Checksum of entire TCP segment and pseudo header (parts of IP header)

Offset

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

RFC 793

Please refer to RFC 793 for the complete Transmission Control Protocol (TCP) Specification.

TCP Connection Spoofing

- ❖ The sequence number field is 32 bits
- ❖ Early implementations just incremented a global counter used to initialize sequence numbers for TCP connections
 - RFC 793 requires counter incrementing every 4 μ s (250 kHz)
 - Early BSD kernels incremented by a large constant every second
- ❖ Later pseudo-random number generators were used
 - PRNGs were still global, weaknesses allowed guessing

TAKEDOWN

**THE PURSUIT AND CAPTURE
OF KEVIN MITNICK,
AMERICA'S MOST WANTED
COMPUTER OUTLAW
—BY THE MAN WHO DID IT**



**TSUTOMU
SHIMOMURA**

with JOHN MARKOFF

Network Routing (Host View)

- ❖ Say I want to send packet to 8.8.8.8 ...
- ❖ Step 1: Is host on local network?
 - Check subnet masks of local networks

Status: **Connected**

Ethernet 1 is currently active and has the IP address 132.239.17.19.

Configure IPv4:

IP Address:

Subnet Mask:

Router:

DNS Server:

Search Domains:

802.1X:

Network Routing (Host View)

- ❖ Say I want to send packet to 8.8.8.8 ...
- ❖ Step 1: Is host on local network?
 - Local: send directly
 - Not local: send via default gateway

Status: **Connected**

Ethernet 1 is currently active and has the IP address 132.239.17.19.

Configure IPv4:

IP Address:

Subnet Mask:

Router:

DNS Server:

Search Domains:

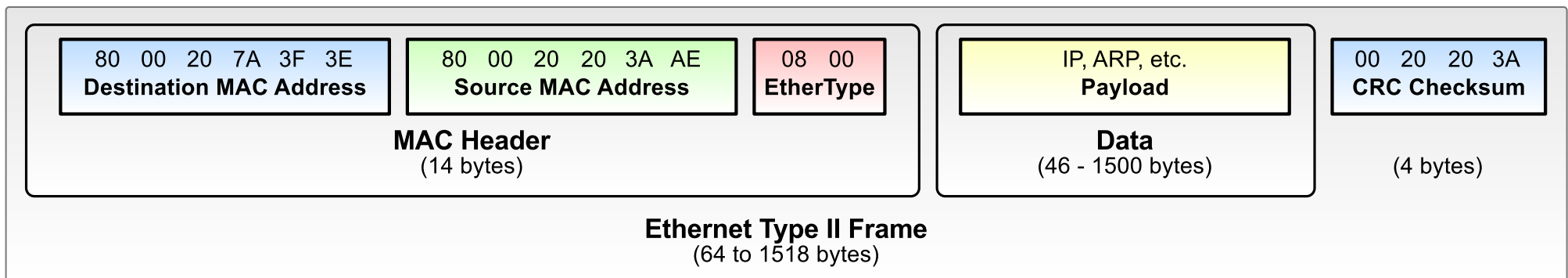
802.1X:

Network Routing (Host View)

- ❖ Say I want to send packet to 8.8.8.8 ...
- ❖ Step 1: Is host on local network?
 - Local: send directly
 - Not local: send via default gateway
- ❖ Step 2: Create IP packet
- ❖ Step 3: Create link layer (e.g. Ethernet) packet

Local Network Attacks

❖ Ethernet frame:



❖ Host needs to fill in Ethernet destination address

- MAC address of host on local network
- MAC address of gateway for host not on local network

ARP

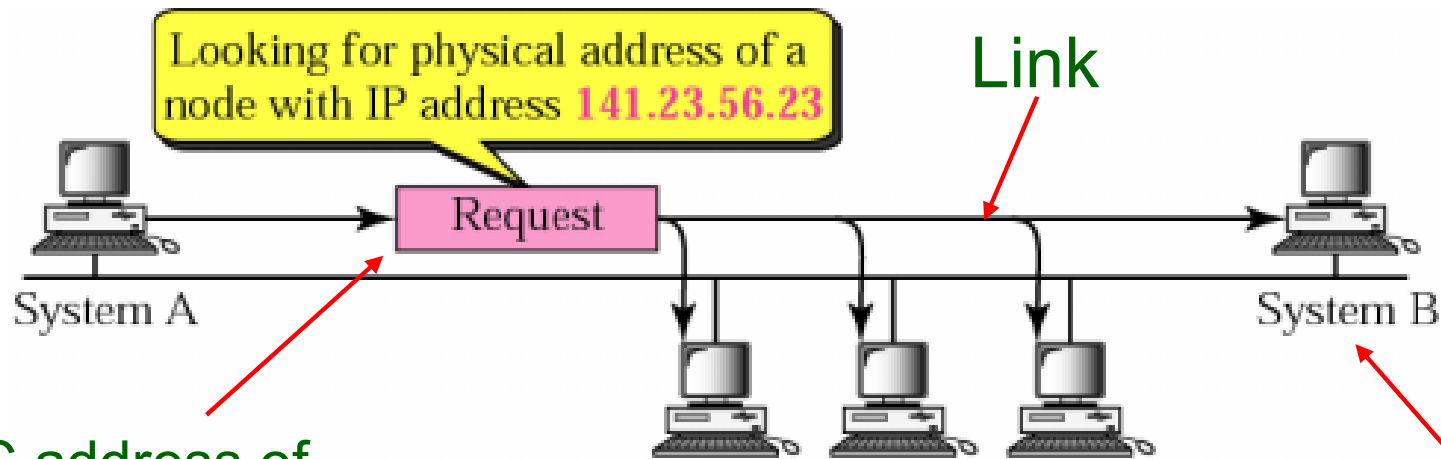
- ❖ Hosts know only IP address of hosts on local network
- ❖ Need a way to associate IP addresses with link-layer addresses (e.g. Ethernet MAC addresses)
- ❖ Address Resolution Protocol (ARP) used to query hosts on local network to get MAC address for an IP address

ARP

- ❖ Alice (looking for Bob's IP) broadcasts:
“What is the MAC address of 10.0.0.3?”
- ❖ Bob sees broadcast and replies:
“The MAC address of 10.0.0.3 is 01:02:03:04:05:06.”
- ❖ Alice sends IP packet for 10.0.0.3 in an Ethernet frame to 01:02:03:04:05:06.

ARP operation

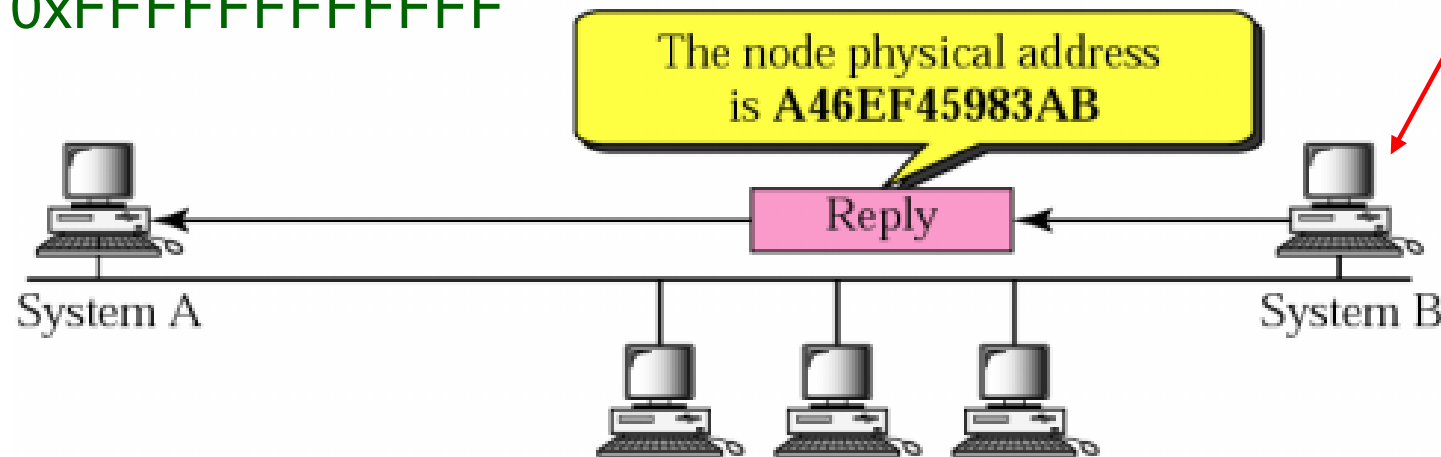
(case: destination is on the same physical network)



a. ARP request is broadcast

The MAC address of destination is broadcast address: 0xFFFFFFFF

IP = 141.23.56.23



b. ARP reply is unicast

ARP packet

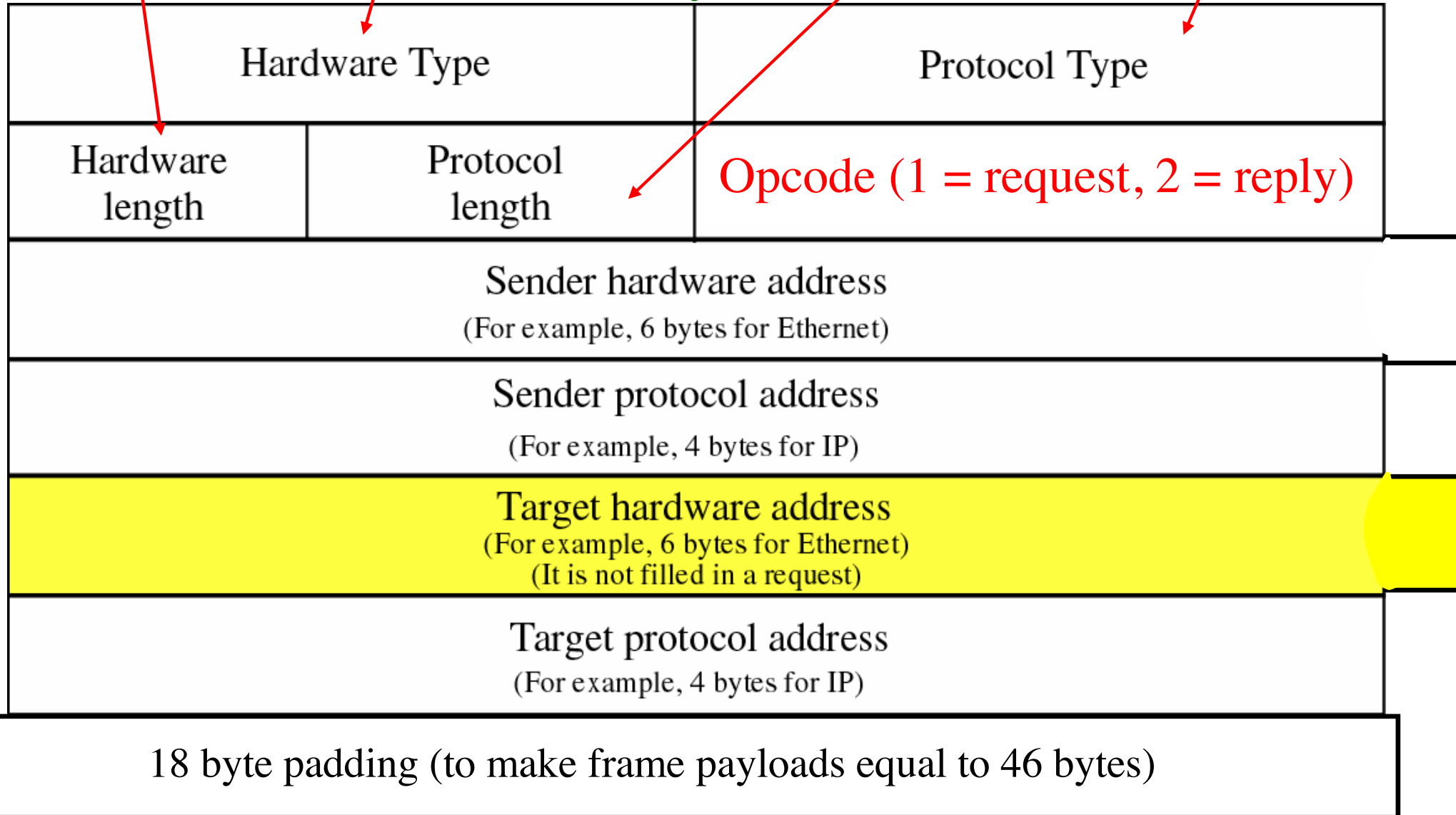
(Ethernet = 6)

(Ethernet = 1)

4 bytes

IPv4 = 4

IPv4 = 0x0800



ARP in Action

```
% sudo tcpdump -v -n -i en0
```

```
tcpdump: listening on en0, link-type EN10MB (Ethernet), capture size 65535 bytes
```

```
10:18:46.827423 ARP, Ethernet (len 6), IPv4 (len 4),
```

```
Request who-has 132.239.17.70 tell 132.239.17.1, length 46
```

```
10:18:47.026680 ARP, Ethernet (len 6), IPv4 (len 4),
```

```
Request who-has 132.239.17.6 tell 132.239.17.1, length 46
```

```
10:18:47.055478 IP (tos 0x0, ttl 111, id 1534, offset 0, flags [none], proto UDP (
```

```
67.183.107.84.27222 > 132.239.17.19.6881: UDP, length 101
```

```
% arp -a
```

```
cse-hazard-gateway.ucsd.edu (132.239.17.1) at 10:8c:cf:57:10:0 on en0 ifscope  
[ethernet]
```

```
xor.ucsd.edu (132.239.17.5) at 0:1b:21:42:80:20 on en0 ifscope [ethernet]
```

```
cselab1.ucsd.edu (132.239.17.111) at 0:22:19:58:2d:d on en0 ifscope [ethernet]
```

```
3dic.ucsd.edu (132.239.17.196) at d4:be:d9:ca:a2:16 on en0 ifscope [ethernet]
```

```
coke.ucsd.edu (132.239.17.244) at 0:1c:c4:d:90:a8 on en0 ifscope [ethernet]
```

ARP Spoofing

- ❖ Anyone can send an ARP reply
- ❖ Attacker on the network can impersonate any other host
- ❖ **Mitigation**
 - **Fixed ARP tables:** impractical for all but small fixed networks
 - **Port binding on switch:** restrict MAC and IP addresses allowed on a physical port switch
- ❖ Higher level host authentication
 - E.g. SSH or SSL

DNS Resolution

- ❖ Say I want to send packet to `www.cs.ucsd.edu` ...
- ❖ Need IP address for `www.cs.ucsd.edu`
- ❖ DNS: distributed system for resolving domain names
 - Resolve: translate domain name to IP address

Status: **Connected**

Ethernet 1 is currently active and has the IP address 132.239.17.19.

Configure IPv4:

IP Address:

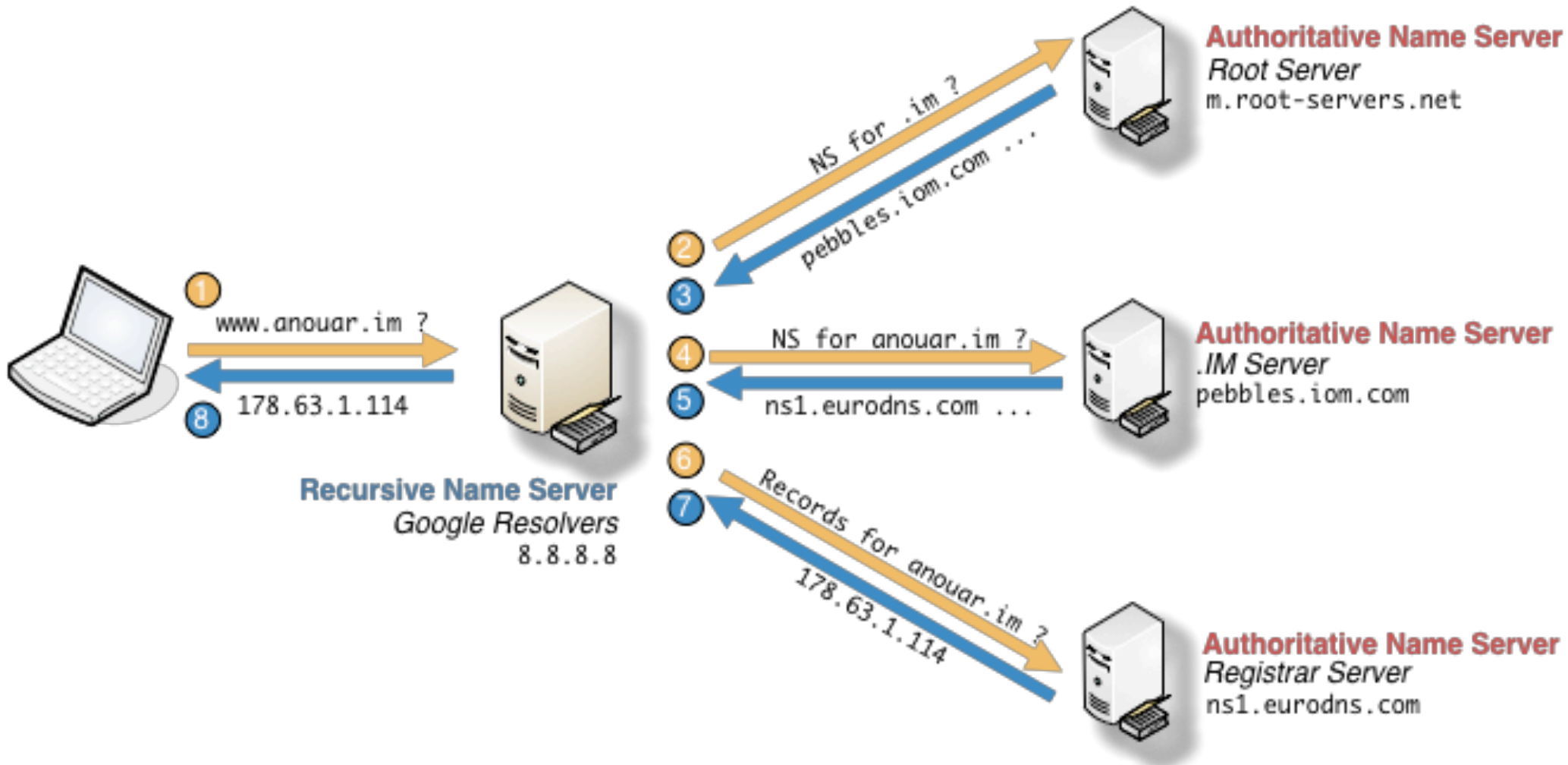
Subnet Mask:

Router:

DNS Server:

Search Domains:

802.1X:



DNS in Action

```
% dig +qr bob.ucsd.edu
```

```
; <<>> DiG 9.6-ESV-R4-P3 <<>> +qr bob.ucsd.edu  
;; global options: +cmd  
;; Sending:  
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439  
;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0  
  
;; QUESTION SECTION:  
;bob.ucsd.edu.          IN A
```

DNS in Action

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:
;bob.ucsd.edu.          IN  A

;; ANSWER SECTION:
bob.ucsd.edu.          3600  IN  A    132.239.80.176

;; AUTHORITY SECTION:
ucsd.edu.              3600  IN  NS    ns0.ucsd.edu.
ucsd.edu.              3600  IN  NS    ns1.ucsd.edu.
ucsd.edu.              3600  IN  NS    ns2.ucsd.edu.

;; ADDITIONAL SECTION:
ns0.ucsd.edu.          3600  IN  A    132.239.1.51
ns0.ucsd.edu.          3600  IN  AAAA  2607:f720:100:100::231
ns1.ucsd.edu.          3600  IN  A    128.54.16.2
ns1.ucsd.edu.          3600  IN  AAAA  2607:f720:300:202::102
ns2.ucsd.edu.          3600  IN  A    132.239.1.52
ns2.ucsd.edu.          3600  IN  AAAA  2607:f720:300:202::52

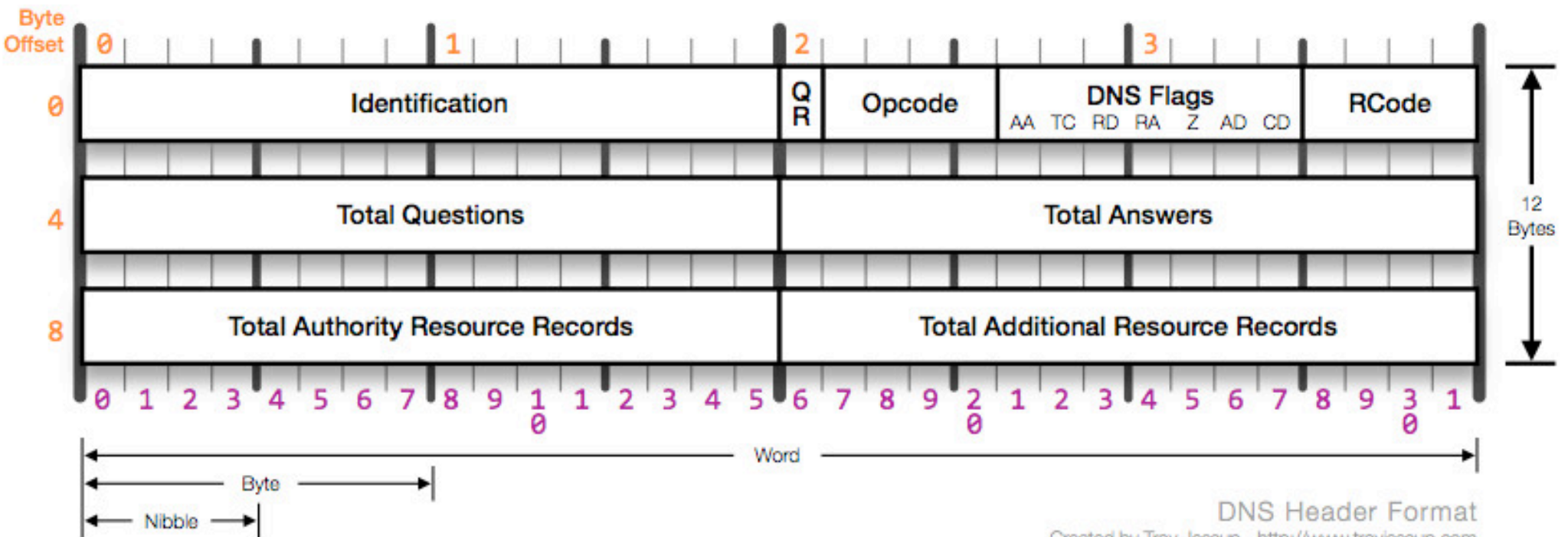
;; Query time: 0 msec
```

DNS

- ❖ DNS uses UDP as transport
- ❖ No authentication of content
- ❖ Attacker can spoof a reply
 - Off-path: need to know port and identifier

DNS Packet

DNS Header



SSL

- ❖ Does SSL protect against ARP poisoning attacks?
- ❖ Does SSL protect against DNS spoofing attacks?

Certificate Semantics

- ❖ Issuer (CA) attests:
 - Public key belongs to subject
C=US, ST=California, L=La Jolla,
O=University of California, San Diego,
OU=ACT Data Center, CN=*.ucsd.edu
 - The domain listed in CN belongs to subject
- ❖ Certificate has expiration and limitations on use

Data:

Version: 3 (0x2)

Serial Number:

0f:77:30:d4:eb:75:d6:c4:22:1e:4b:a1:f6:16:2b:83

Signature Algorithm: sha1WithRSAEncryption

Issuer: C=US, O=DigiCert Inc, OU=www.digicert.com,
CN=DigiCert High Assurance CA-3

Validity

Not Before: Sep 7 00:00:00 2012 GMT

Not After : Nov 11 12:00:00 2015 GMT

Subject: C=US, ST=California, L=La Jolla,
O=University of California, San Diego,
OU=ACT Data Center, CN=*.ucsd.edu

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (2048 bit)

Modulus (2048 bit):

00:cf:73:a9:a0:dd:69:de:98:c5:65:2d:fa:c0:dc:
47:ed:ff:f9:0b:16:3a:ee:e4:74:6a:de:26:37:7b:
ce:f7:de:3e:50:25:13:49:23:ec:c8:b3:19:5f:05:
9e:05:72:41:a9:f7:26:b3:d2:bd:88:37:51:e8:d5:
c3:01:d9:c2:15:bf:eb:87:a3:4b:80:3b:6c:f6:ce:
c5:78:4c:d2:b3:24:af:3d:8b:d8:ba:b9:c9:eb:16:
b4:83:68:06:b6:1e:96:0e:2e:1c:78:91:41:b4:8d:
3c:fe:2a:f5:93:ac:e5:bd:98:78:e5:db:4a:c2:88:
46:3a:1f:1e:07:fd:79:8a:96:c7:e9:b7:05:4d:40:
5d:4d:52:2c:e4:bc:6b:eb:2c:3e:09:e1:27:49:1b:
16:eb:53:cf:d9:df:8f:35:71:b1:10:1f:0b:7f:c1:

SSL/TLS and DNSSEC

- ❖ Only host with private key for bob.ucsd.edu will be able to complete SSL/TLS handshake
 - Certificate ensures that rightful owner of bob.ucsd.edu has key
- ❖ DNSSEC: Signed DNS records
 - How do you authenticate the non-existence of a record?