# Network Applications: DNS; Network App Programming--UDP

Qiao Xiang

https://qiaoxiang.me/courses/cnns-xmuf21/index.shtml

10/05/2021

This deck of slides are heavily based on CPSC 433/533 at Yale University, by courtesy of Dr. Y. Richard Yang.

# Outline

- Admin. and recap
- DNS
  - High-level design
  - Details
  - Extensions/alternatives
- Network app programming
  - UDP

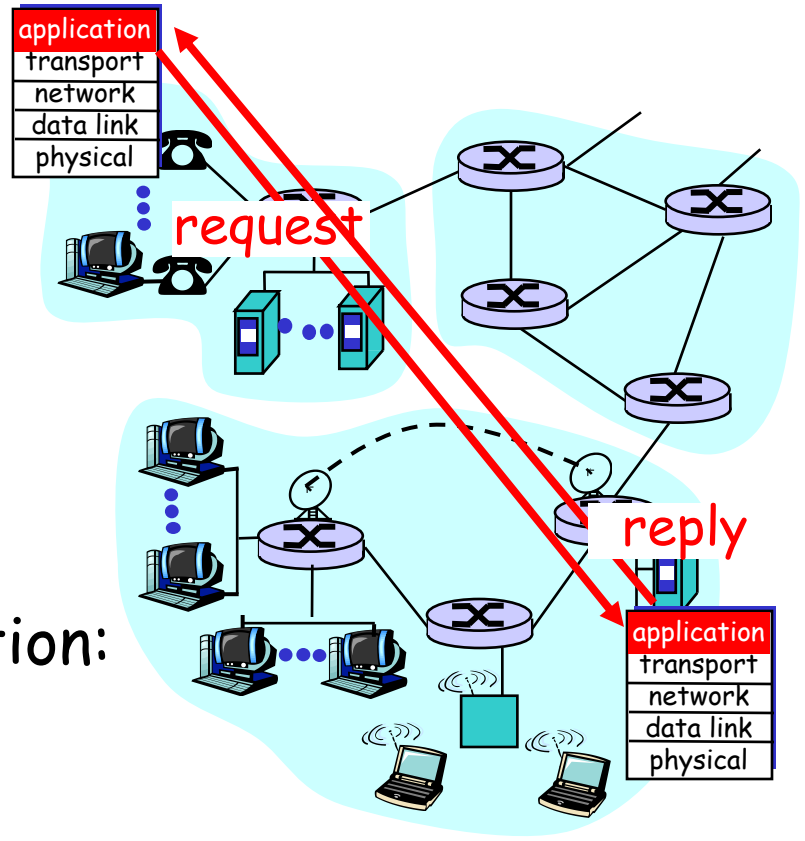
# Admin

- Assignment Two linked on the schedule page
  - Oct. 28, in class or by email to the instructor

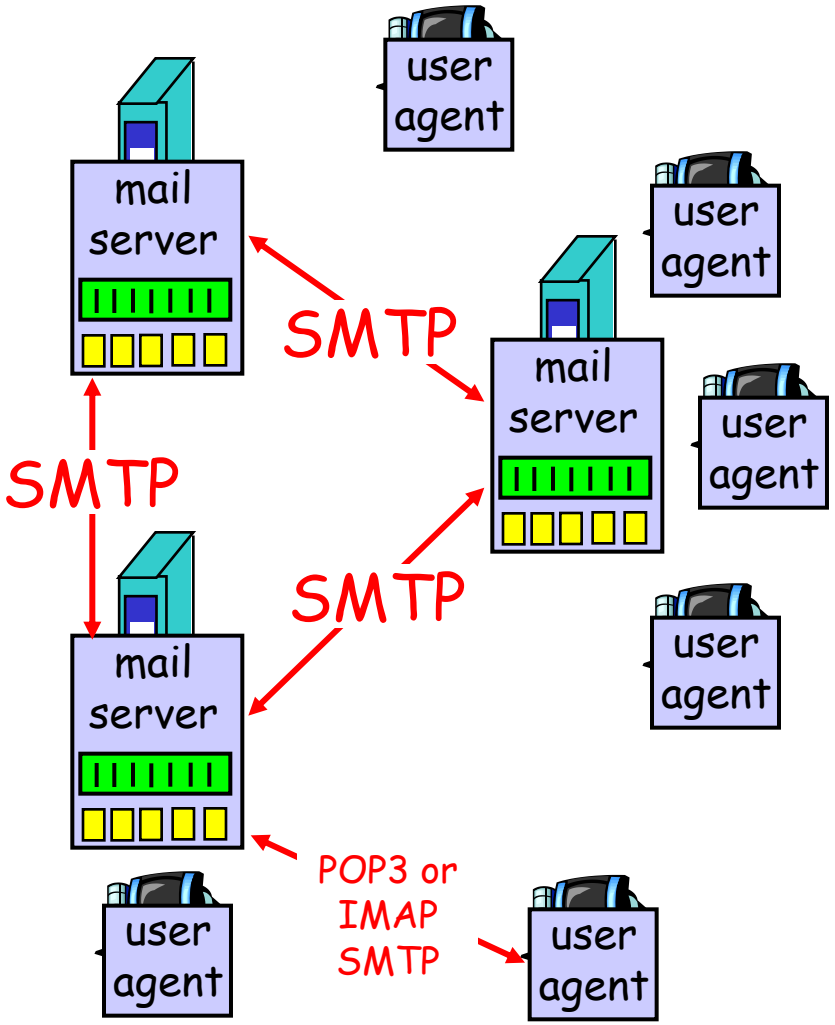
# Recap: Client-Server Paradigm

- The basic paradigm of network applications is the client-server (C-S) paradigm

- Some key design questions to ask about a C-S application:
  - extensibility
  - scalability
  - robustness
  - security



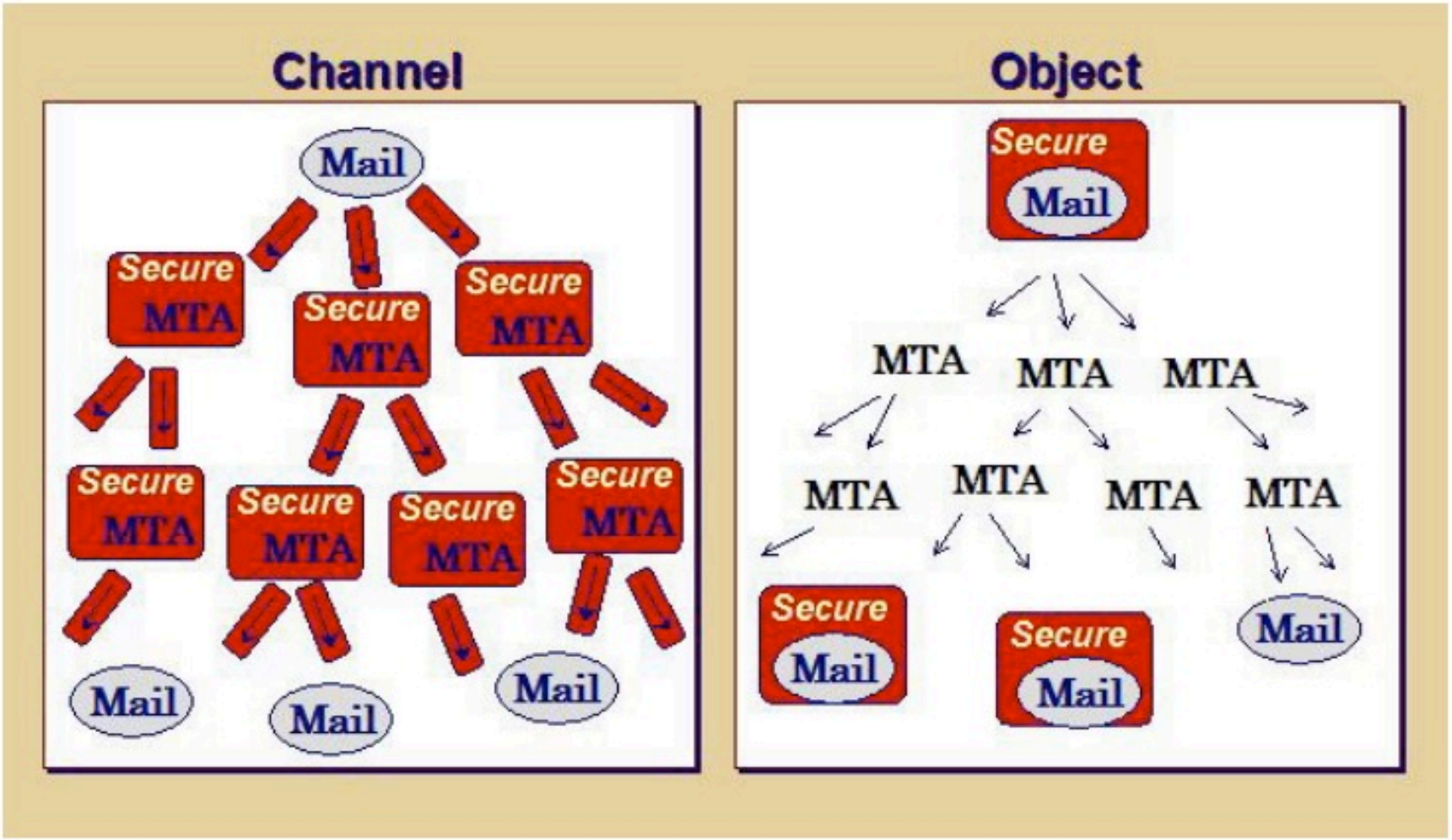
# Recap: Email App




Some key design features of Email

- Separate protocols for different functions
  - email access (e.g., POP3, IMAP)
  - email transport (SMTP)
- Separation of envelop and message body (end-to-end arguments)
  - envelop: simple/basic requests to implement transport control;
  - message body: fine-grain control through ASCII header and message body
    - MIME type as self-describing data type
- Status code in response makes message easy to parse

# Recap: Email Authentication Approaches




Sender Policy Frame (SPF)

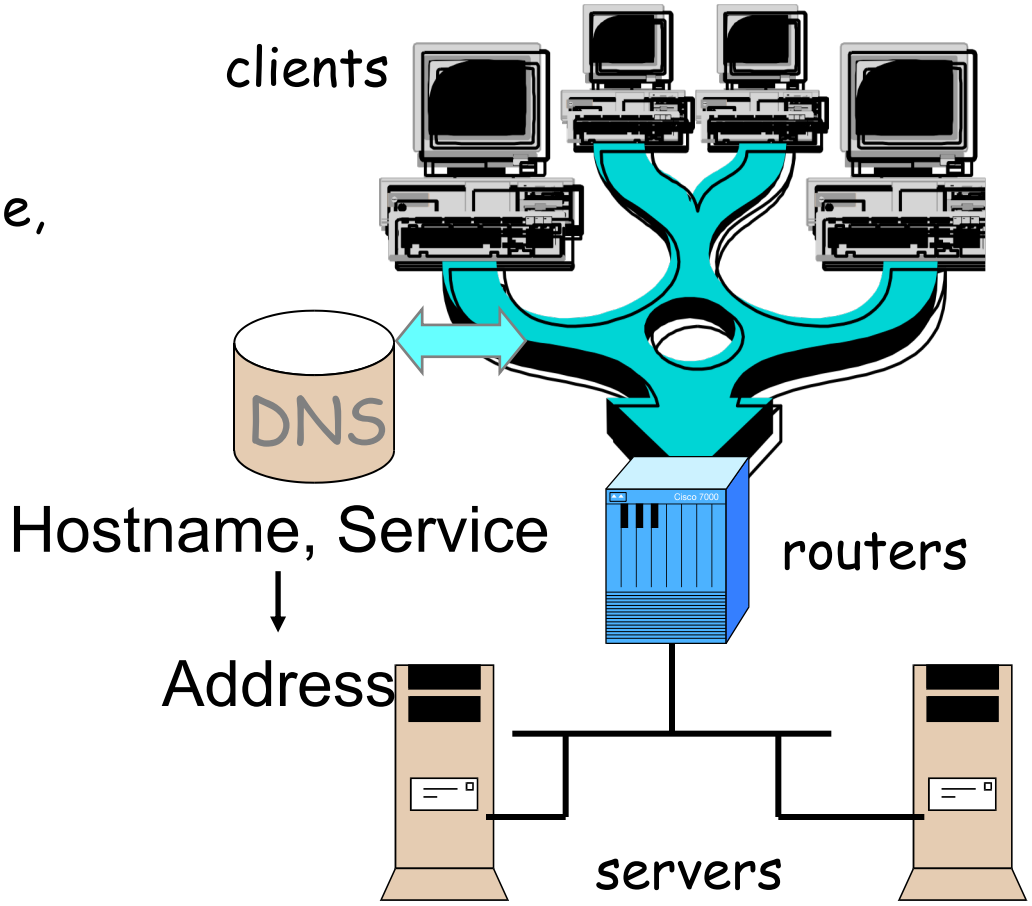
DomainKeys Identified Mail (DKIM)
Authenticated Results Chain (ARC)

# Summary: Some Key Remaining Issues about Email

- Basic: How to find the email server of a domain?
- Scalability/robustness: how to find multiple servers for the email domain?
- Security
  - SPF: How does SPF know if its neighbor MTA is a permitted sender of the domain?
  - DKIM: How does DKIM retrieve the public key of the author domain?

# Recap: Domain Name System (DNS)

- Function
  - map between (domain name, service) to value, e.g.,
    - (xmu.edu.cn, addr) -> 210.34.0.35
    - (xmu.edu.cn, email) -> cmsn1.xmu.edu.cn



# Recap: DNS Records

http://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

DNS: stores resource records (RR)

RR format: **(name, type, value, ttl)**

- Type=A
  - **name** is hostname
  - **value** is IP address
- Type=NS
  - **name** is domain (e.g. xmu.edu.cn)
  - **value** is the name of the authoritative name server for this domain
- Type=TXT
  - general txt
- Type=CNAME
  - **name** is an alias of a "canonical" (real) name
  - **value** is canonical name
- Type=MX
  - **value** is hostname of mail server associated with **name**
- Type=SRV
  - general extension for services
- Type=PTR
  - a pointer to another name

# Recap: Observations

- MX can return multiple servers
- DNS may rotate the servers in answer
- Address can also return multiple addresses
- SPF is encoded as the txt type

# Outline

- Admin. and recap
- DNS
  - High-level design
  - *Details*

# DKIM Example

- Send email from hotmail and check message

# DKIM Example

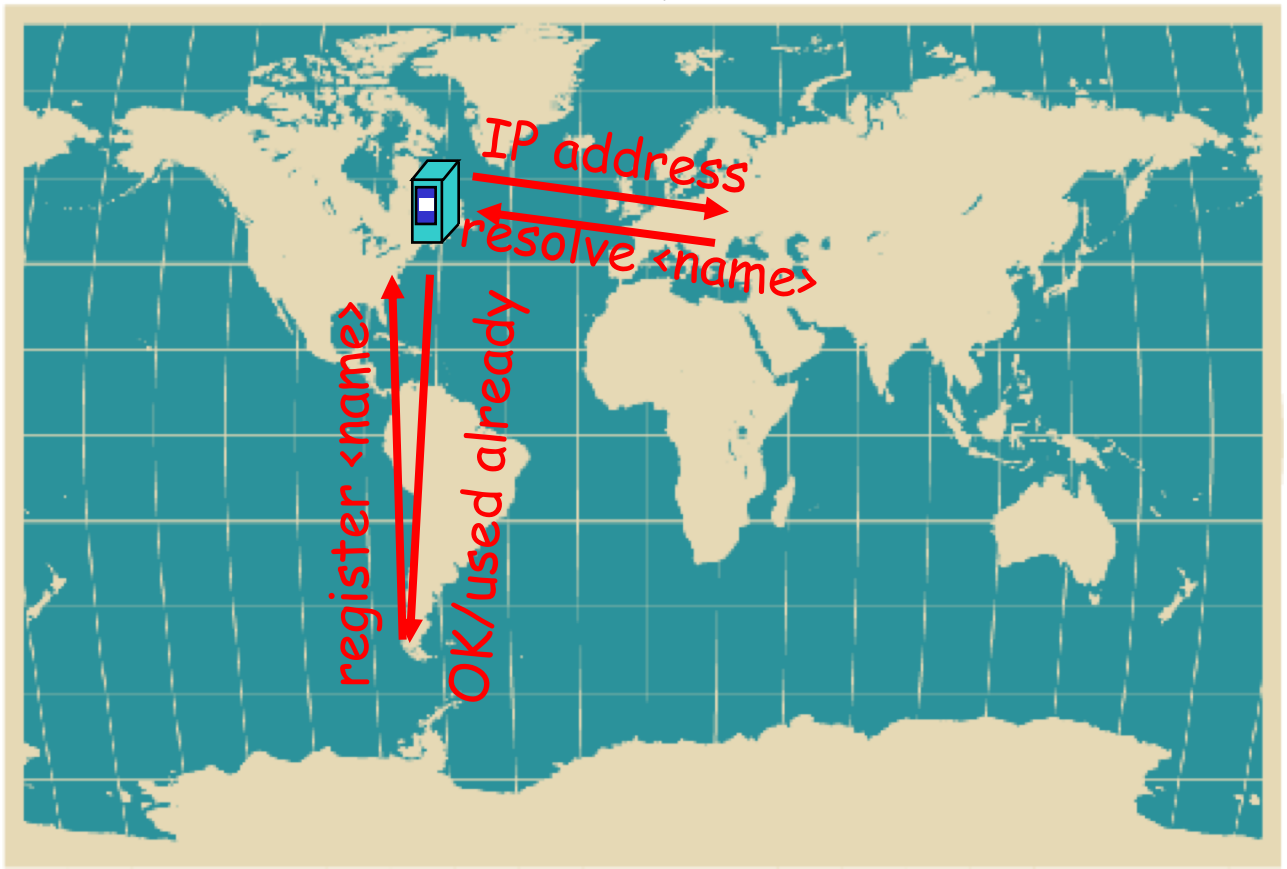
- DKIM / ARC:
  Msg: ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed; d=microsoft.com; s=arcselector9901; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version; bh=bO91TxHI+4MjgAusrfg0EWGiDmvQ5hZRZ/aqb1MKLY8=; …

  DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed; d=hotmail.com; s=selector1; h=From:Date:Subject:Message-ID:Content-Type:MIME-Version:X-MS-Exchange-SenderADCheck;…
- Query: dig arcselector9901._domainkey.microsoft.com txt
- DKIM introduces a session key to allow multiple public keys
  - <session>._domainkey.<domain>

# DNS Design: Dummy Design

- DNS itself can be considered as a client-server system as well
- How about a dummy design: introducing one super Internet DNS server?

THE DNS server of the Internet

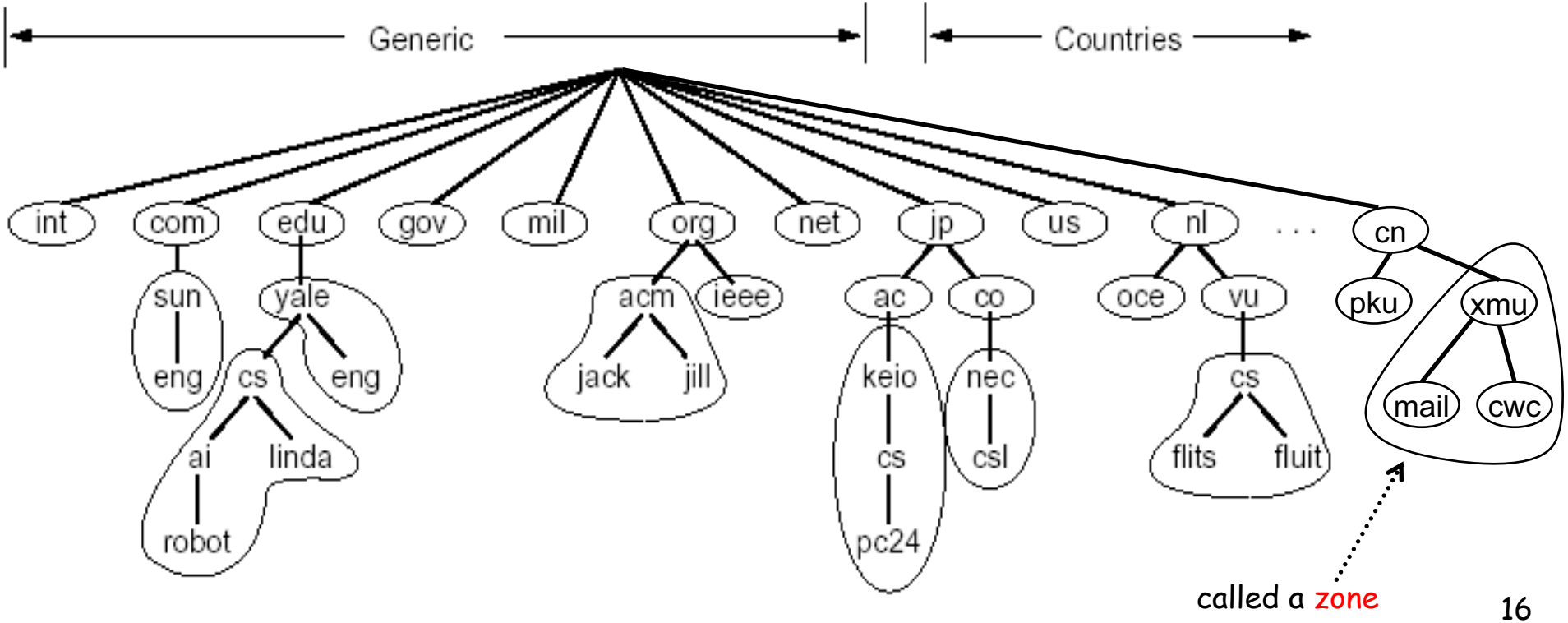


# Problems of a Single DNS Server

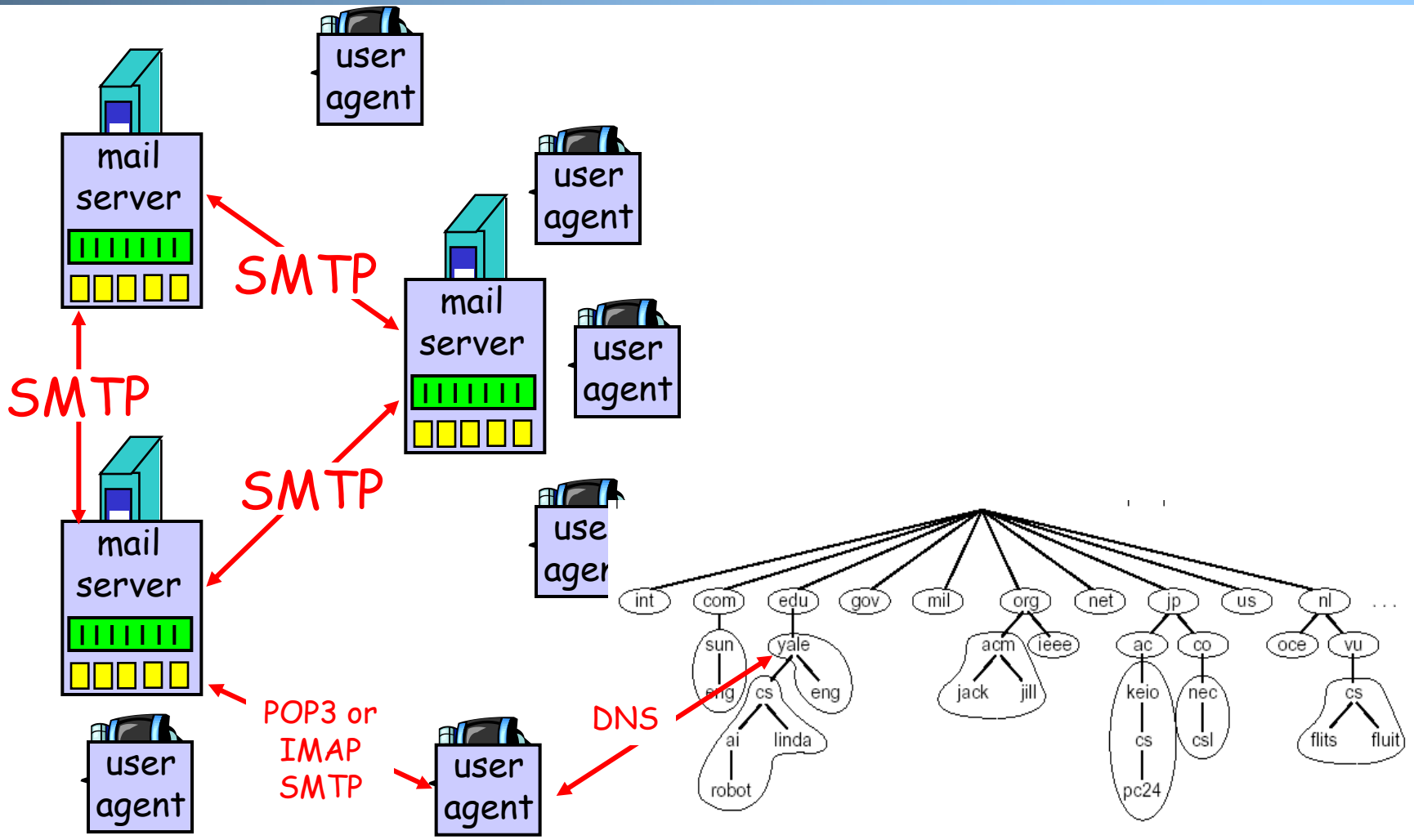
- Scalability and robustness bottleneck
- Administrative bottleneck

# DNS: Distributed Management of the Domain Name Space

- A distributed database managed by authoritative name servers
  - divided into zones, where each zone is a sub-tree of the global tree
  - each zone has its own **authoritative name servers**
  - an authoritative name server of a zone may delegate a subset (i.e. a sub-tree) of its zone to another name server



# Email Architecture + DNS



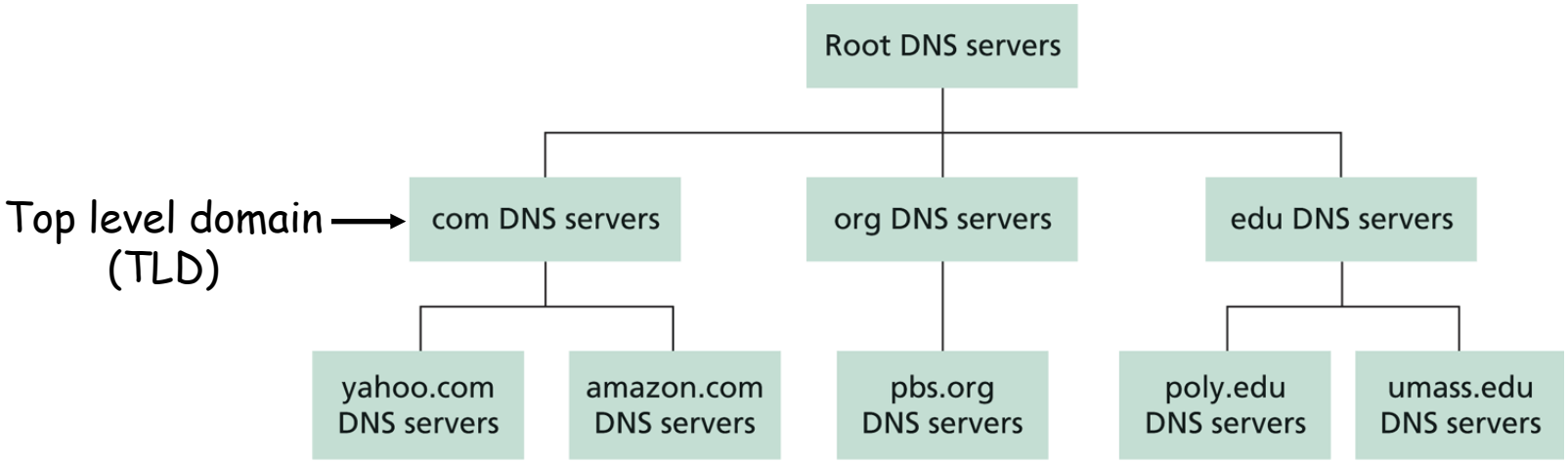
# Root Zone and Root Servers

- The root zone is managed by the root name servers
  - 13 root name servers worldwide


See http://root-servers.org/ for more details

# Linking the Name Servers

- Each name server knows the addresses of the root servers
- Each name server knows the addresses of its immediate children (i.e., those it delegates)




Q: how to query a hierarchy?

# DNS Message Flow: Two Types of Queries

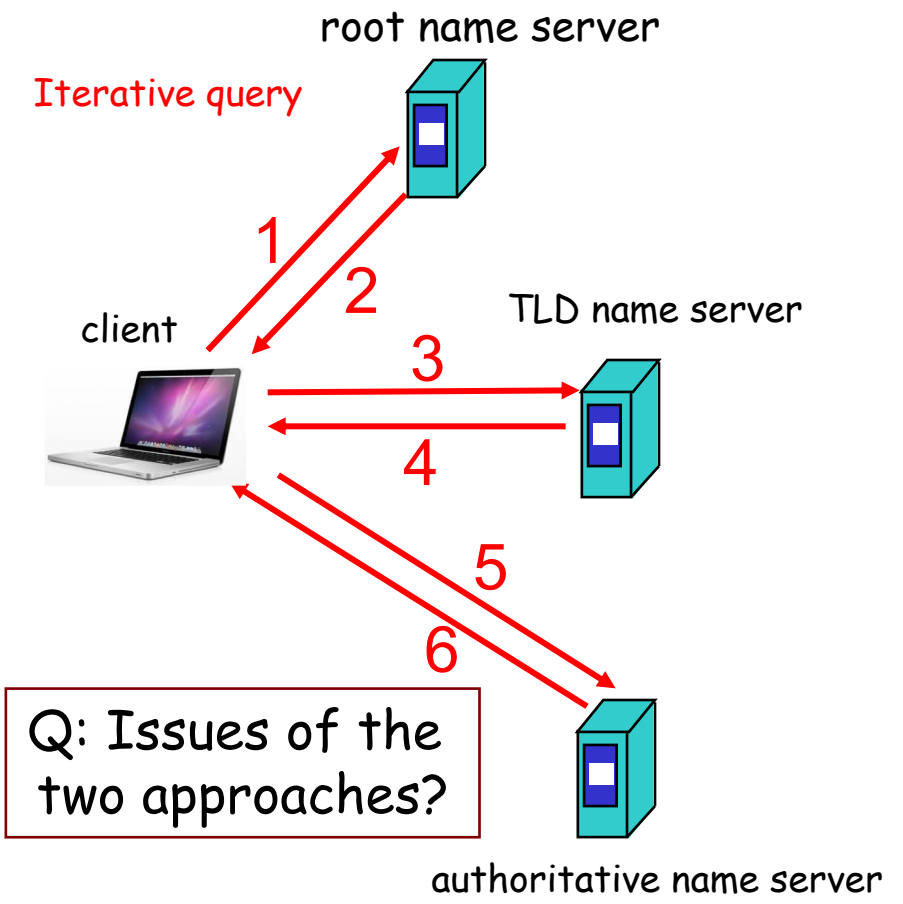
## Recursive query:

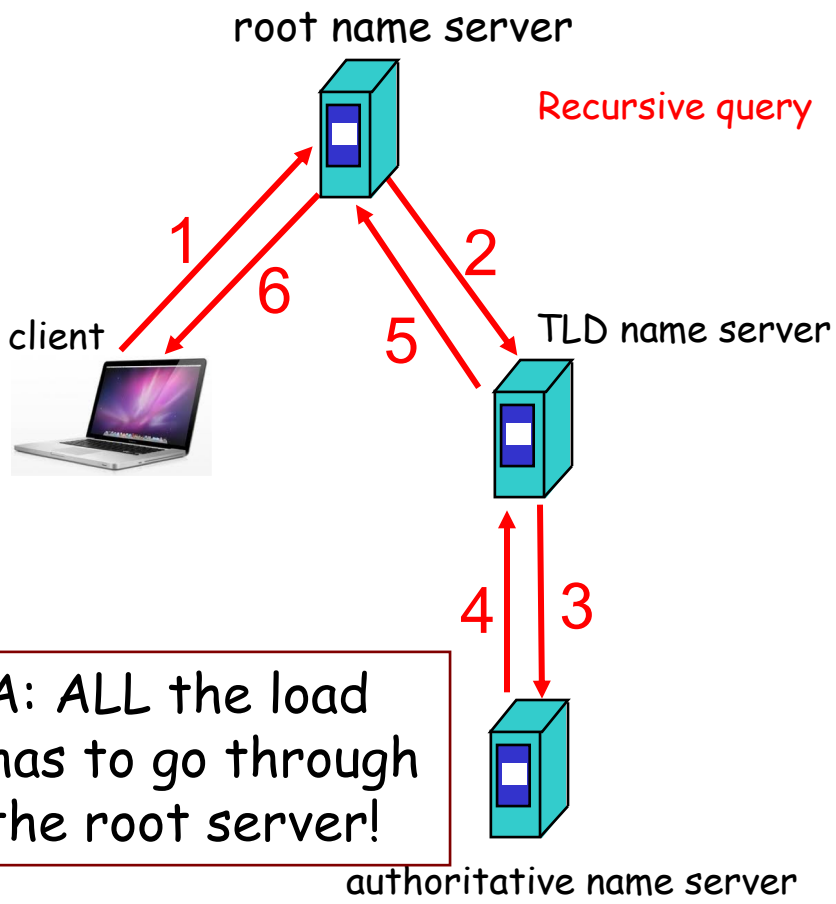
- The contacted name server resolves the name completely

## Iterated query:

- Contacted server replies with name of server to contact
  - "I don't know this name, but ask this server"

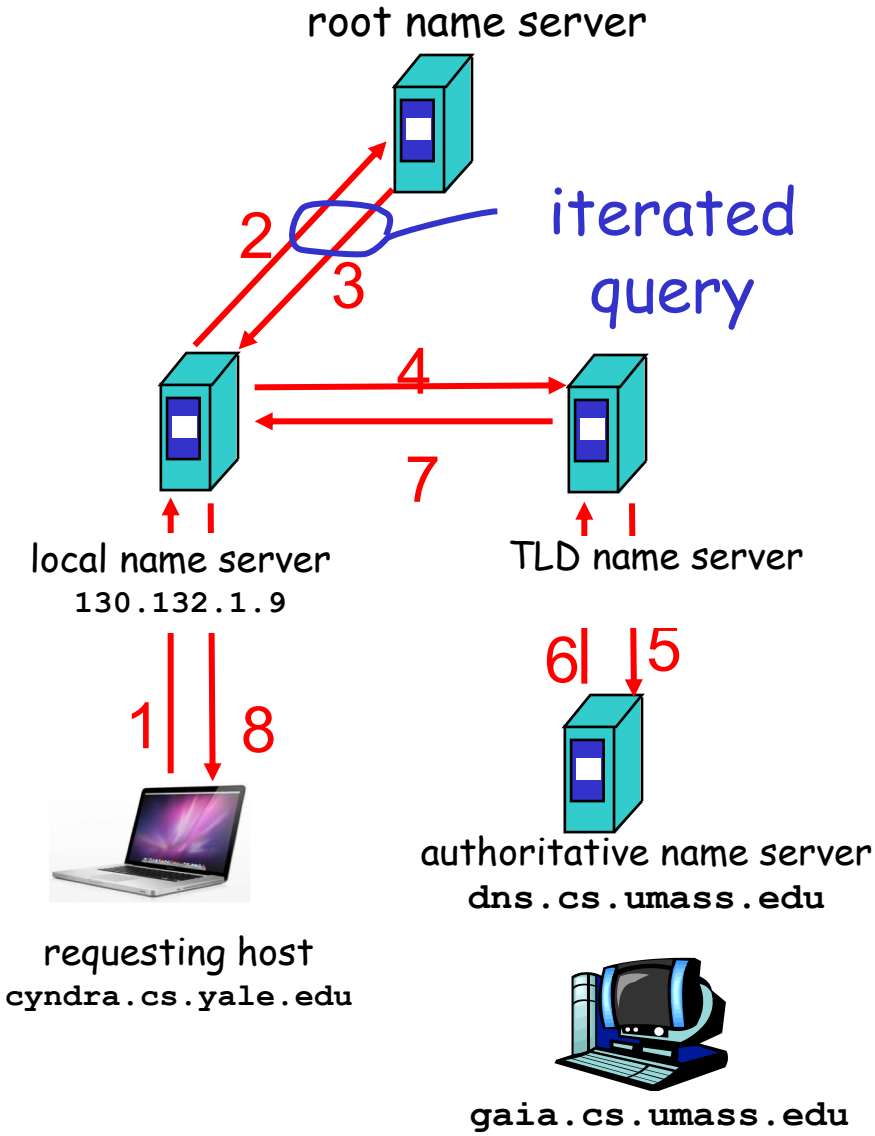
# Two Extreme DNS Message Flows



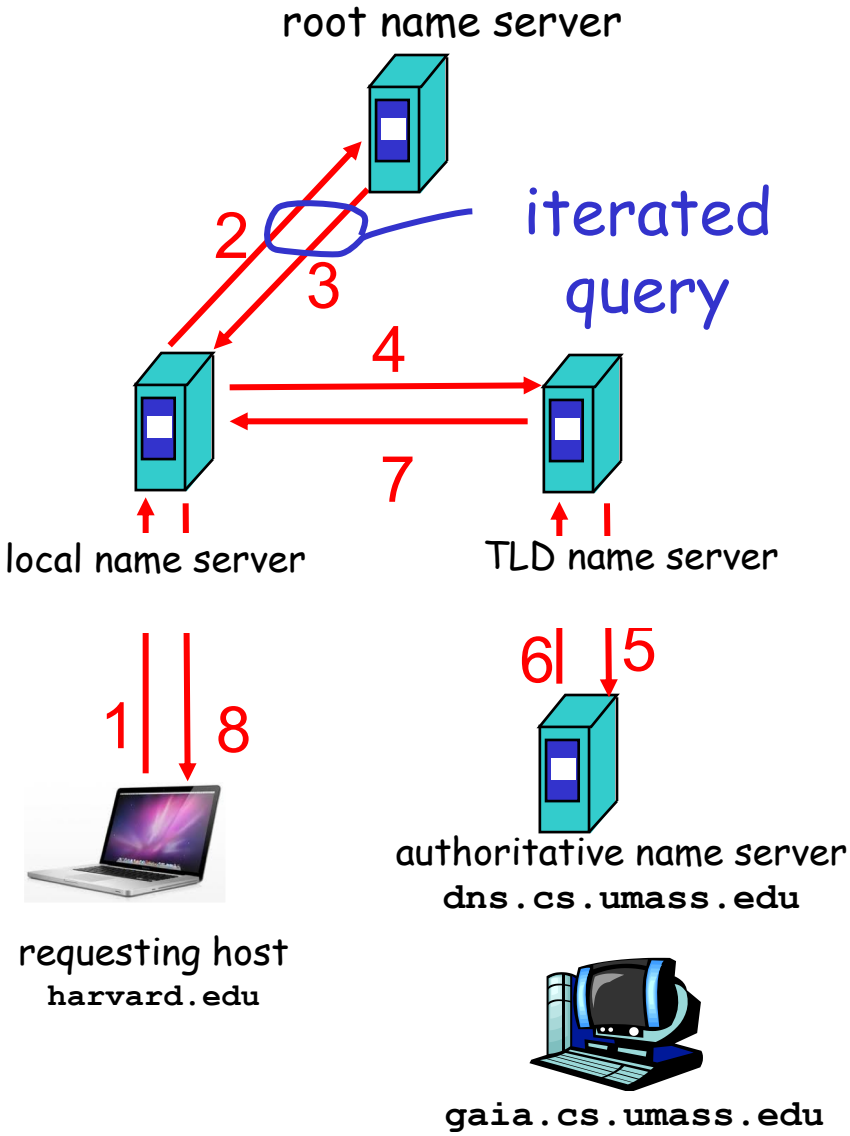
# Typical DNS Message Flow: The Hybrid Case

- Host knows only local name server
- Local name server is learned from DHCP, or configured, e.g. /etc/resolv.conf
- Local DNS server helps clients resolve DNS names



# Typical DNS Message Flow: The Hybrid Case

- Host knows only local name server
- Local name server is learned from DHCP, or configured, e.g. /etc/resolv.conf
- Local DNS server helps clients resolve DNS names
- Benefits of local name servers (often called **resolvers**)
  - simplifies client
  - caches/reuses results



# Outline

- Admin. and recap
- *DNS*
  - High-level design
  - *Details*

# DNS Message Format?

Basic encoding decisions: UDP/TCP, how to encode domain name, how to encode answers…

# Observing DNS Messages

- Capture the messages
  - DNS server is at port 53
    - Display and clear DNS cache
      - MacOS: https://support.apple.com/en-us/HT202516
        sudo killall -HUP mDNSResponder
      - Ubuntu:

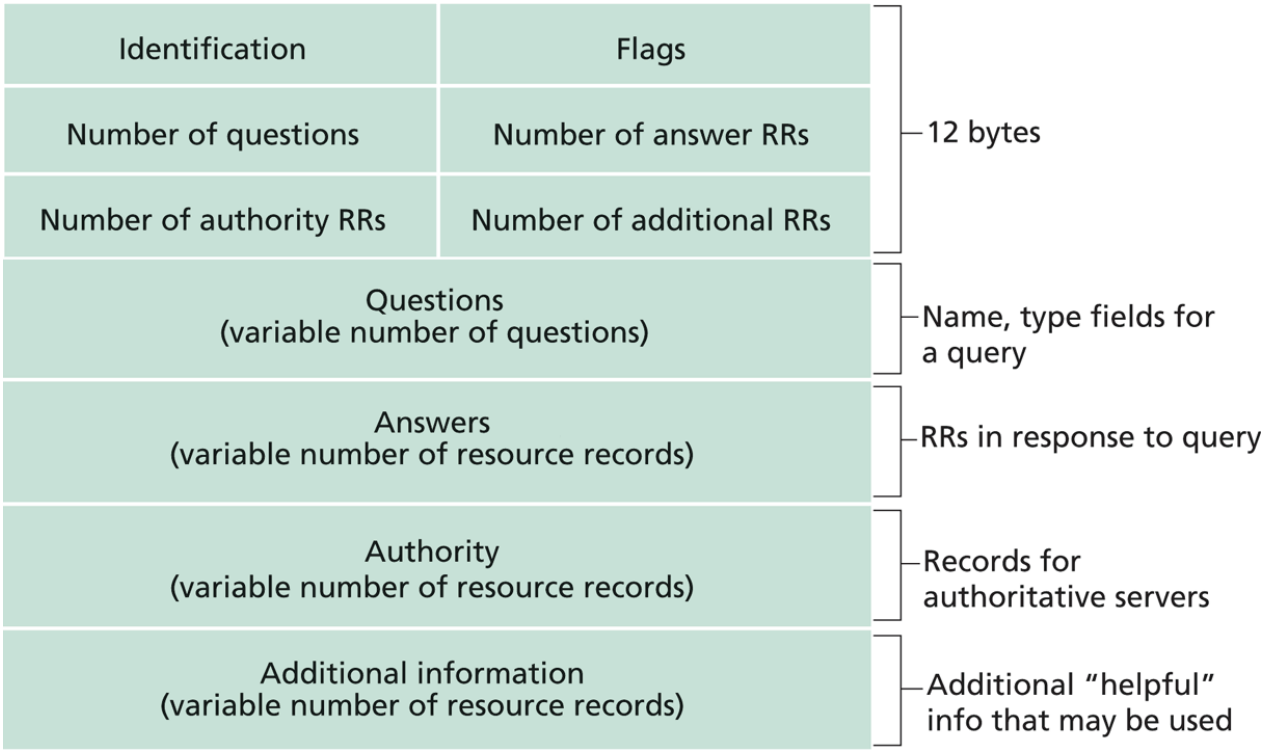
        sudo systemd-resolve --flush-caches

        sudo systemd-resolve --statistics
  - Try to load the dns-capture file from class Schedule page, if you do not want live capture

https://www.ietf.org/rfc/rfc1035.txt

# DNS Protocol, Messages

DNS protocol : typically over UDP (can use TCP); *query* and *reply* messages, both with the same *message format*

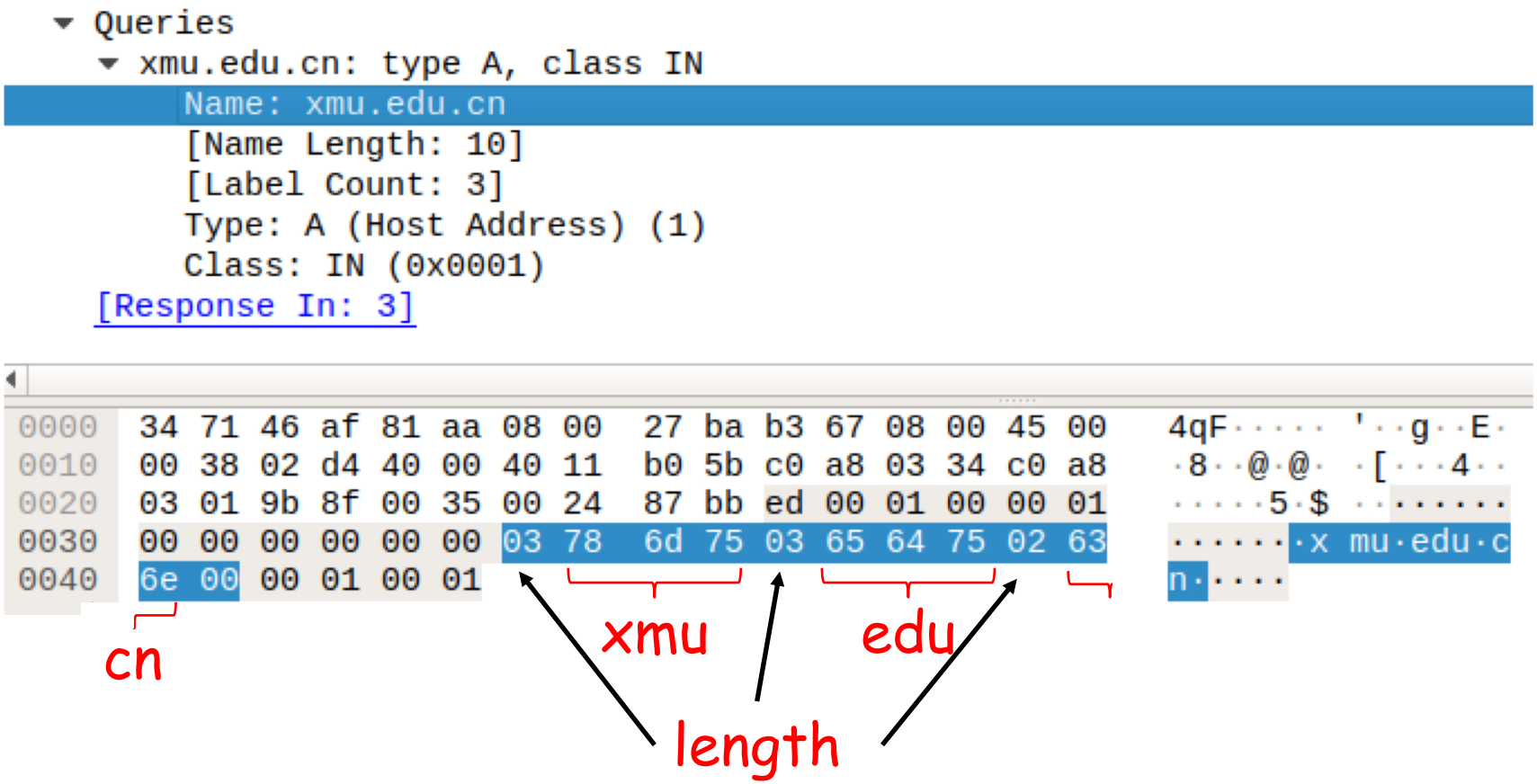


# DNS Details

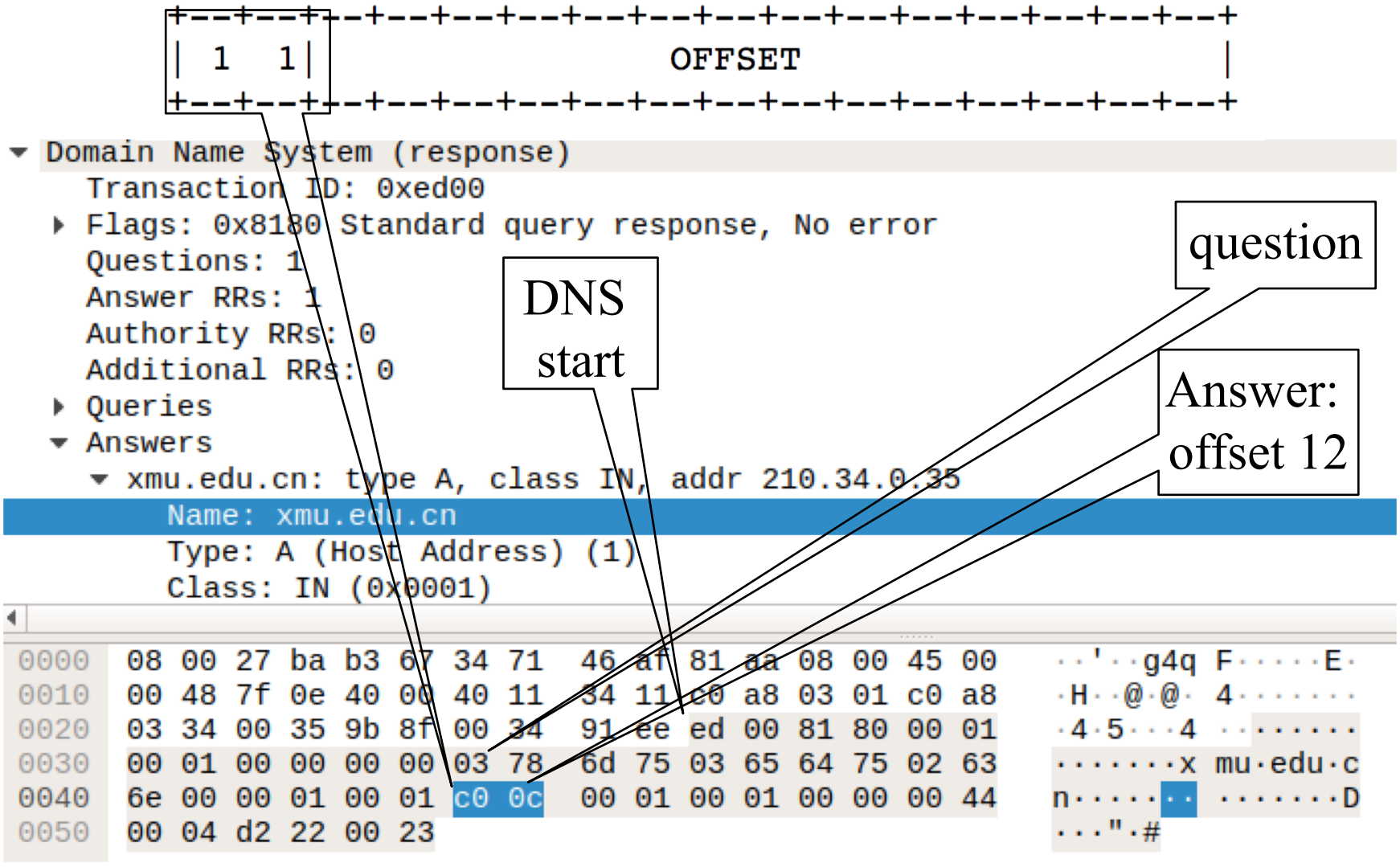
- Header (Sec. 4.1.1 of https://www.ietf.org/rfc/rfc1035.txt)
- Encoding of questions (Sec. 4.1.2):
  - [Label-length label-chars]
- Encoding of answers (Sec. 4.1.3)
  - Pointer format (http://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml)

- See example DNS packets

# Name Encoding



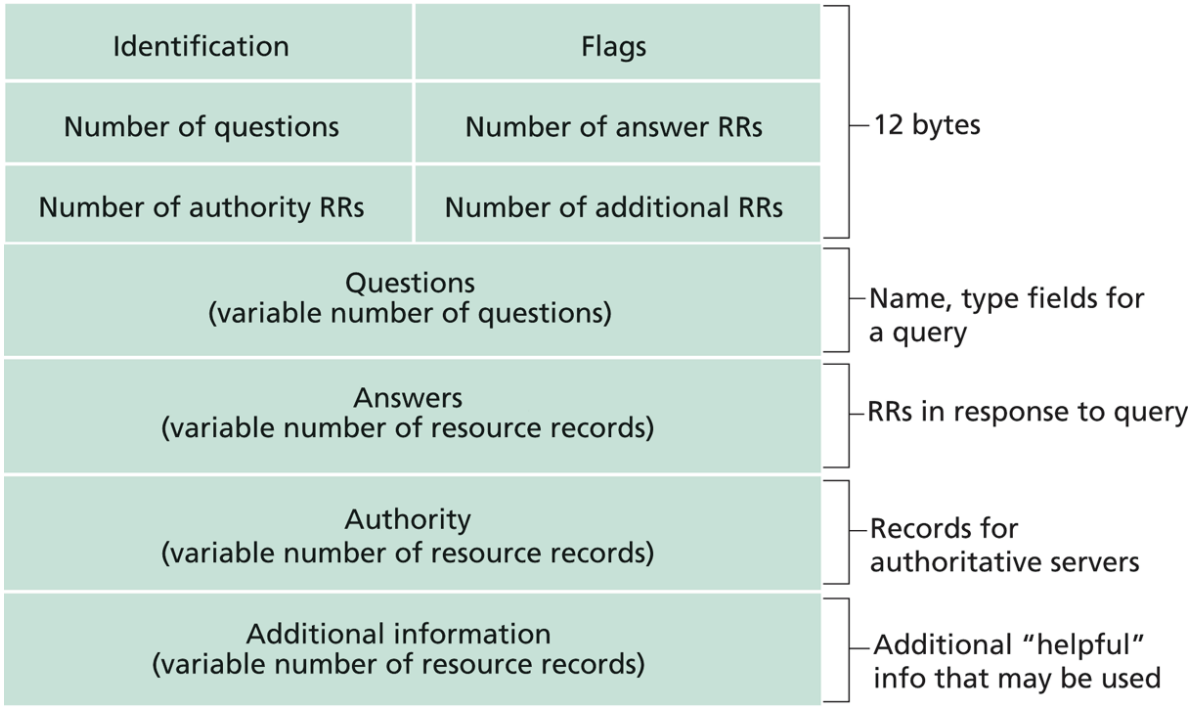
# Message Compression (Label Pointer)



https://www.ietf.org/rfc/rfc1035.txt

# Recap: DNS Protocol, Messages

Many features: typically over UDP (can use TCP); *query* and *reply* messages with the same *message format; length/content encoding of names; simple compression; additional info as server push*



# What DNS did Right?

- Hierarchical delegation avoids central control, improving manageability and scalability
- Redundant servers improve robustness
  - see http://www.internetnews.com/dev-news/article.php/1486981 for DDoS attack on root servers in Oct. 2002 (9 of the 13 root servers were crippled, but only slowed the network)
- Caching reduces workload and improves robustness
- Proactive answers reduce # queries on server and latency on client

# Problems of DNS

- Simple query model, relatively static resource values and types make it harder to implement generic service discovery
  - e.g., service discovery of all printers
  - Although theoretically you can update the values of the records, it is rarely enabled
- Early binding (separation of DNS query from application query) does not work well in mobile, dynamic environments
  - e.g., load balancing, locate the nearest printer
- Each local domain needs servers, but an ad hoc domain may not have a DNS server

# Outline

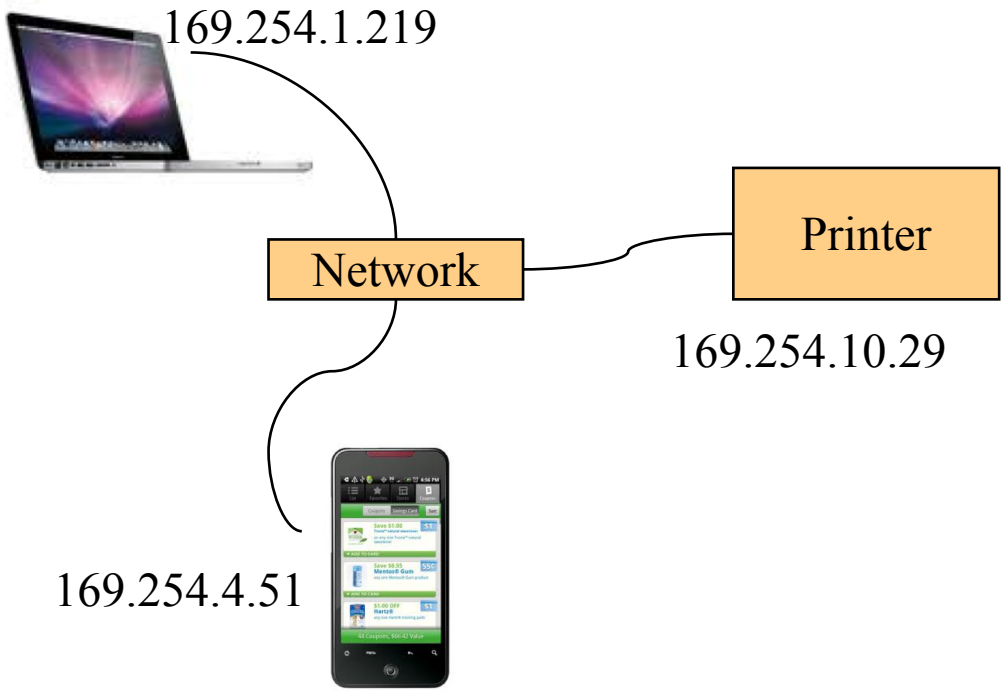
- Admin. and recap
- DNS
  - High-level design
  - Details
  - *Extensions/alternatives*

# Discussions

- What extension(s) to standard DNS operations do we need to allow service discovery, say to implement Bonjour (discover all local printers)?
  - each printer needs to provide the following info: host, port, printer info (e.g., support postscript)

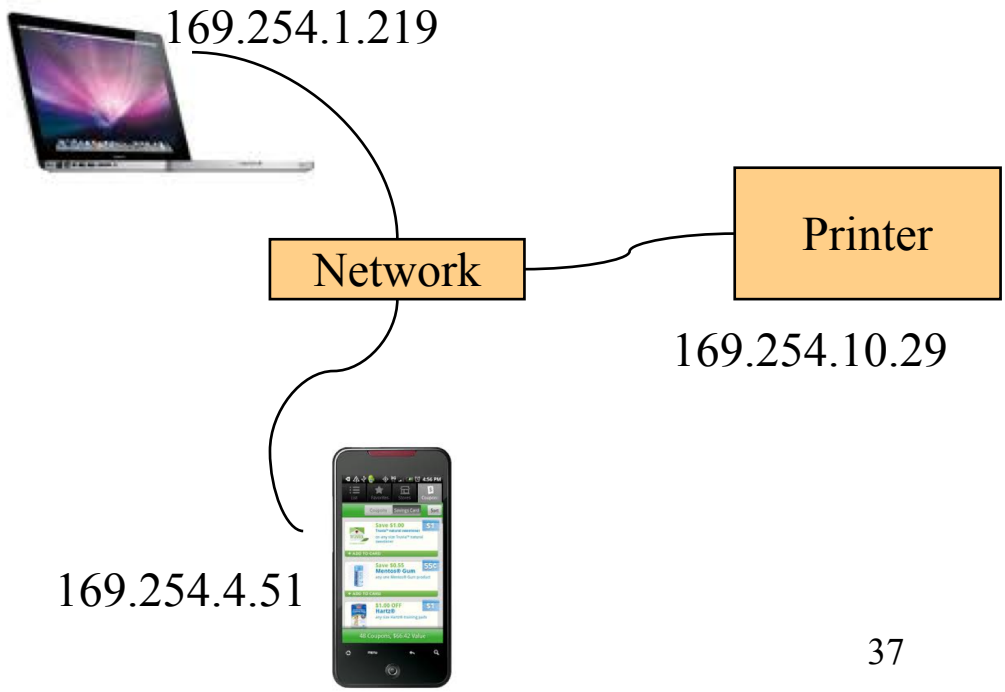
# DNS-Service Discovery

- Leverage DNS message format, but each node can announce its own services



# Realizing DNS-SD without Central DNS Server: mDNS

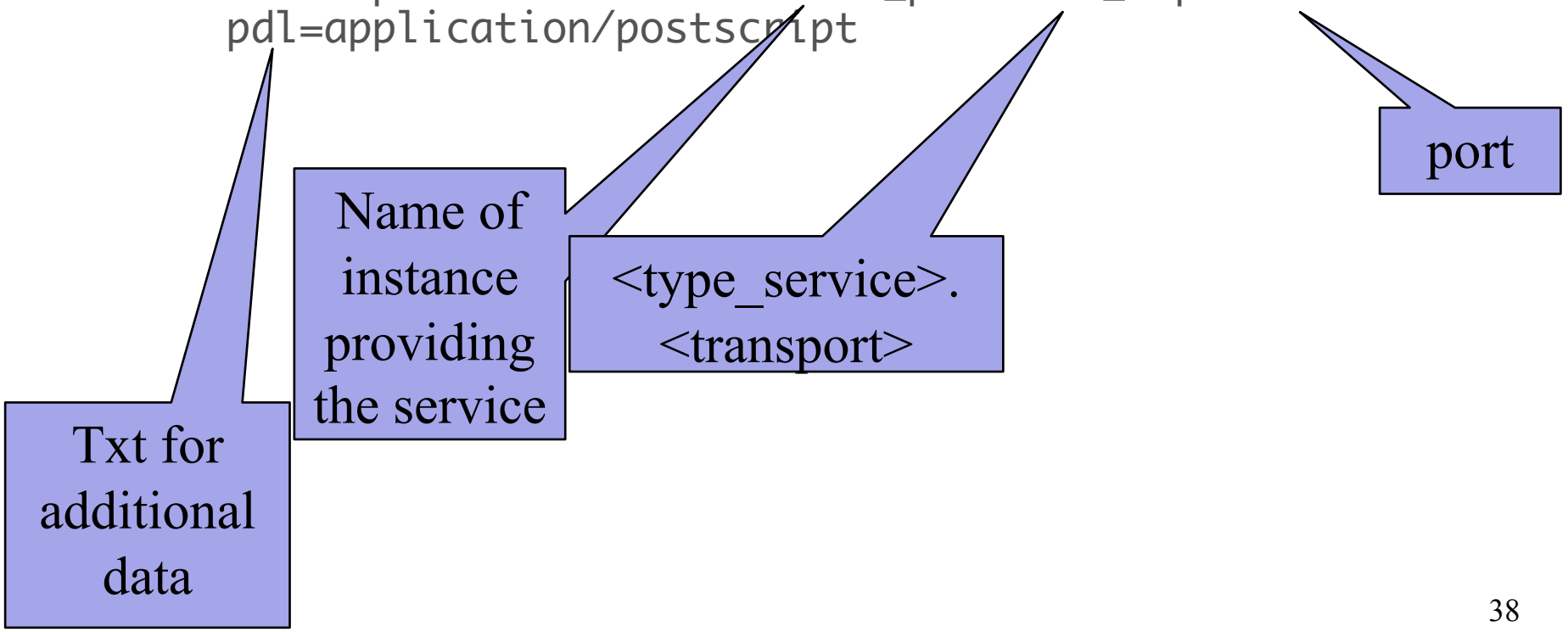
- Multicast in a small world
  - no central address server
    - each node is a responder
  - link-local addressing
    - send to multicast address: 224.0.0.251



# Example

- Use the avahi-publish-service command on Ubuntu as example
  - Advertise (register) an LPR printer on port 515

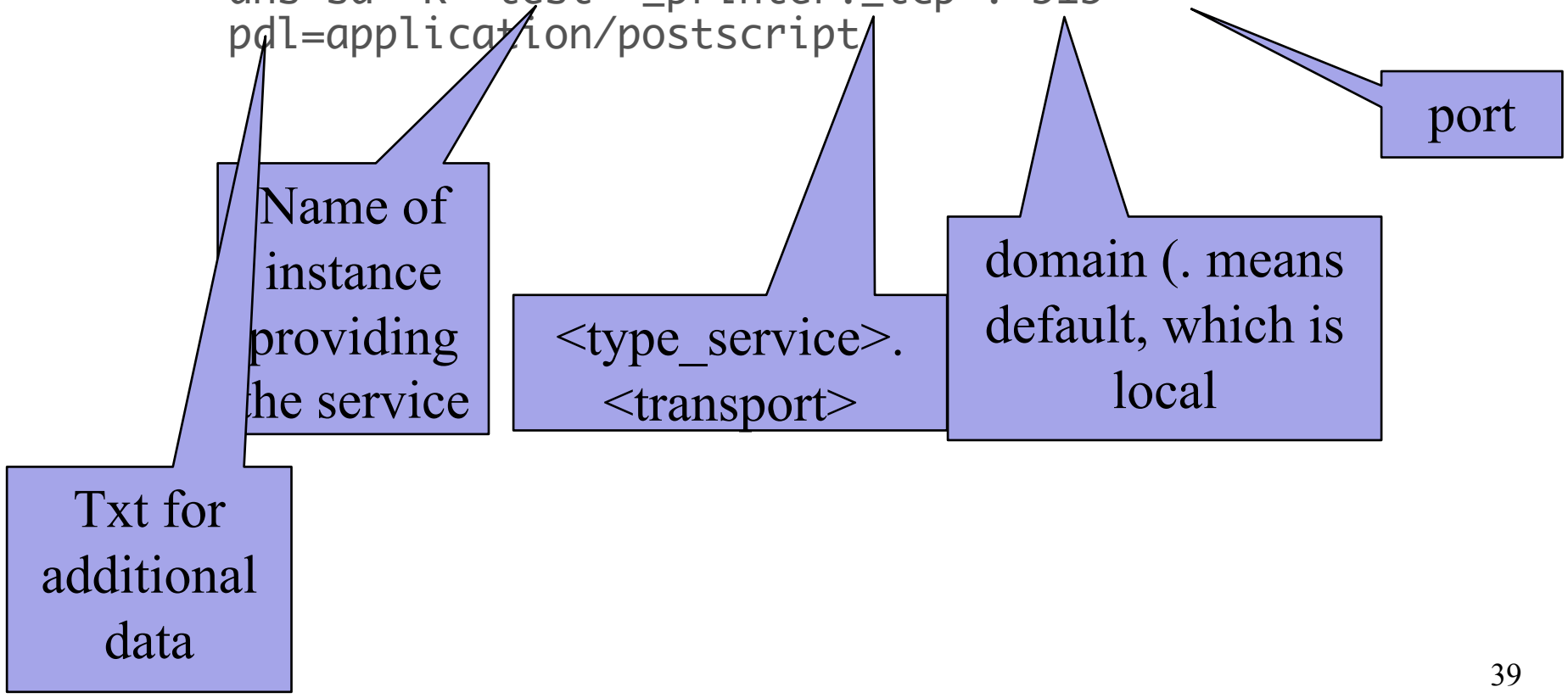
```
avahi-publish-service test _printer._tcp . 515
pdl=application/postscript
```

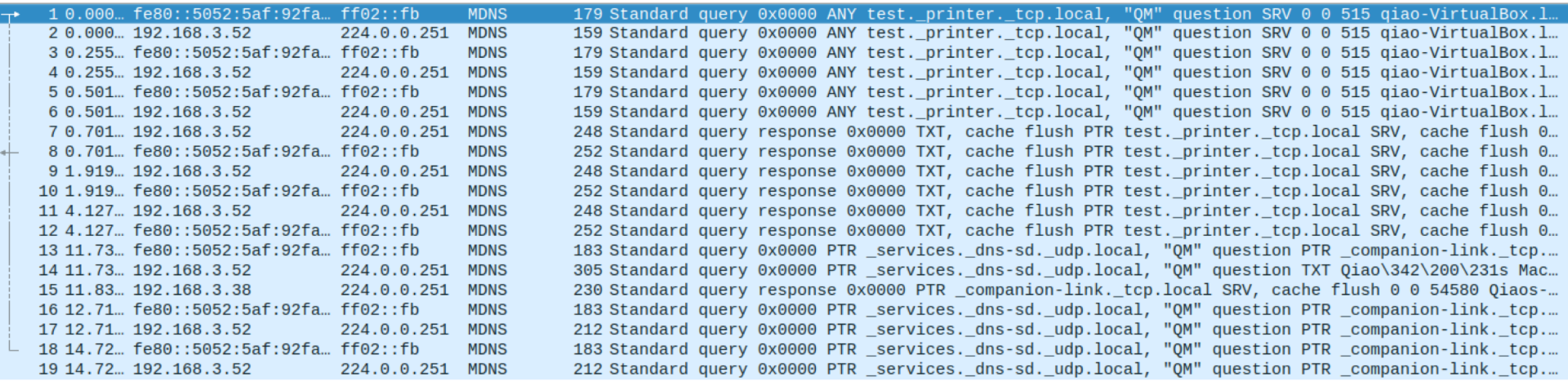


# Example

- Use the dns-sd command on Mac as example
  - Advertise (register) an LPR printer on port 515

```
dns-sd -R "test" _printer._tcp . 515
pdl=application/postscript
```





| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0.000… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 179 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 2 | 0.000… | 192.168.3.52 | 224.0.0.251 | MDNS | 159 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 3 | 0.255… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 179 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 4 | 0.255… | 192.168.3.52 | 224.0.0.251 | MDNS | 159 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 5 | 0.501… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 179 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 6 | 0.501… | 192.168.3.52 | 224.0.0.251 | MDNS | 159 | Standard query 0x0000 ANY test._printer._tcp.local, "QM" question SRV 0 0 515 qiao-VirtualBox.l… |
| 7 | 0.701… | 192.168.3.52 | 224.0.0.251 | MDNS | 248 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 8 | 0.701… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 252 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 9 | 1.919… | 192.168.3.52 | 224.0.0.251 | MDNS | 248 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 10 | 1.919… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 252 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 11 | 4.127… | 192.168.3.52 | 224.0.0.251 | MDNS | 248 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 12 | 4.127… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 252 | Standard query response 0x0000 TXT, cache flush PTR test._printer._tcp.local SRV, cache flush 0… |
| 13 | 11.73… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 183 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question PTR _companion-link._tcp.… |
| 14 | 11.73… | 192.168.3.52 | 224.0.0.251 | MDNS | 305 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question TXT Qiao\342\200\231s Mac… |
| 15 | 11.83… | 192.168.3.38 | 224.0.0.251 | MDNS | 230 | Standard query response 0x0000 PTR _companion-link._tcp.local SRV, cache flush 0 0 54580 Qiaos-… |
| 16 | 12.71… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 183 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question PTR _companion-link._tcp.… |
| 17 | 12.71… | 192.168.3.52 | 224.0.0.251 | MDNS | 212 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question PTR _companion-link._tcp.… |
| 18 | 14.72… | fe80::5052:5af:92fa… | ff02::fb | MDNS | 183 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question PTR _companion-link._tcp.… |
| 19 | 14.72… | 192.168.3.52 | 224.0.0.251 | MDNS | 212 | Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question PTR _companion-link._tcp.… |

# Offline Exercise

- Use the dns-sd /avahi-publish-service command as example
  - Advertise (register) a web page on local machine

```
dns-sd -R "My Test" _http._tcp . 80
path=/path-to-page.html
```

# Issue: How to Query

- Query needs a back pointer, PTR records
- Exercise: Use the dns-sd / avahi-service-publish command as example

    - Browse web pages on local machines

```
dns-sd -B _http._tcp
avahi-browse –rt _http._tcp
```

# Network Service Discovery in Android

- Based on DNS-SD/mDNS
- Foundation for peer-to-peer/Wi-Fi Direct in Android

- See https://developer.android.com/training/connect-devices-wirelessly/nsd.html for programming using nsd

# Backup Slides (General Service/Naming Discovery)

# General Service/Naming Discovery Paradigm: Linda

- "Distributed workspace" by David Gelernter in the 80's at Yale
- Very influential in naming and resource discovery
- Key issues
  - How to name services/resources
  - How to resolve names

# The Linda Paradigm

- Naming scheme:
  - arbitrary tuples (heterogeneous-type vectors)

- Name resolution:
  - Nodes write into shared memory
  - Nodes read matching tuples from shared memory
    - exact matching is required for extraction

# Linda: Core API

- out(): writes tuples to shared space
  - example: *out("abc", 1.5, 12).*
  - result: insert ("abc", 1.5, 12) into space

- read(): retrieves tuple copy matching arg list (blocking)
  - example: *read("abc", ? A, ? B)*
  - result: finds ("abc", 1.5, 12) and sets local variables A = 1.5, B = 12. Tuple ("abc", 1.5, 12) is still resident in space.

- in(): retrieves and deletes matching tuple from space (blocking)
  - example: same as above except ("abc", 1.5, 12) is deleted

- eval(expression): similar to out except that the tuple argument to eval is evaluated
  - example: eval("ab",-6,abs(-6)) creates tuple ("ab", -6, 6)

# Linda Extension: JavaSpaces

- Industry took Linda principles and made modifications
  - add transactions, leases, events
  - store Java objects instead of tuples
  - a very comprehensive service discovery system

- Definitive book, "JavaSpaces Principles, Patterns, and Practice"