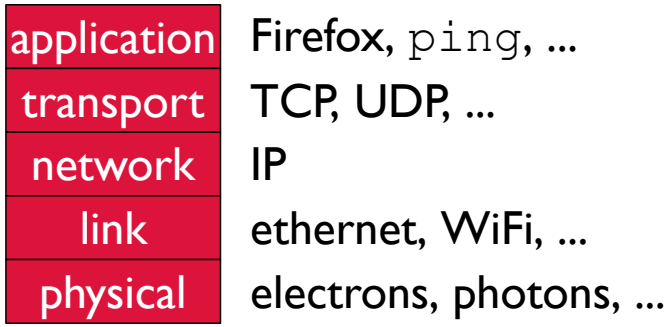
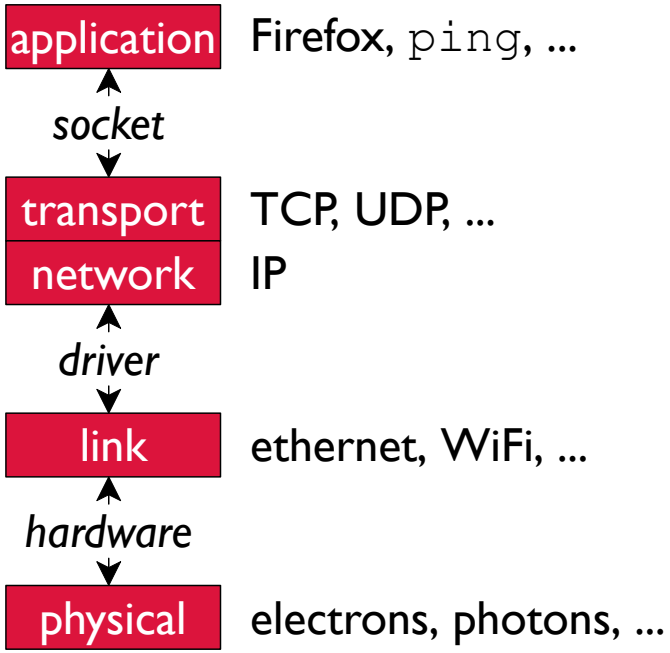
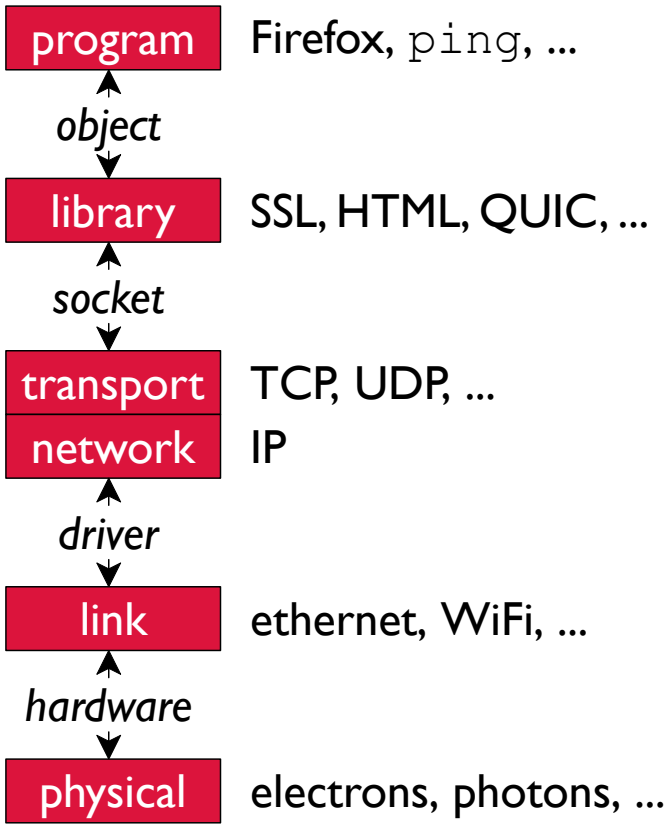
Network Layers



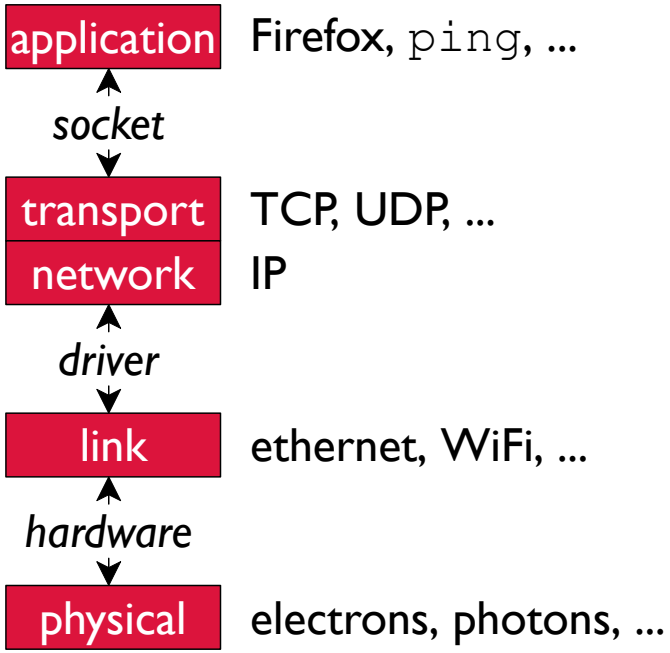
Network Layers



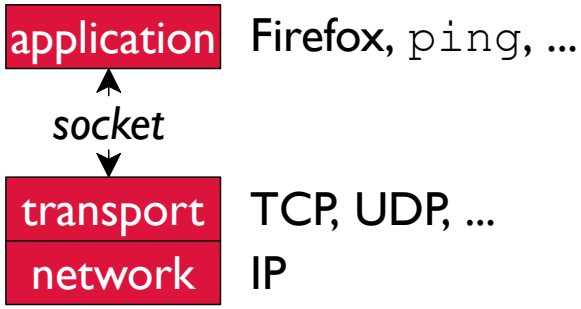
Network Layers



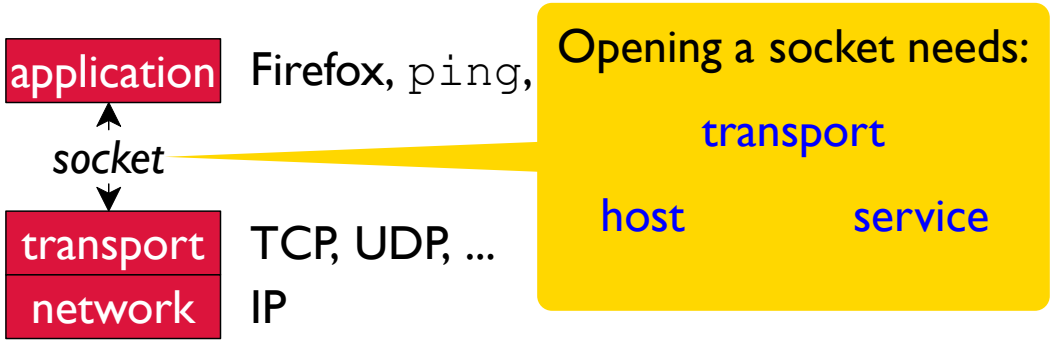
Network Layers



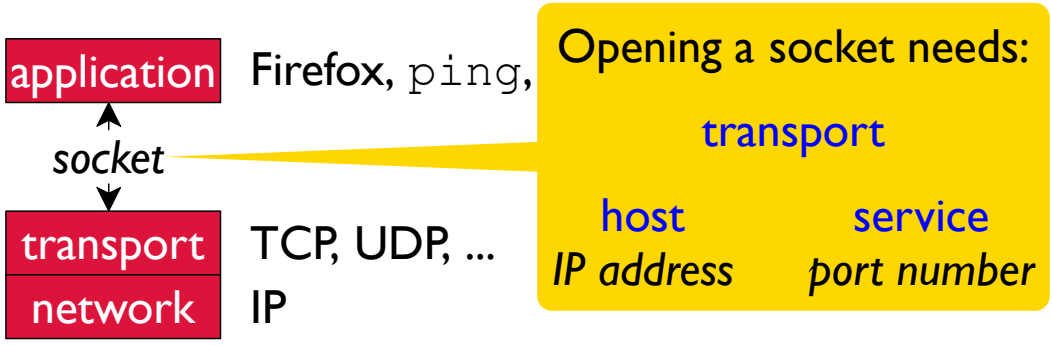
Network Layers



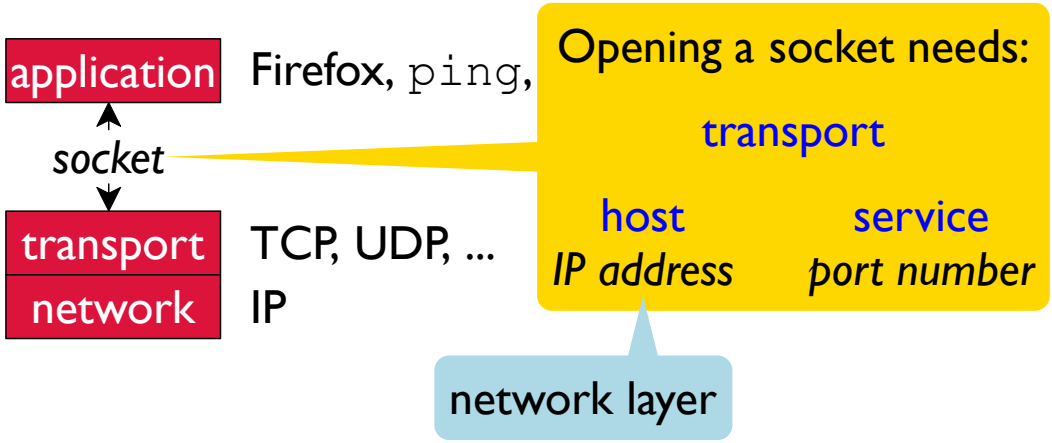
Network Layers



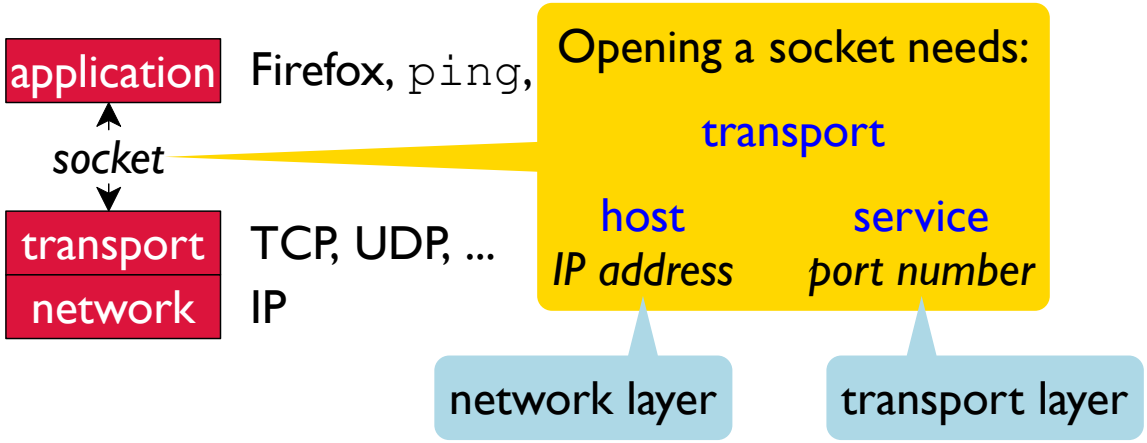
Network Layers



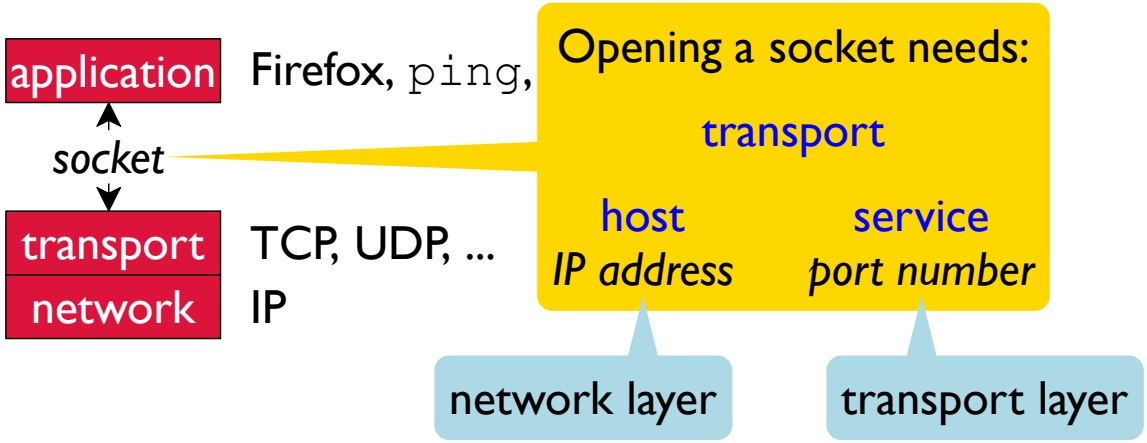
Network Layers



Network Layers

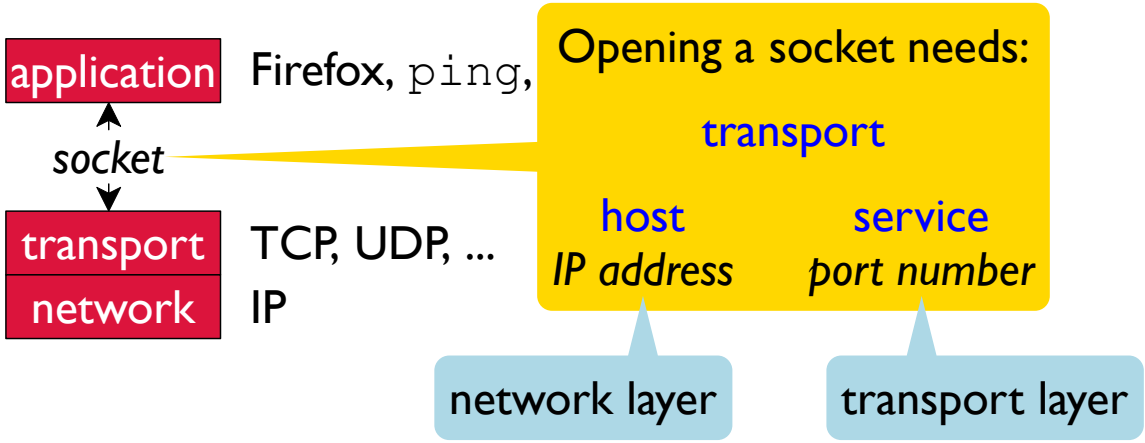


Network Layers



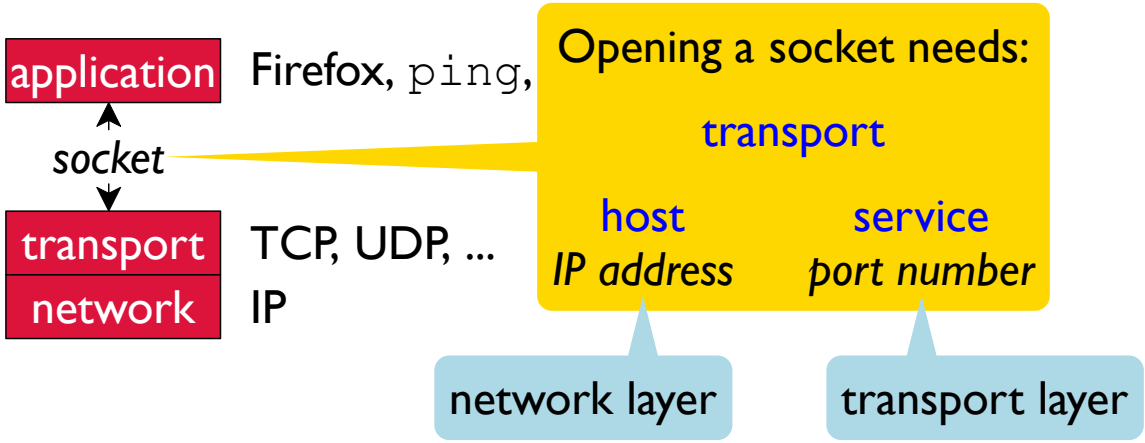
Web server TCP `www.cs.utah.edu` `80`

Network Layers



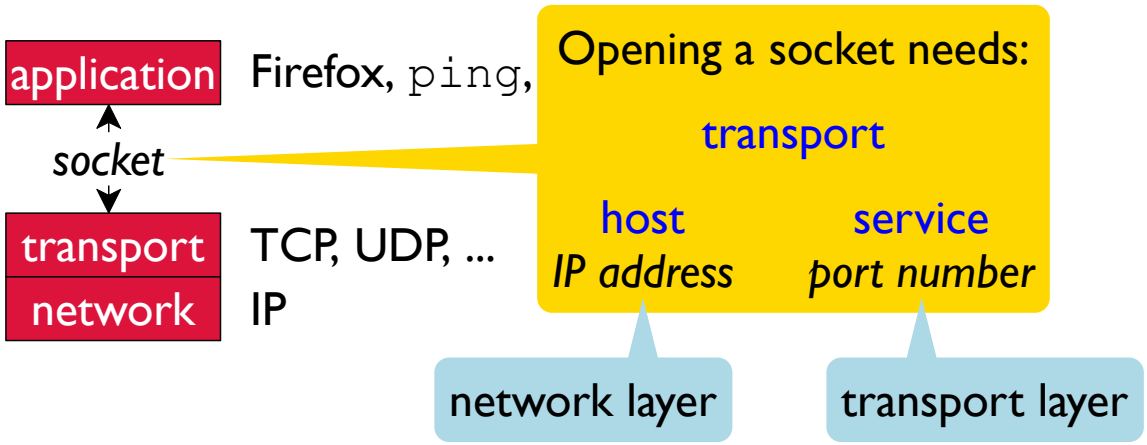
Web server	TCP	`www.cs.utah.edu`	`80`
Modern web server	TCP	`www.cs.utah.edu`	`443`

Network Layers



Web server	TCP	`www.cs.utah.edu`	`80`
Modern web server	TCP	`www.cs.utah.edu`	`443`
Mail receiver	TCP	`smtp.cs.utah.edu`	`25`

Network Layers



Web server	TCP	`www.cs.utah.edu`	`80`
Modern web server	TCP	`www.cs.utah.edu`	`443`
Mail receiver	TCP	`smtp.cs.utah.edu`	`25`
Name resolver	UDP	`8.8.8.8`	`53`

NAME

socket – create an endpoint for communication

SYNOPSIS

```
#include <sys/socket.h>

int
socket(int domain, int type, int protocol);
```

DESCRIPTION

socket() creates an endpoint for communication and returns a descriptor.

The domain parameter specifies a communications domain within which communication will take place; this selects the protocol family which should be used. These families are defined in the include file ⟨sys/socket.h⟩. The currently understood formats are

PF_LOCAL	Host-internal protocols, formerly called PF_UNIX,
PF_UNIX	Host-internal protocols, deprecated, use PF_LOCAL,
PF_INET	Internet version 4 protocols,
PF_ROUTE	Internal Routing protocol,
PF_KEY	Internal key-management function,
PF_INET6	Internet version 6 protocols,
PF_SYSTEM	System domain,
PF_NDRV	Raw access to network device,
PF_VSOCK	VM Sockets protocols

The socket has the indicated type, which specifies the semantics of communication. Currently defined types are:

SOCK_STREAM
SOCK_DGRAM
SOCK_RAW

A SOCK_STREAM type provides sequenced, reliable, two-way connection based byte streams. An out-of-band data transmission mechanism may be supported. A SOCK_DGRAM socket supports datagrams (connectionless, unreliable messages of a fixed (typically small) maximum length). SOCK_RAW sockets provide access to internal network protocols and interfaces. The type SOCK_RAW, which is available only to the super-user.

CONNECT(2) System Calls Manual CONNECT(2)

NAME

connect – initiate a connection on a socket

SYNOPSIS

```
#include <sys/types.h>
#include <sys/socket.h>

int
connect(int socket, const struct sockaddr *address, socklen_t address_len);
```

DESCRIPTION

The parameter socket is a socket. If it is of type SOCK_DGRAM, this call specifies the peer with which the socket is to be associated; this address is that to which datagrams are to be sent, and the only address from which datagrams are to be received. If the socket is of type SOCK_STREAM, this call attempts to make a connection to another socket. The other socket is specified by address, which is an address in the communications space of the socket.

Each communications space interprets the address parameter in its own way. Generally, stream sockets may successfully connect() only once; datagram sockets may use connect() multiple times to change their association. Datagram sockets may dissolve the association by calling disconnectx(2), or by connecting to an invalid address, such as a null address or an address with the address family set to AF_UNSPEC (the error EAFNOSUPPORT will be harmlessly returned).

RETURN VALUES

Upon successful completion, a value of 0 is returned. Otherwise, a value of -1 is returned and the global integer variable errno is set to indicate the error.

ERRORS

The connect() system call will fail if:

[EACCES] The destination address is a broadcast address and the socket option SO_BROADCAST is not set.

[EADDRINUSE] The address is already in use.

[EADDRNOTAVAIL] The specified address is not available on this machine.

[EAFNOSUPPORT] Addresses in the specified address family cannot be used with this socket.

Transport Layer Options

TCP **UDP**

Transport Layer Options

TCP

stream

UDP

datagram

Transport Layer Options

TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery

Transport Layer Options

TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery
reliable order	unreliable order

Transport Layer Options

TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery
reliable order	unreliable order
flow/congestion control	no flow/congestion control

Transport Layer Options

TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery
reliable order	unreliable order
flow/congestion control	no flow/congestion control
potentially slow	potentially fast

Transport Layer Options

When you need the right answer and can wait a little while

TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery
reliable order	unreliable order
flow/congestion control	no flow/congestion control
potentially slow	potentially fast

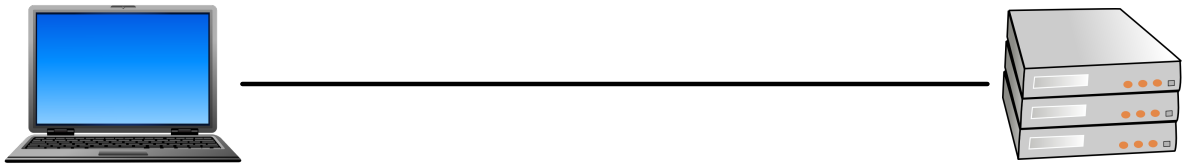
Transport Layer Options

When you need the right answer and can wait a little while

For small data, or for getting most data as fast as possible

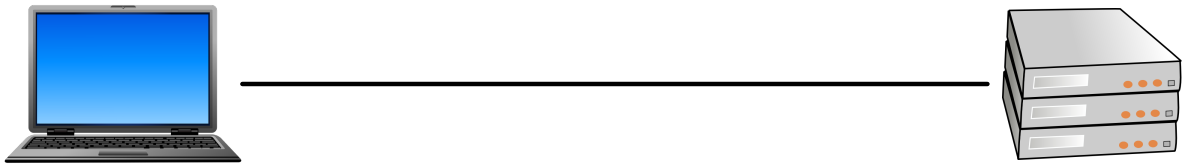
TCP	**UDP**
stream	*datagram*
reliable delivery	unreliable delivery
reliable order	unreliable order
flow/congestion control	no flow/congestion control
potentially slow	potentially fast

Example: Web Browsing



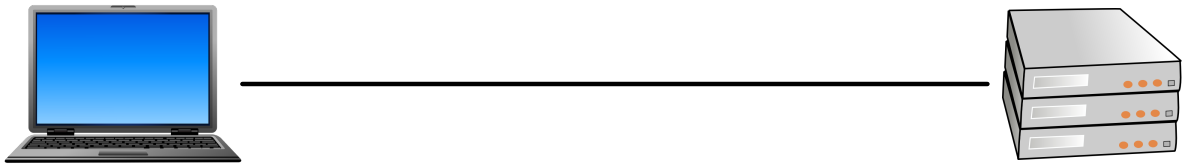
Example: Web Browsing

client–server

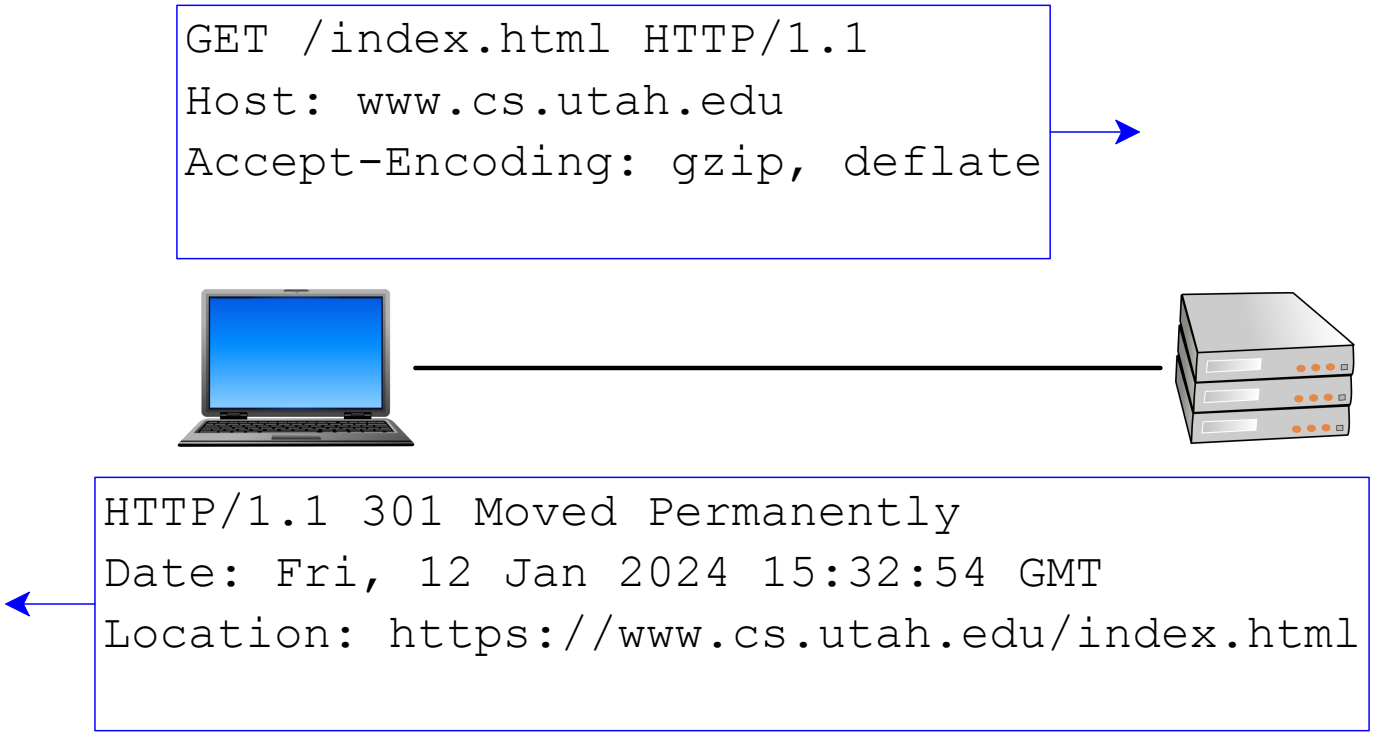


Example: Web Browsing

```
GET /index.html HTTP/1.1
Host: www.cs.utah.edu
Accept-Encoding: gzip, deflate
```



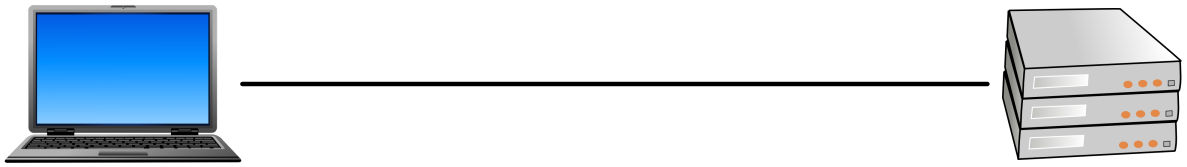
Example: Web Browsing



Example: Web Browsing

```
GET /index.html HTTP/1.1
Host: www.cs.utah.edu
Accept-Encoding: gzip, deflate
```

TCP

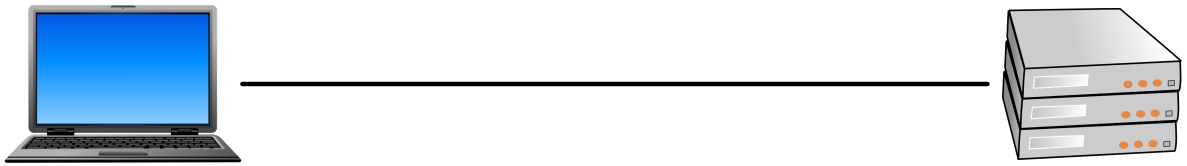


```
HTTP/1.1 301 Moved Permanently
Date: Fri, 12 Jan 2024 15:32:54 GMT
Location: https://www.cs.utah.edu/index.html
```

Example: Web Browsing

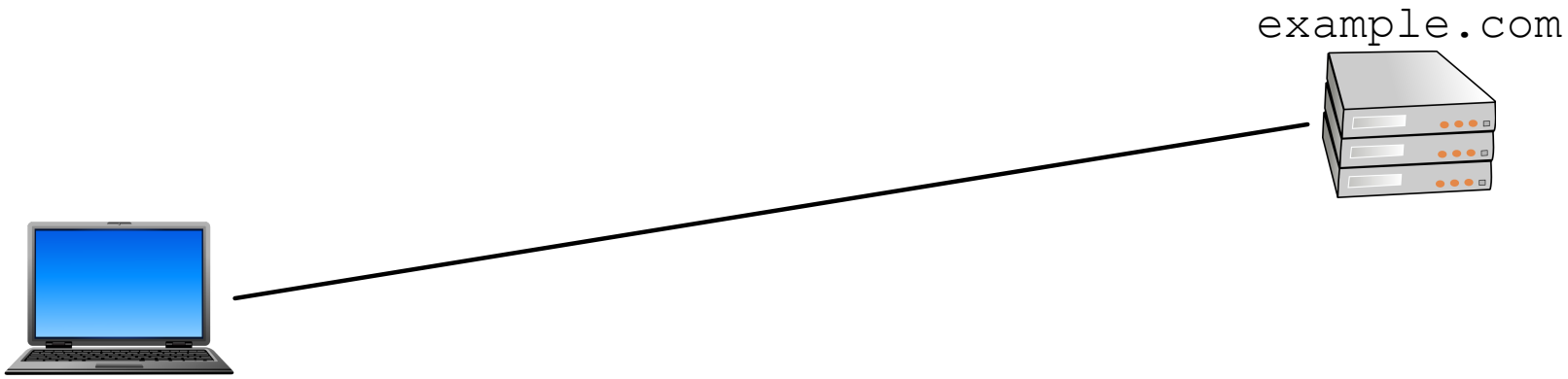
```
netcat www.cs.utah.edu 80
```

```
GET /index.html HTTP/1.1
Host: www.cs.utah.edu
Accept-Encoding: gzip, deflate
```

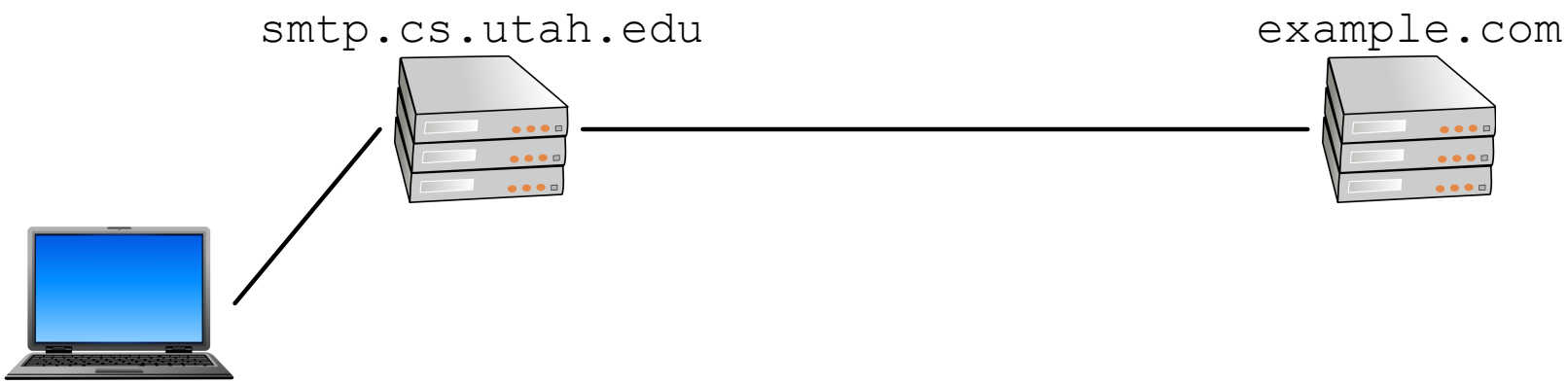


```
HTTP/1.1 301 Moved Permanently
Date: Fri, 12 Jan 2024 15:32:54 GMT
Location: https://www.cs.utah.edu/index.html
```

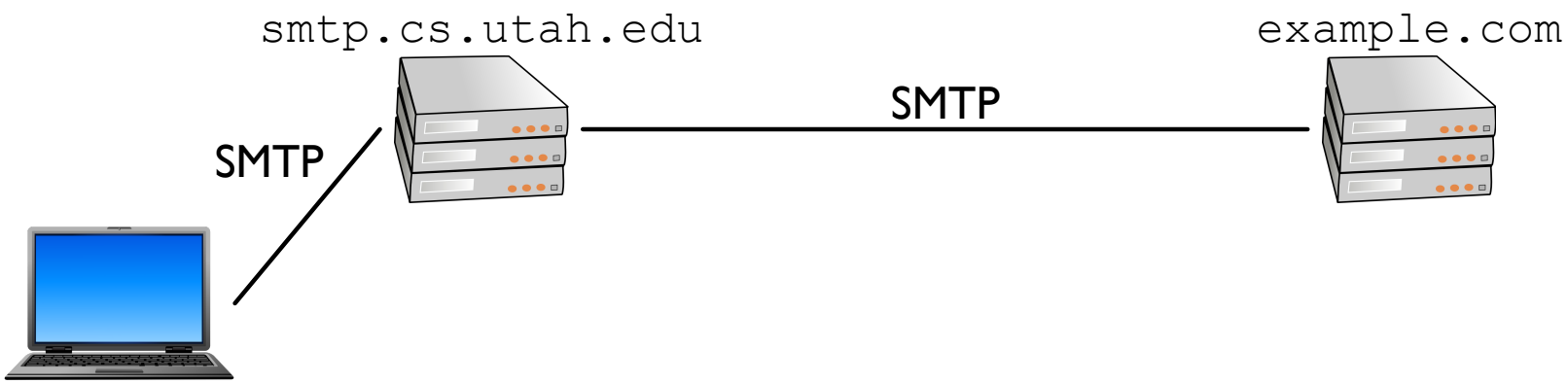
Example: Email



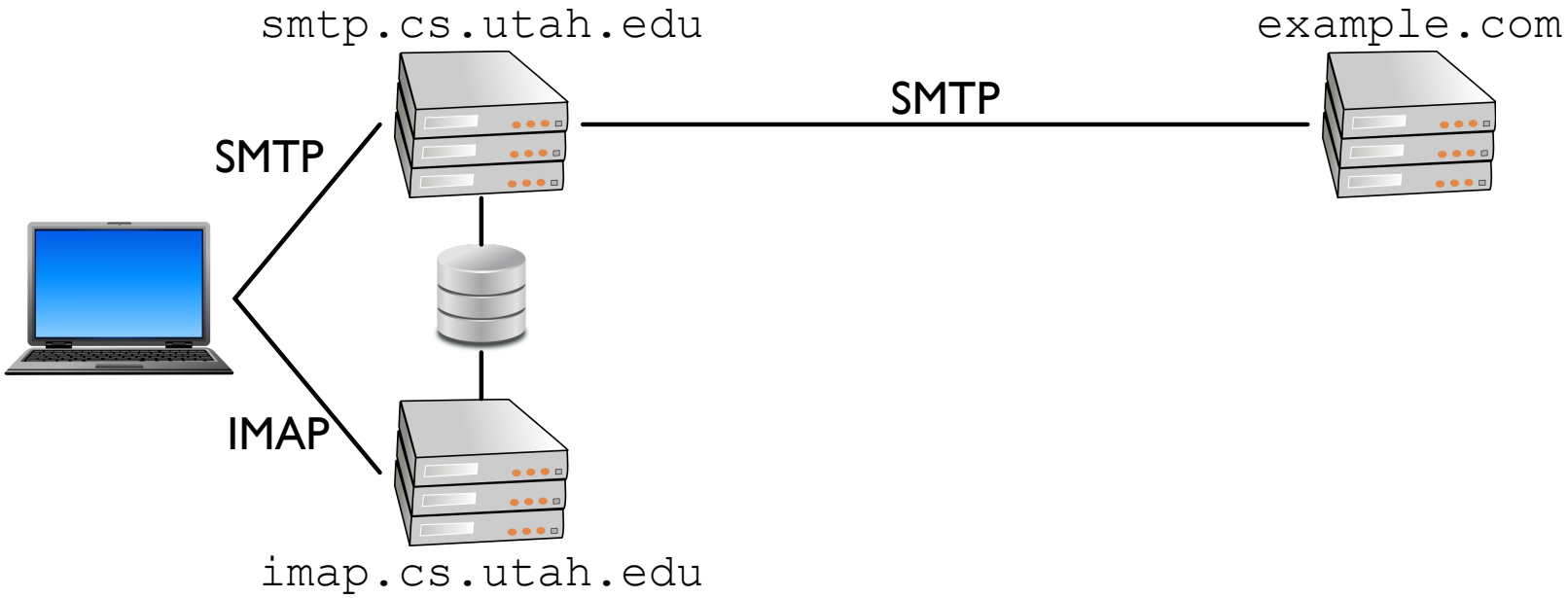
Example: Email



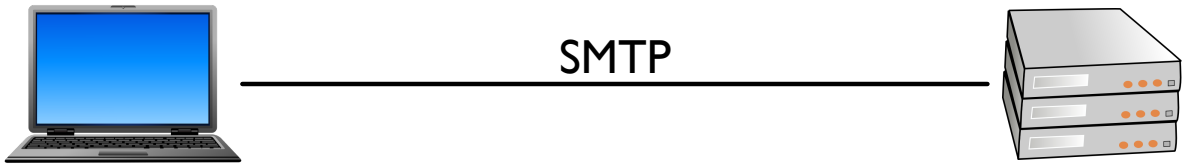
Example: Email



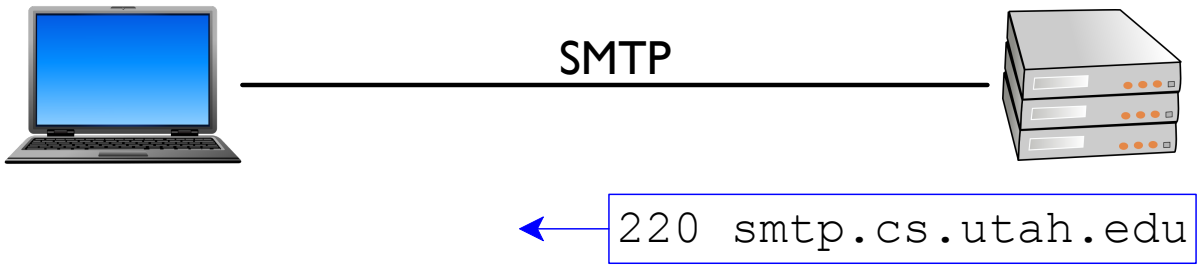
Example: Email



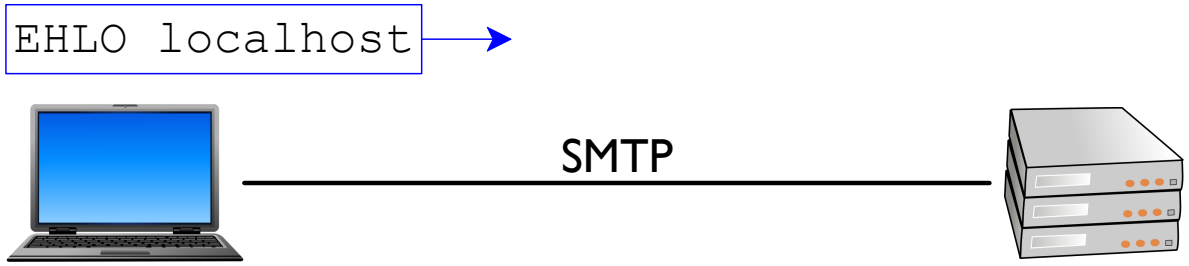
Example: Sending Email



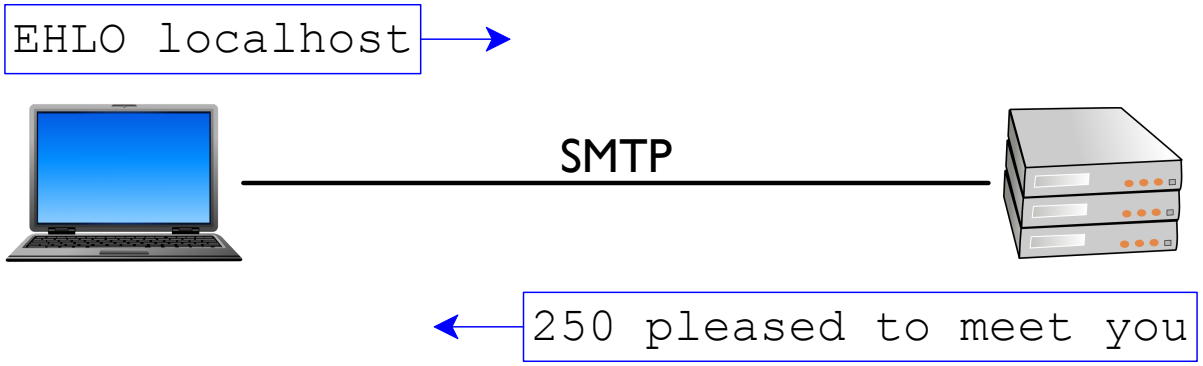
Example: Sending Email



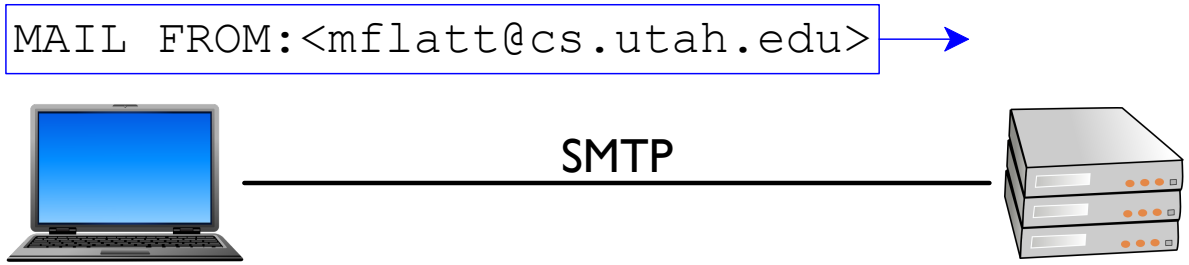
Example: Sending Email



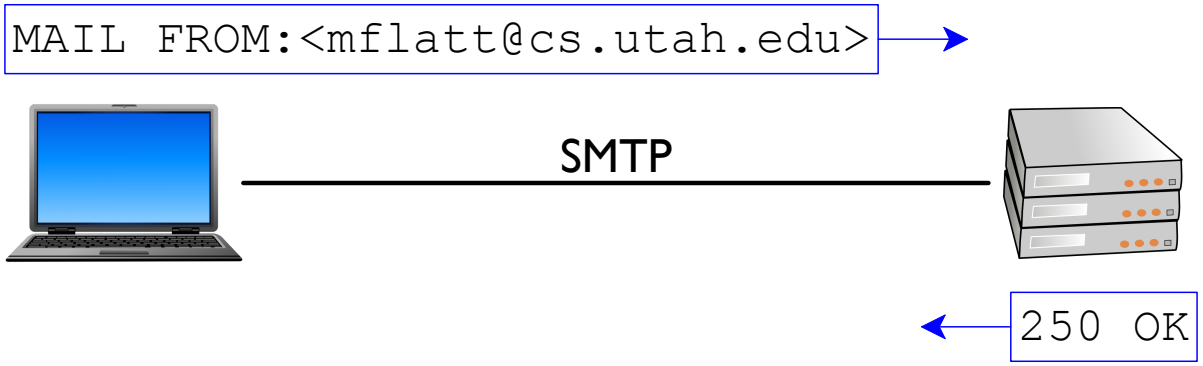
Example: Sending Email



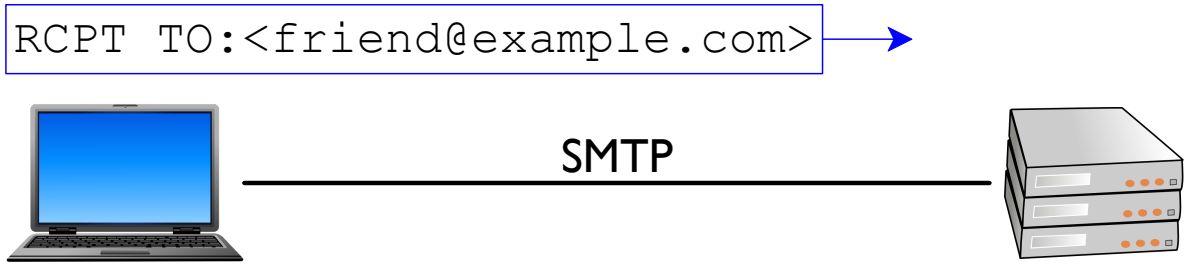
Example: Sending Email



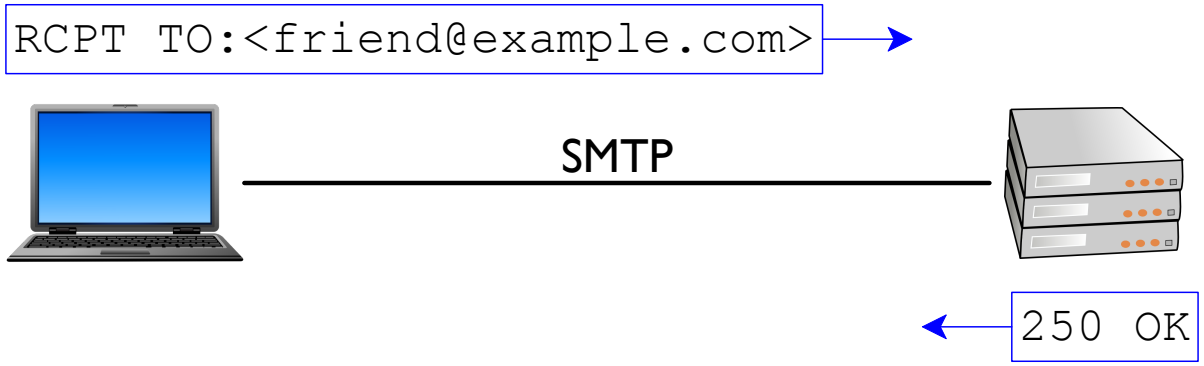
Example: Sending Email



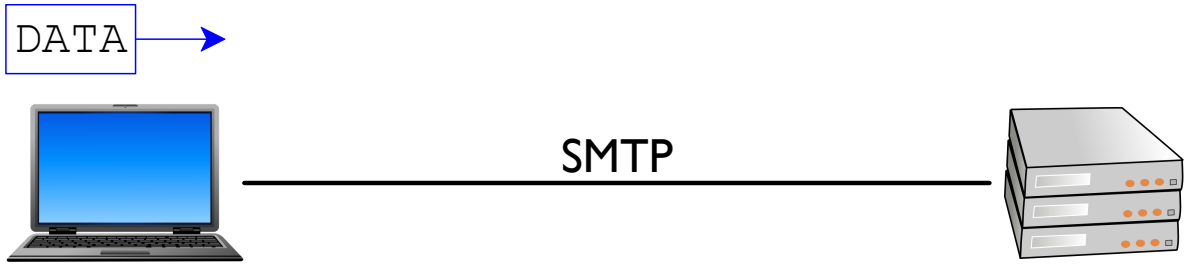
Example: Sending Email



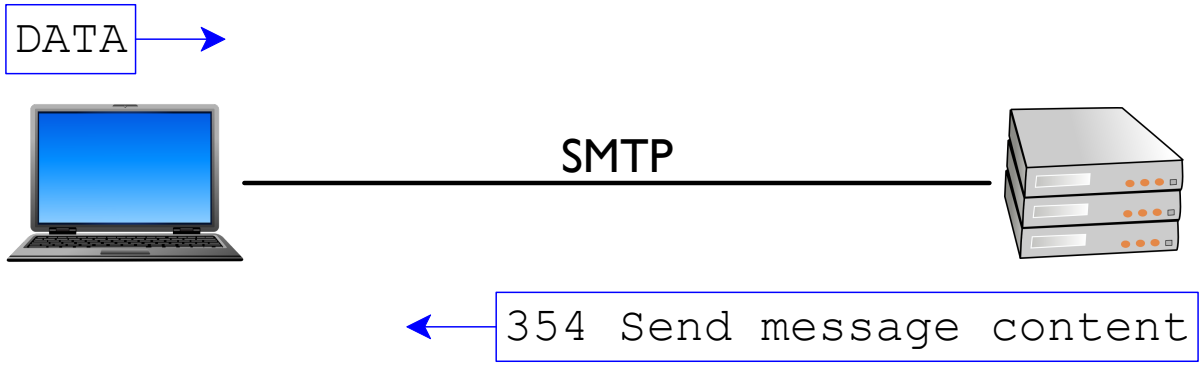
Example: Sending Email



Example: Sending Email

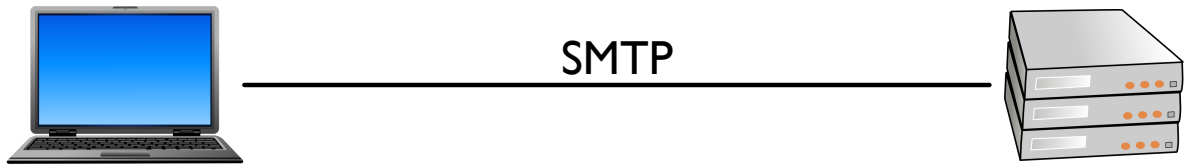


Example: Sending Email

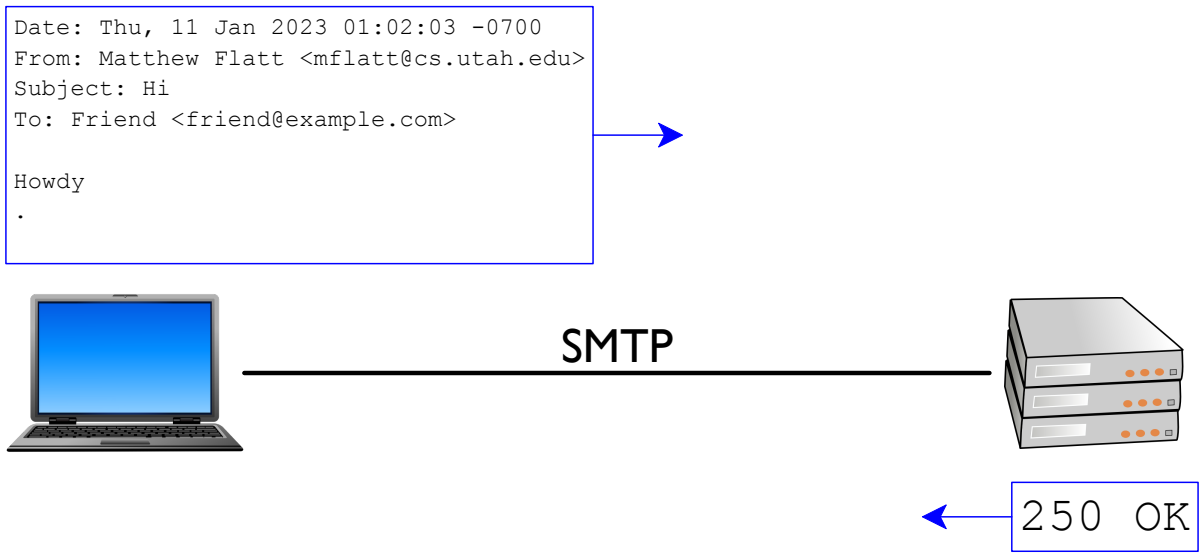


Example: Sending Email

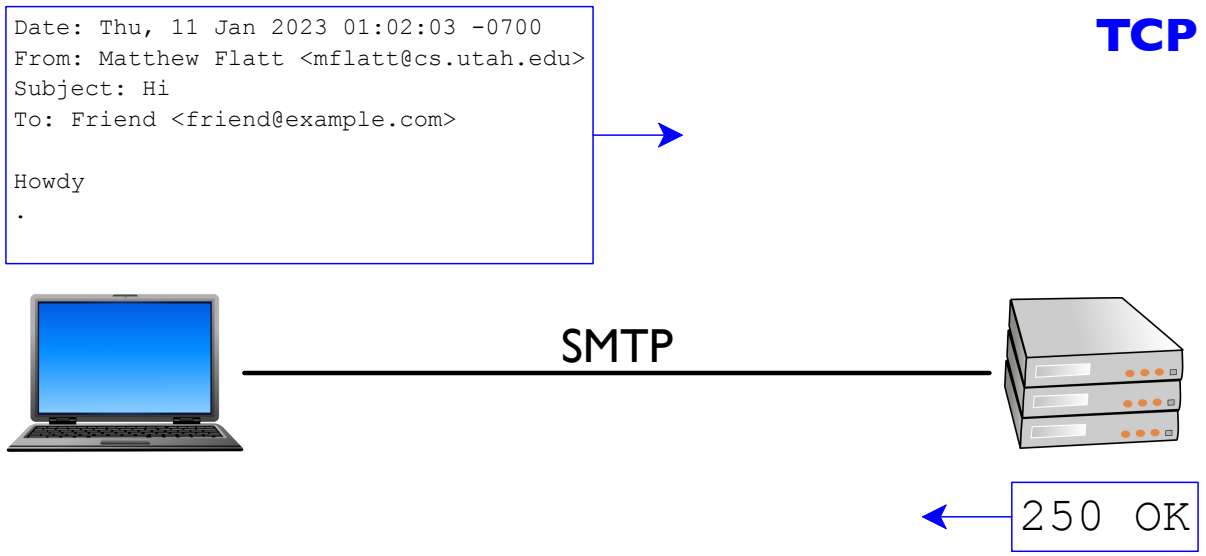
```
Date: Thu, 11 Jan 2023 01:02:03 -0700
From: Matthew Flatt <mflatt@cs.utah.edu>
Subject: Hi
To: Friend <friend@example.com>

Howdy
.
```



Example: Sending Email



Example: Sending Email



Example: Sending Email

```
netcat smtp.cs.utah.edu 25
```

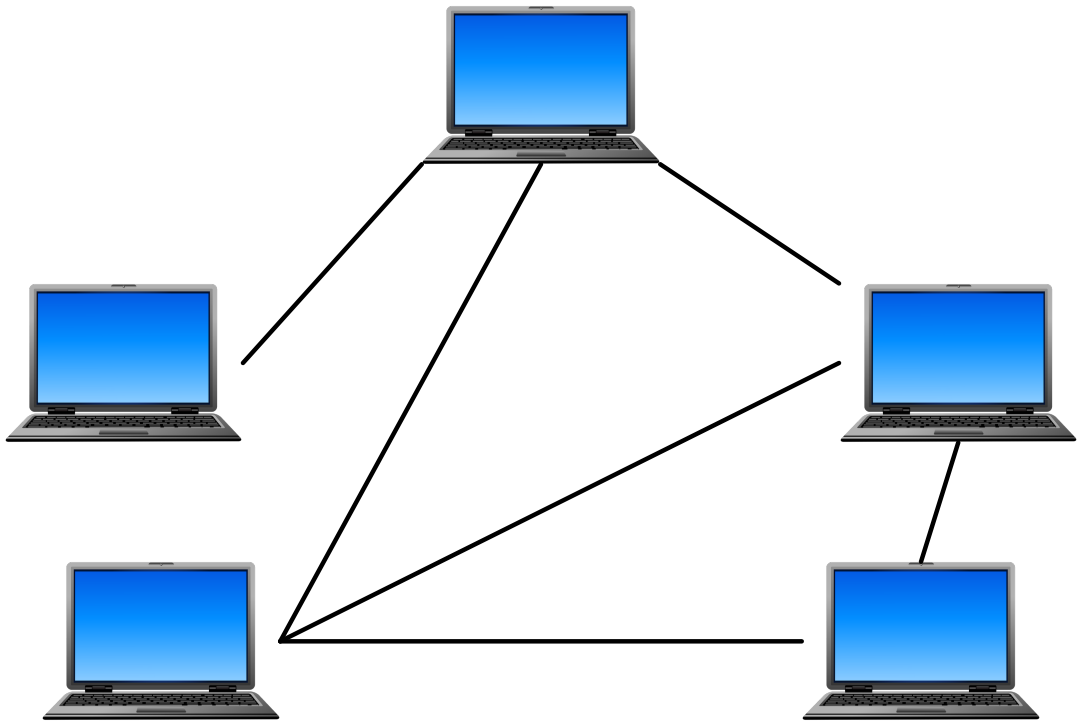
```
Date: Thu, 11 Jan 2023 01:02:03 -0700
From: Matthew Flatt <mflatt@cs.utah.edu>
Subject: Hi
To: Friend <friend@example.com>

Howdy
.
```

SMTP

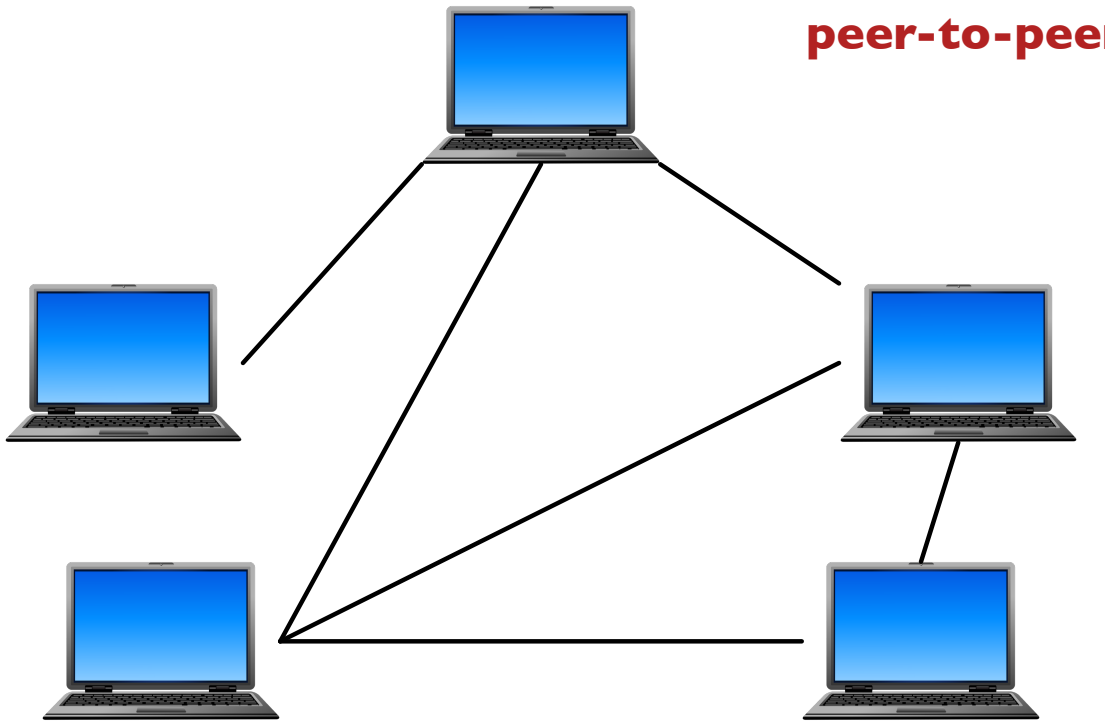
TCP

```
250 OK
```

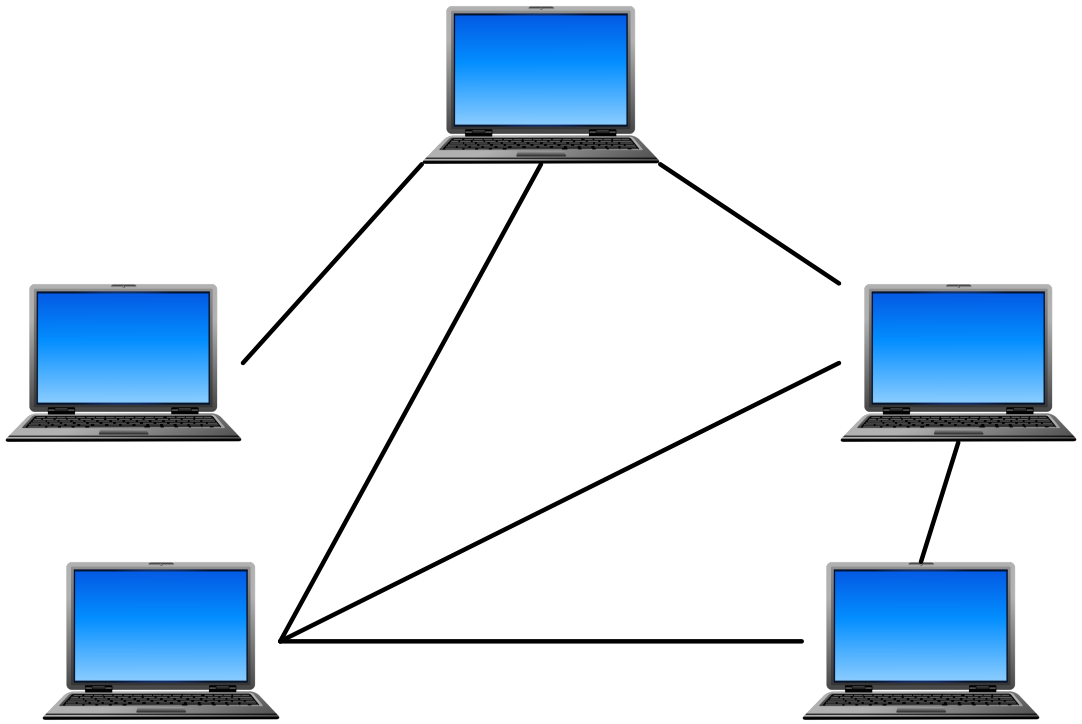
Example: BitTorrent



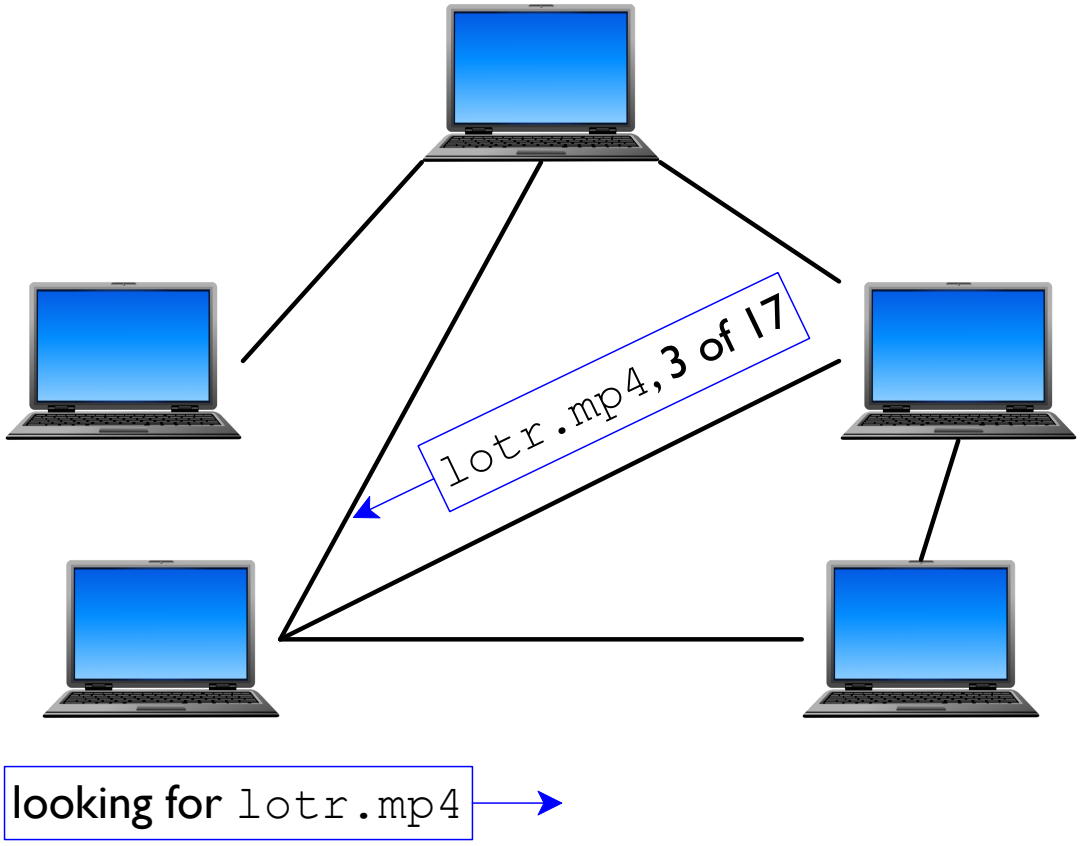
Example: BitTorrent



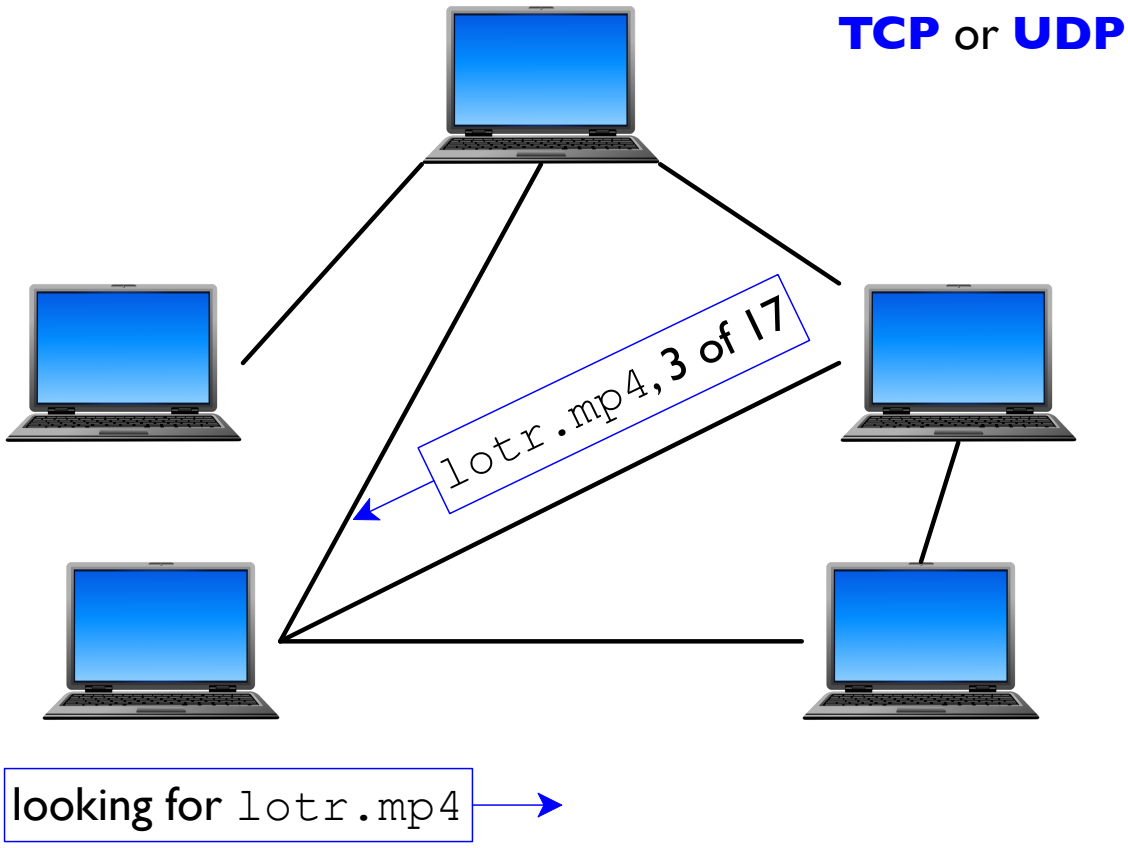
Example: BitTorrent



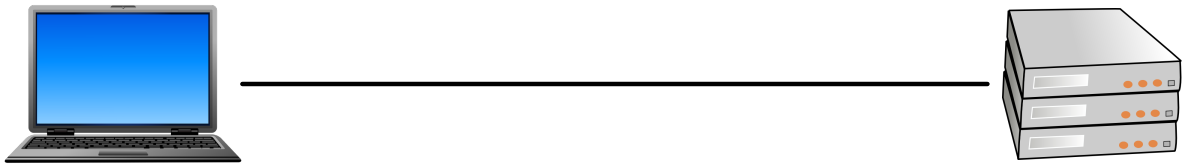
Example: BitTorrent



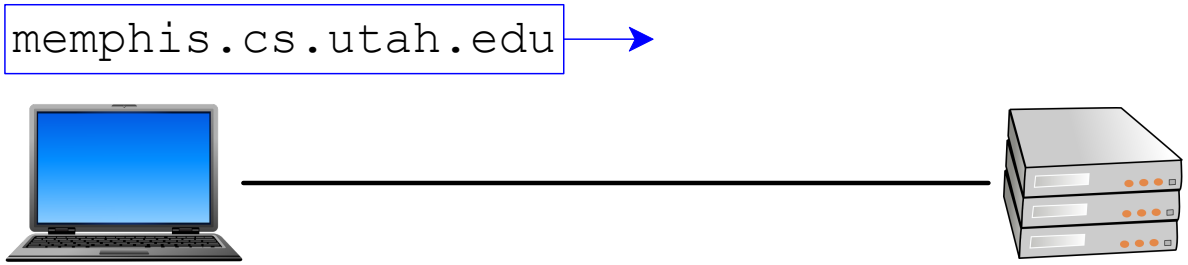
Example: BitTorrent



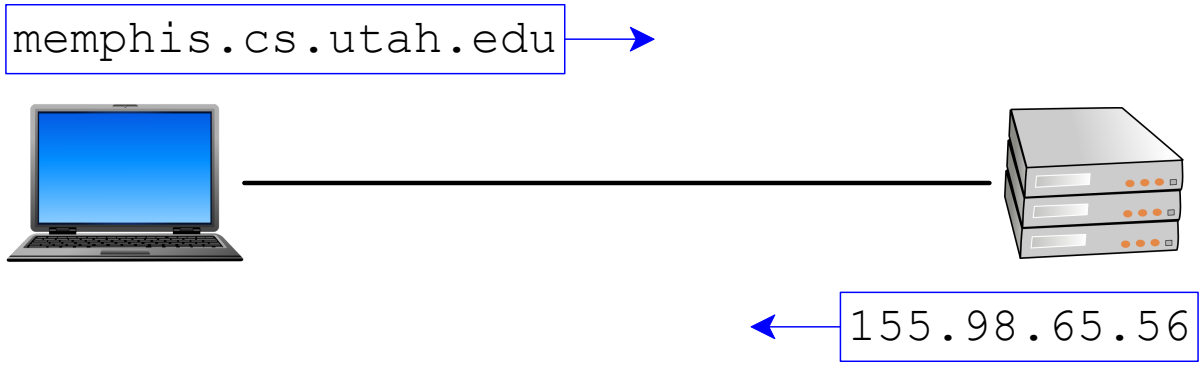
Example: DNS



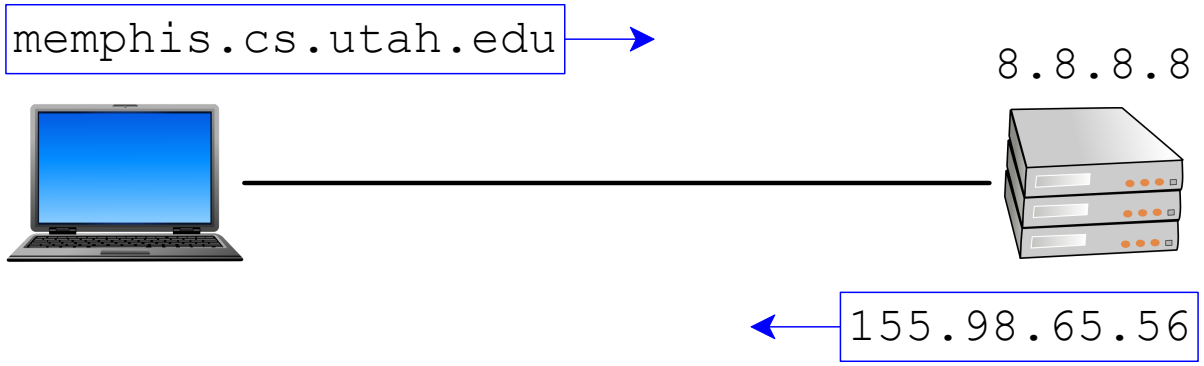
Example: DNS



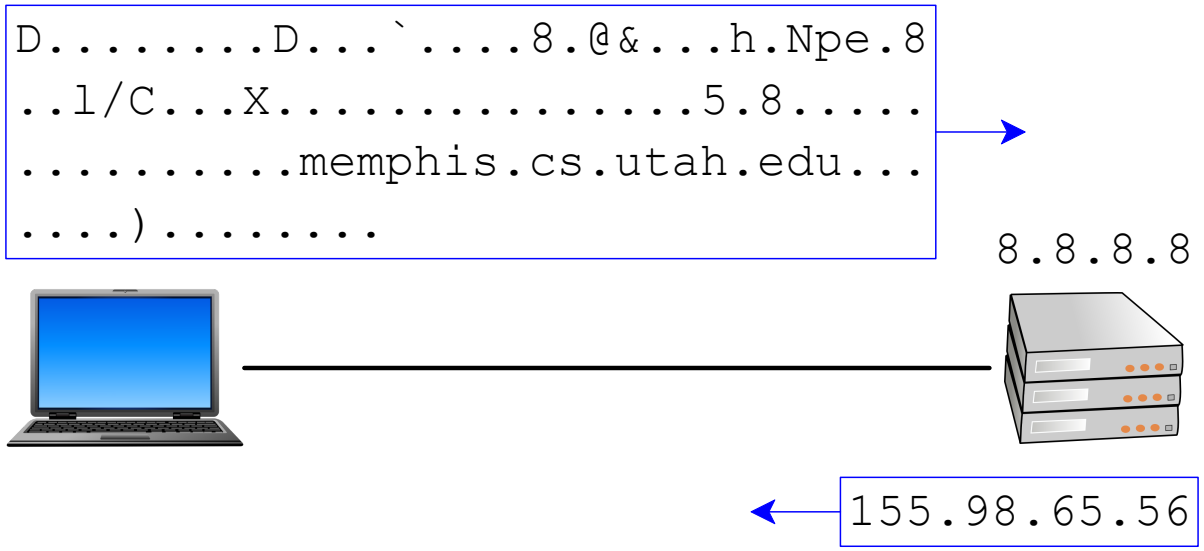
Example: DNS



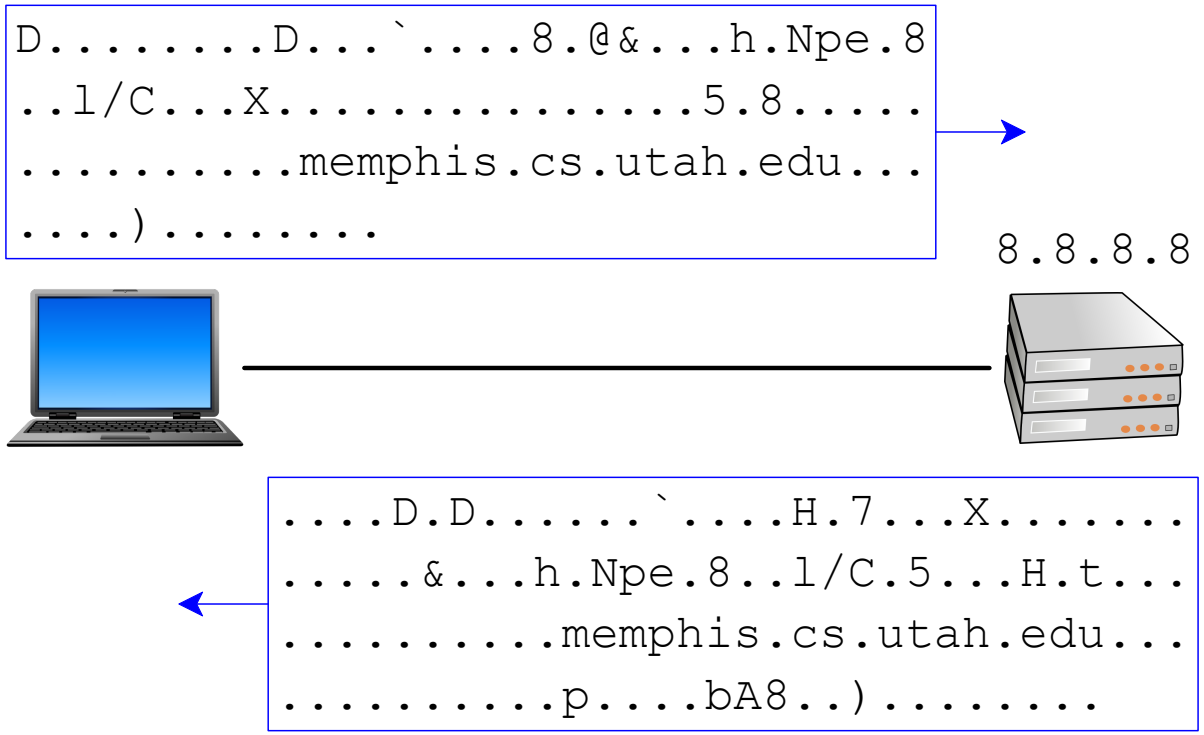
Example: DNS



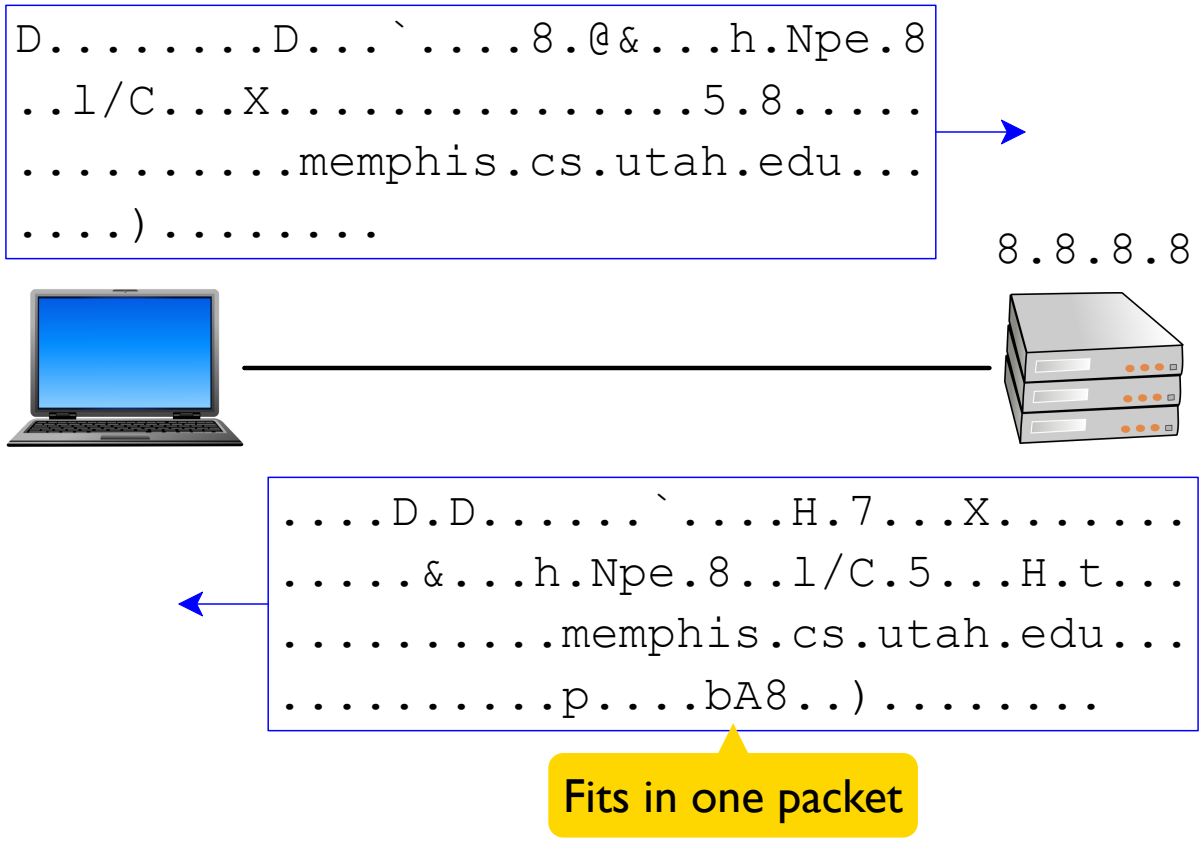
Example: DNS



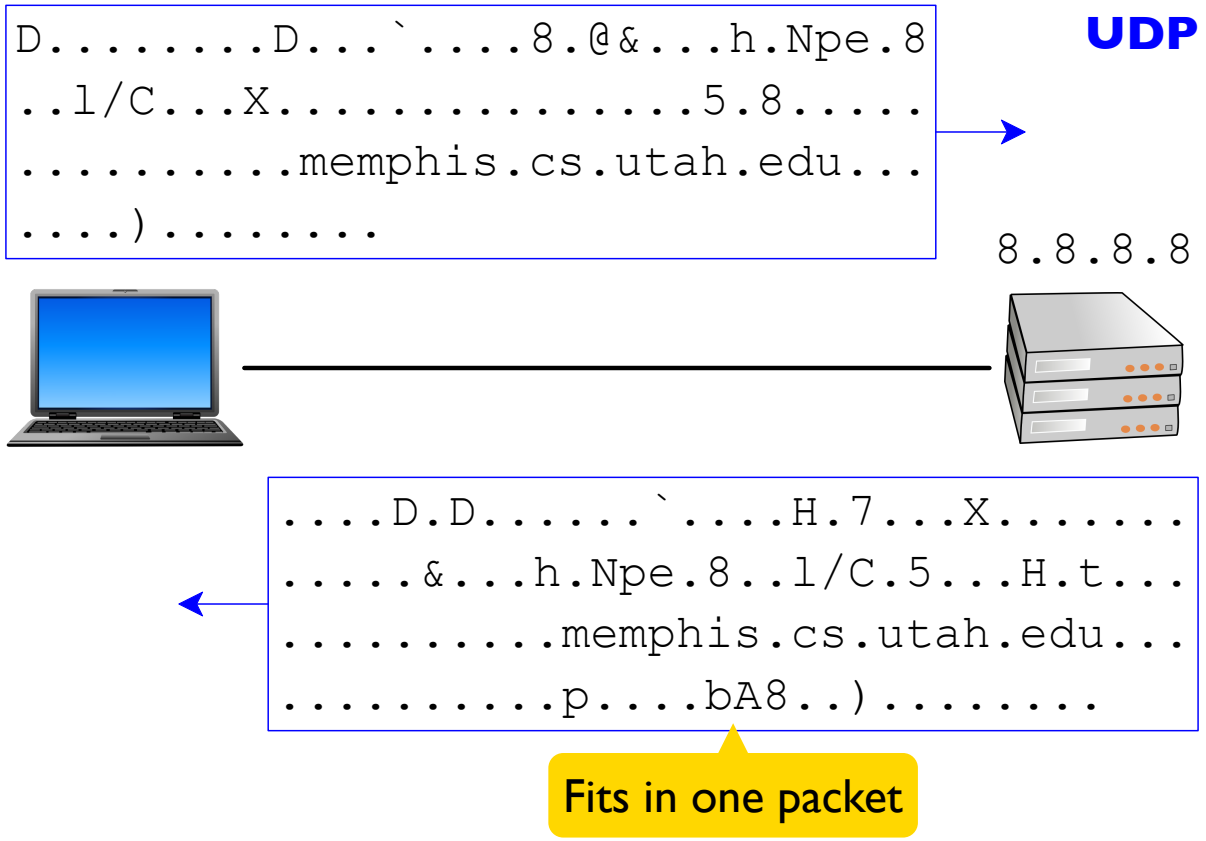
Example: DNS


D.......D...`....8.@&...h.Npe.8
..l/C...X..............5.8.....
.........memphis.cs.utah.edu...
....).......
8.8.8.8
....D.D.....`....H.7...X.......
.....&...h.Npe.8..l/C.5...H.t...
.........memphis.cs.utah.edu...
.........p....bA8..).......

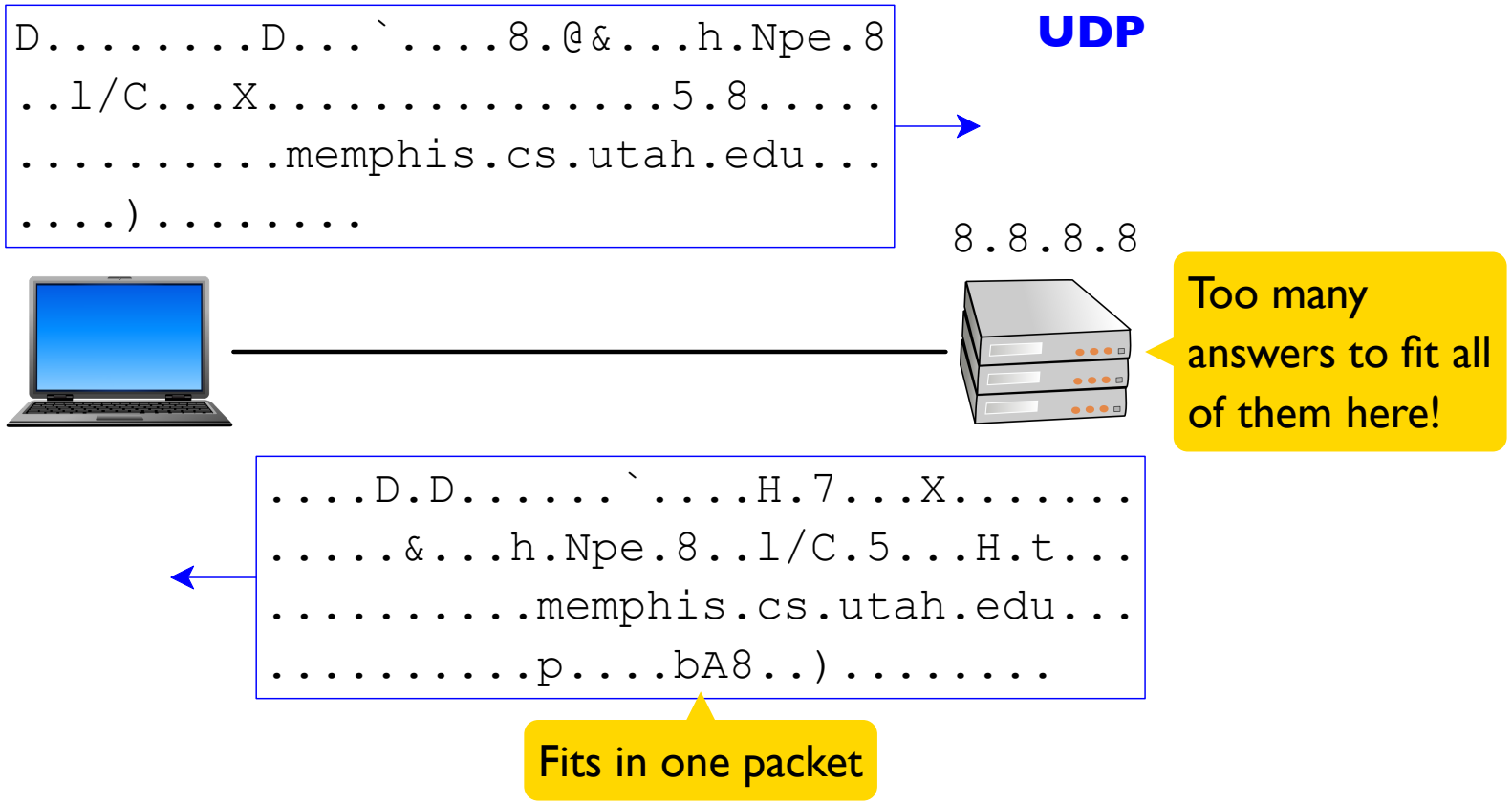
Example: DNS


D.......D...`....8.@&...h.Npe.8
..l/C...X..............5.8.....
.........memphis.cs.utah.edu...
....).......
8.8.8.8
....D.D.....`....H.7...X.......
.....&...h.Npe.8..l/C.5...H.t...
.........memphis.cs.utah.edu...
.........p....bA8..).......
Fits in one packet

Example: DNS


D.......D...`....8.@&...h.Npe.8
..l/C...X.............5.8.....
.........memphis.cs.utah.edu...
....).......
UDP
8.8.8.8
....D.D.....`....H.7...X.......
.....&...h.Npe.8..l/C.5...H.t...
.........memphis.cs.utah.edu...
.........p....bA8..).......
Fits in one packet

Example: DNS


D.......D...`....8.@&...h.Npe.8
..l/C...X.............5.8.....
.........memphis.cs.utah.edu...
....).........
UDP
8.8.8.8
Too many answers to fit all of them here!
....D.D.....`....H.7...X.......
.....&...h.Npe.8..l/C.5...H.t...
.........memphis.cs.utah.edu...
.........p....bA8..).........
Fits in one packet

Name Resolution

`www.cs.utah.edu`

Name Resolution

domain name

`www.cs.utah.edu`

Name Resolution

domain name

`www.cs.utah.edu`

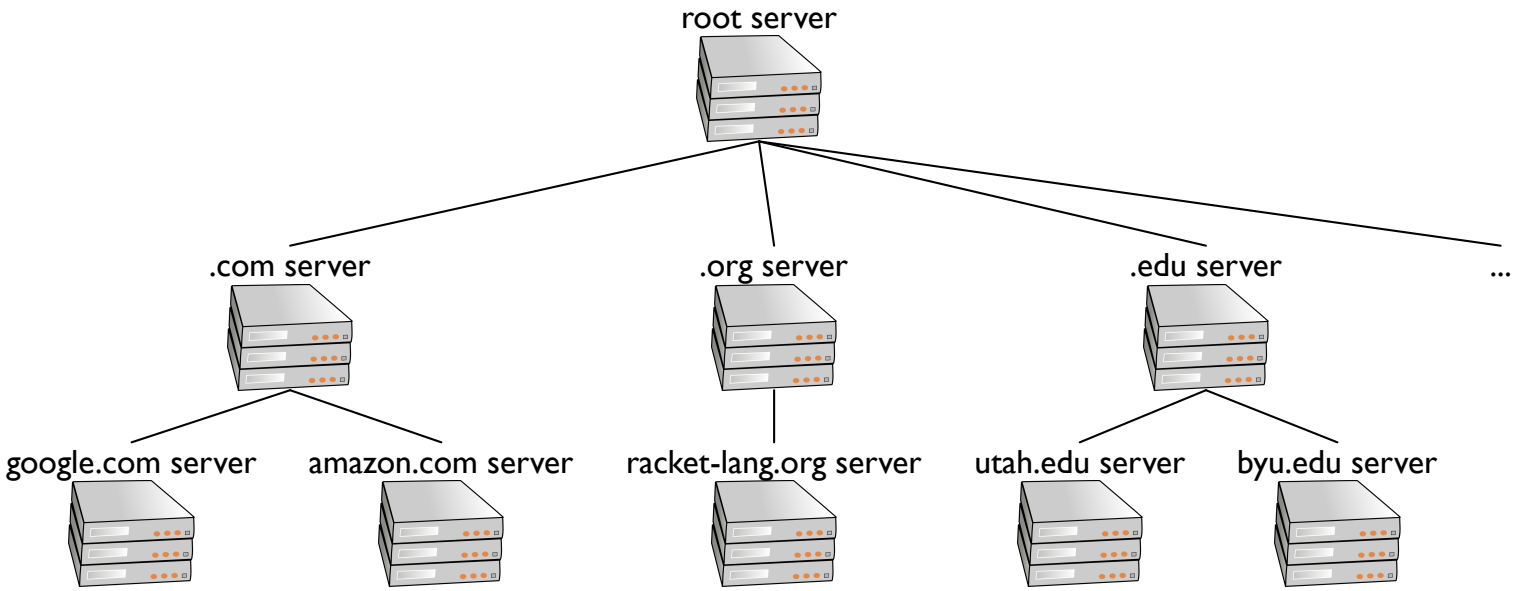
top-level domain

Name Resolution

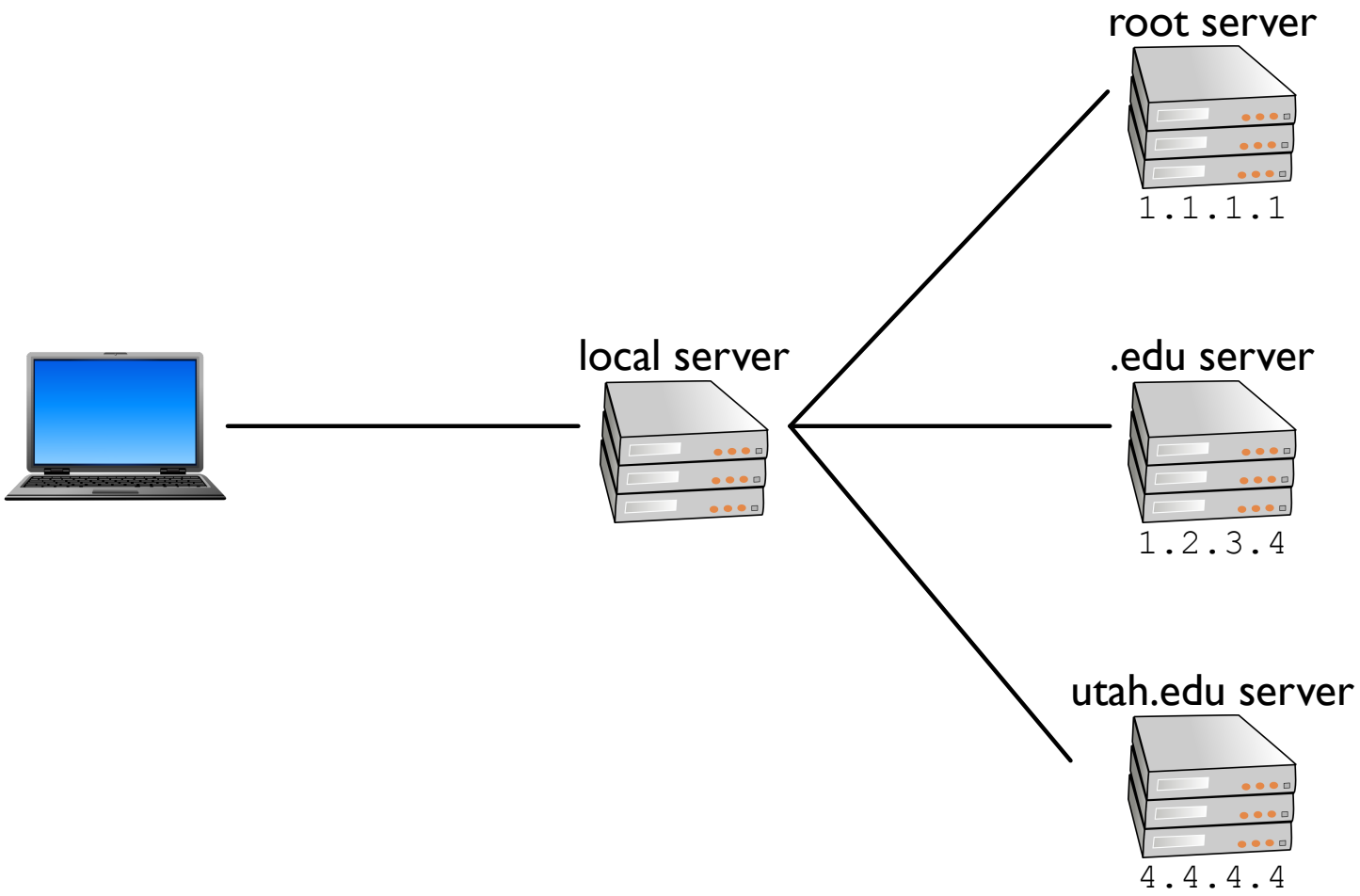
domain name

`www.cs.utah.edu`

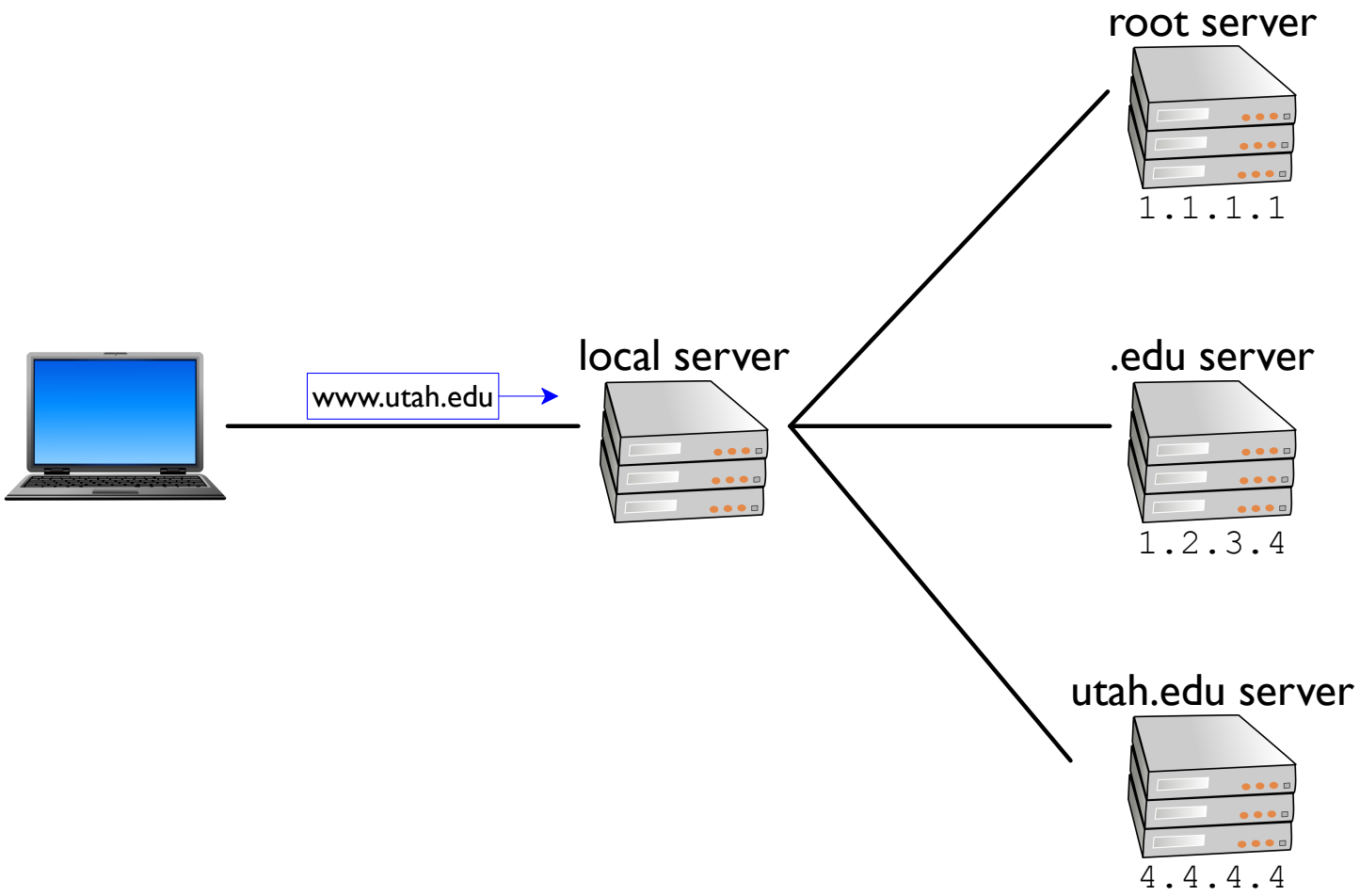
top-level domain



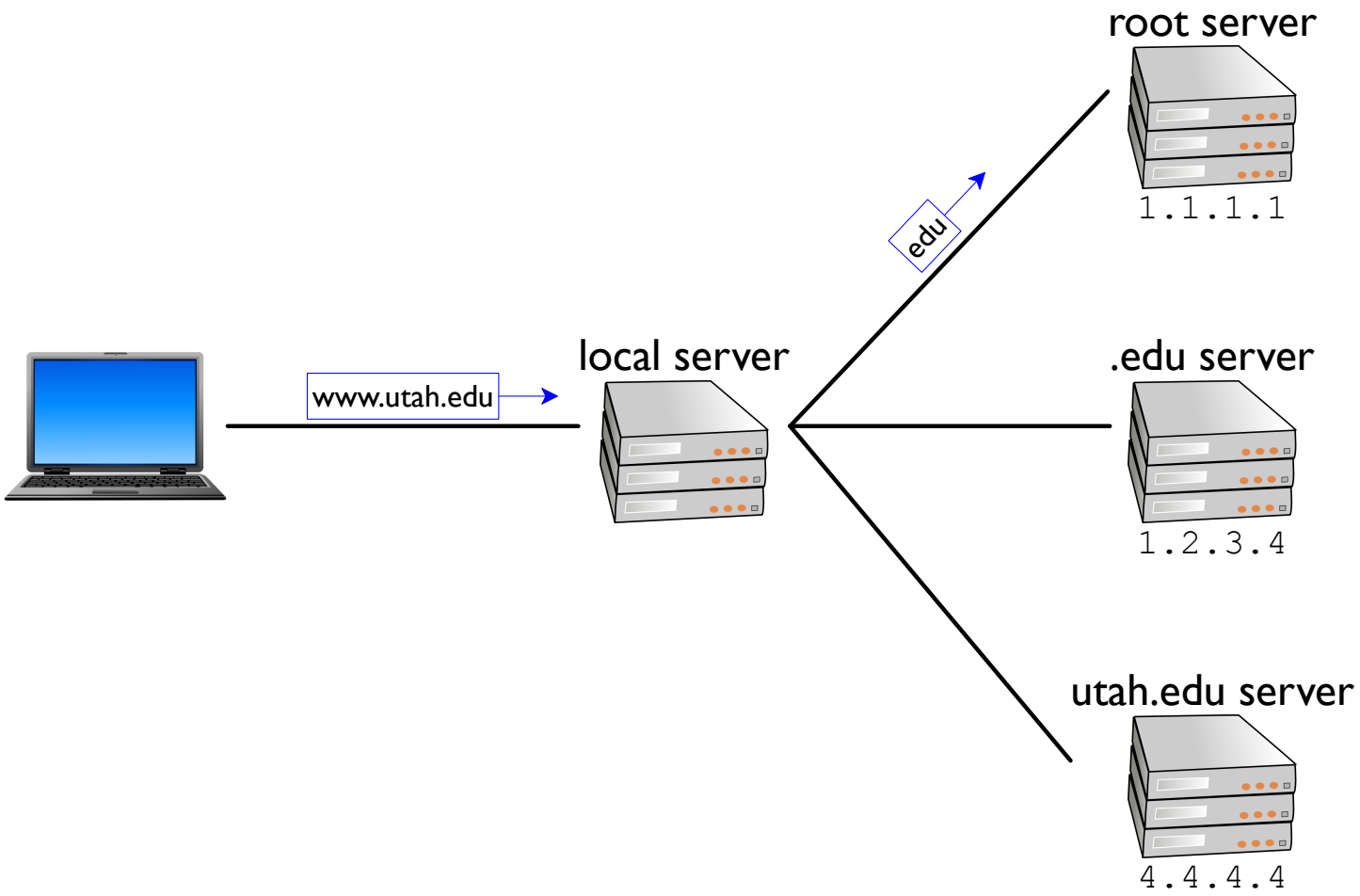
Name Resolution



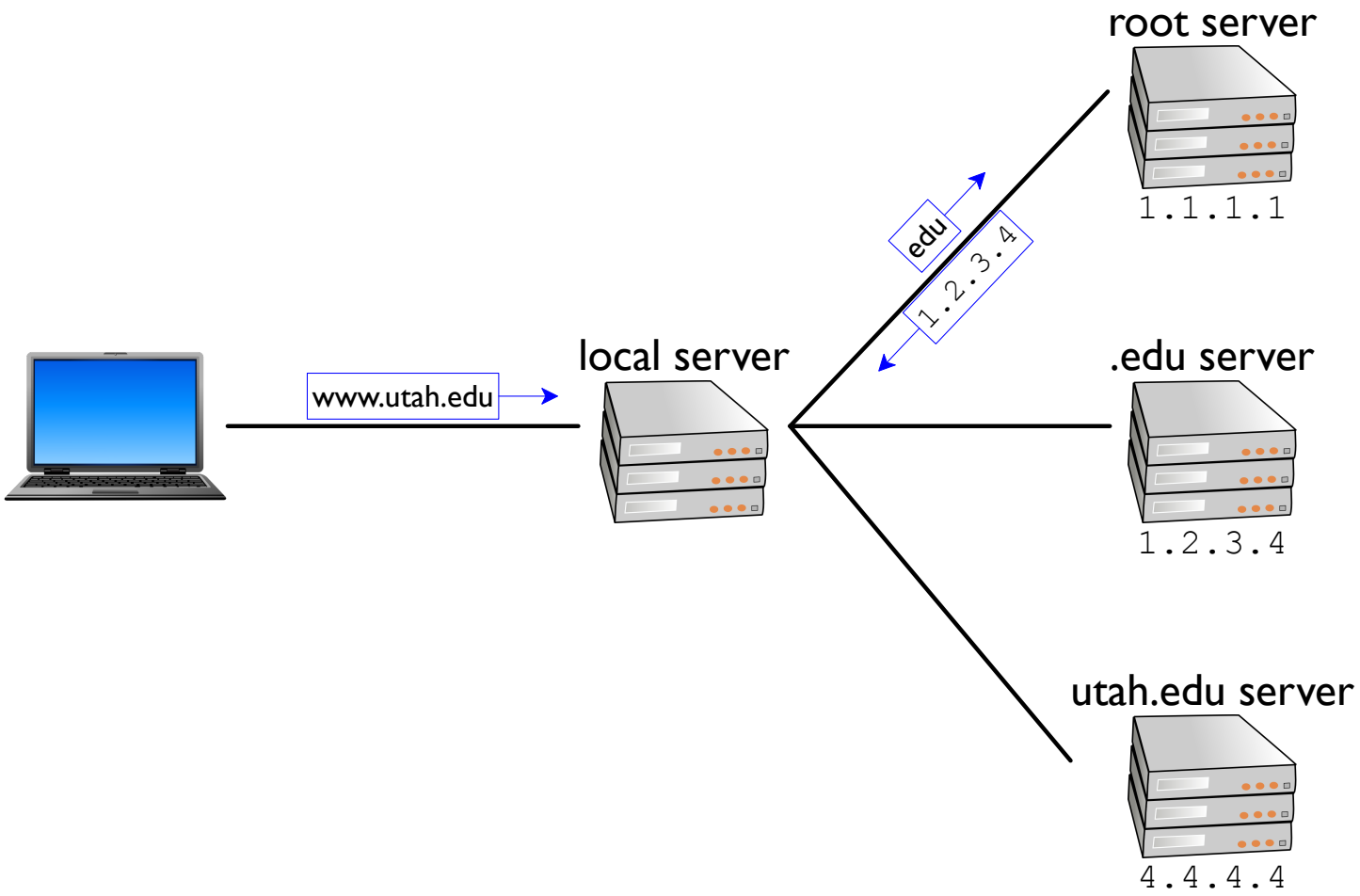
Name Resolution



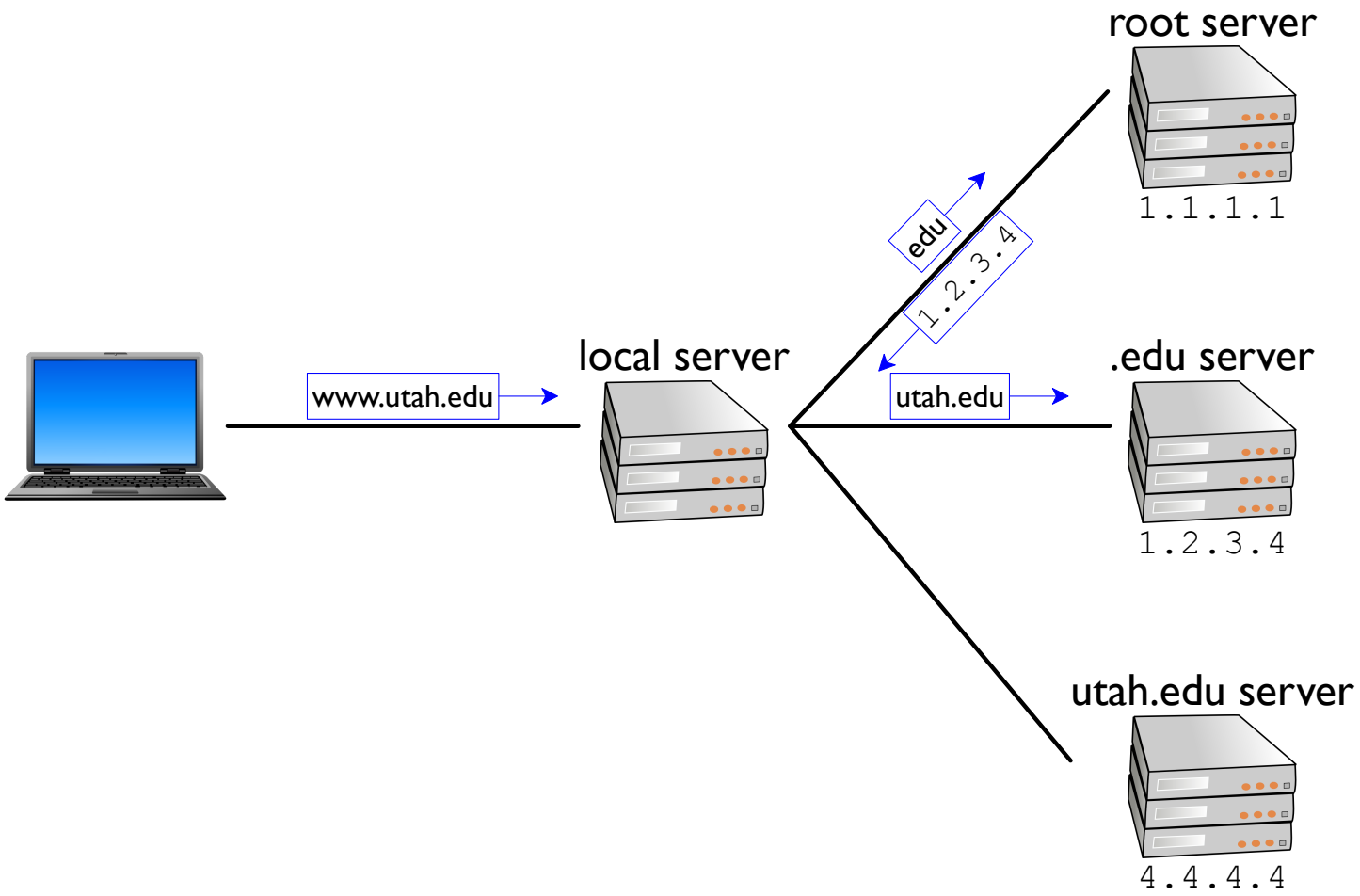
Name Resolution



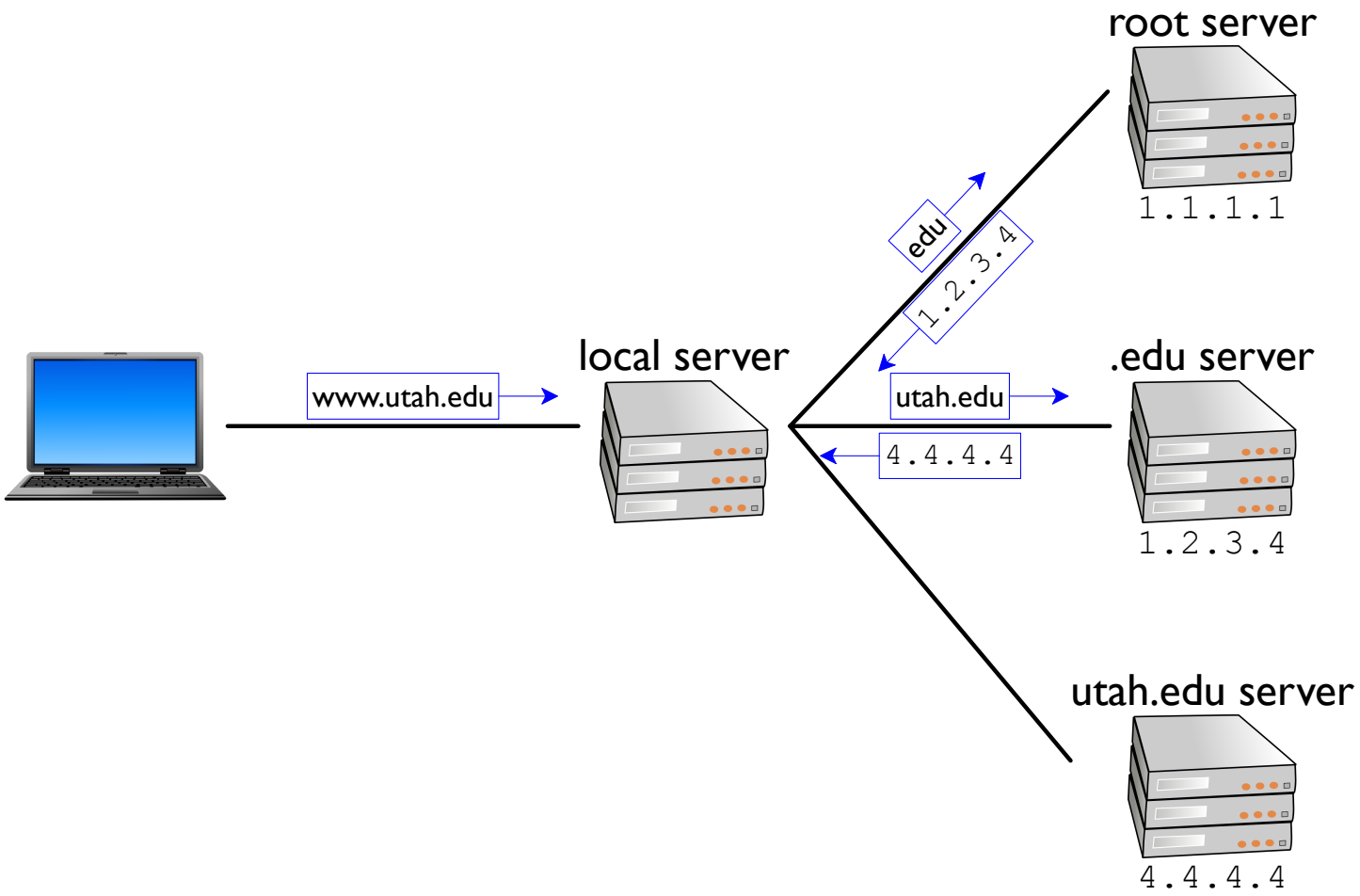
Name Resolution



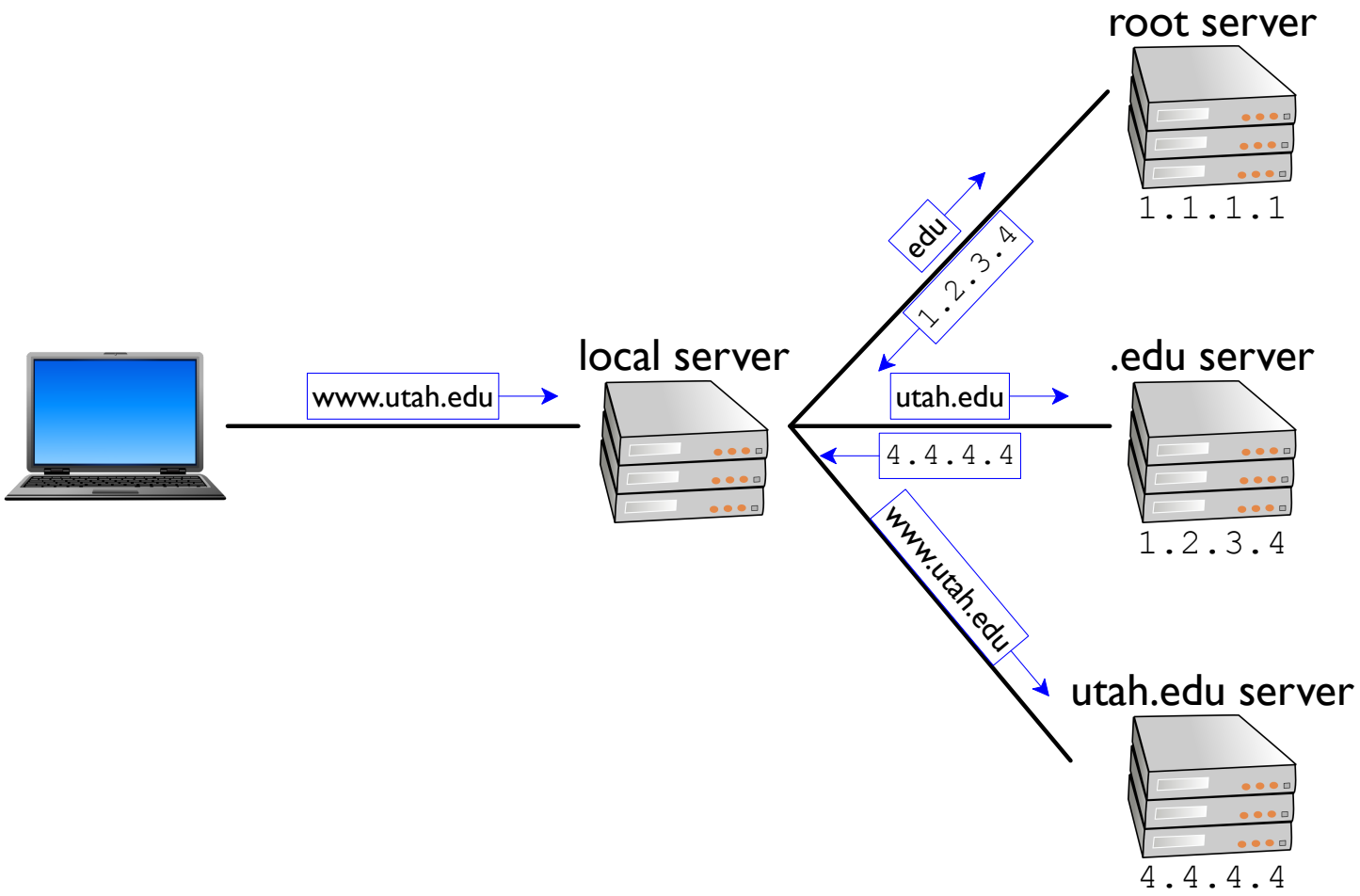
Name Resolution



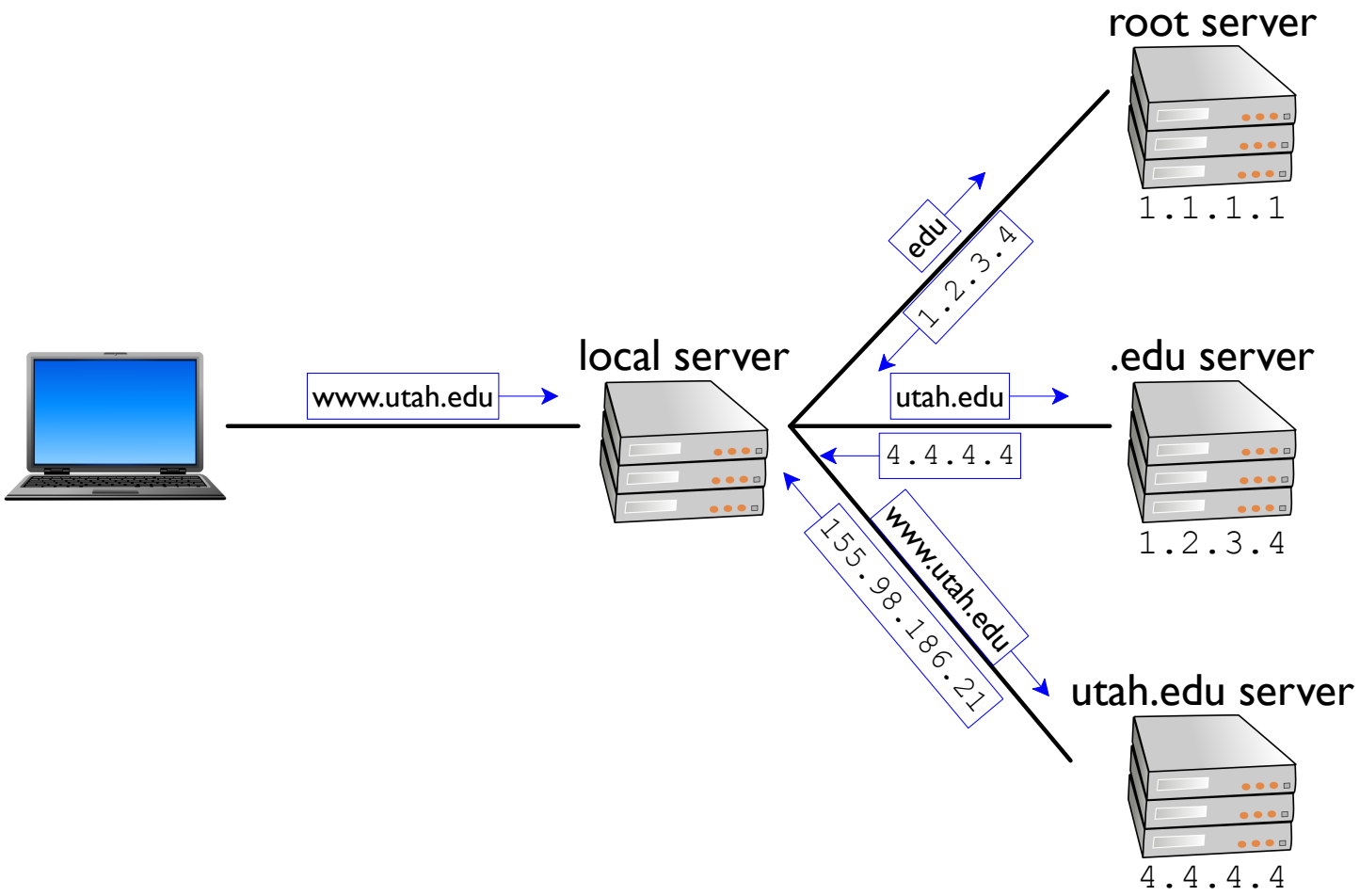
Name Resolution



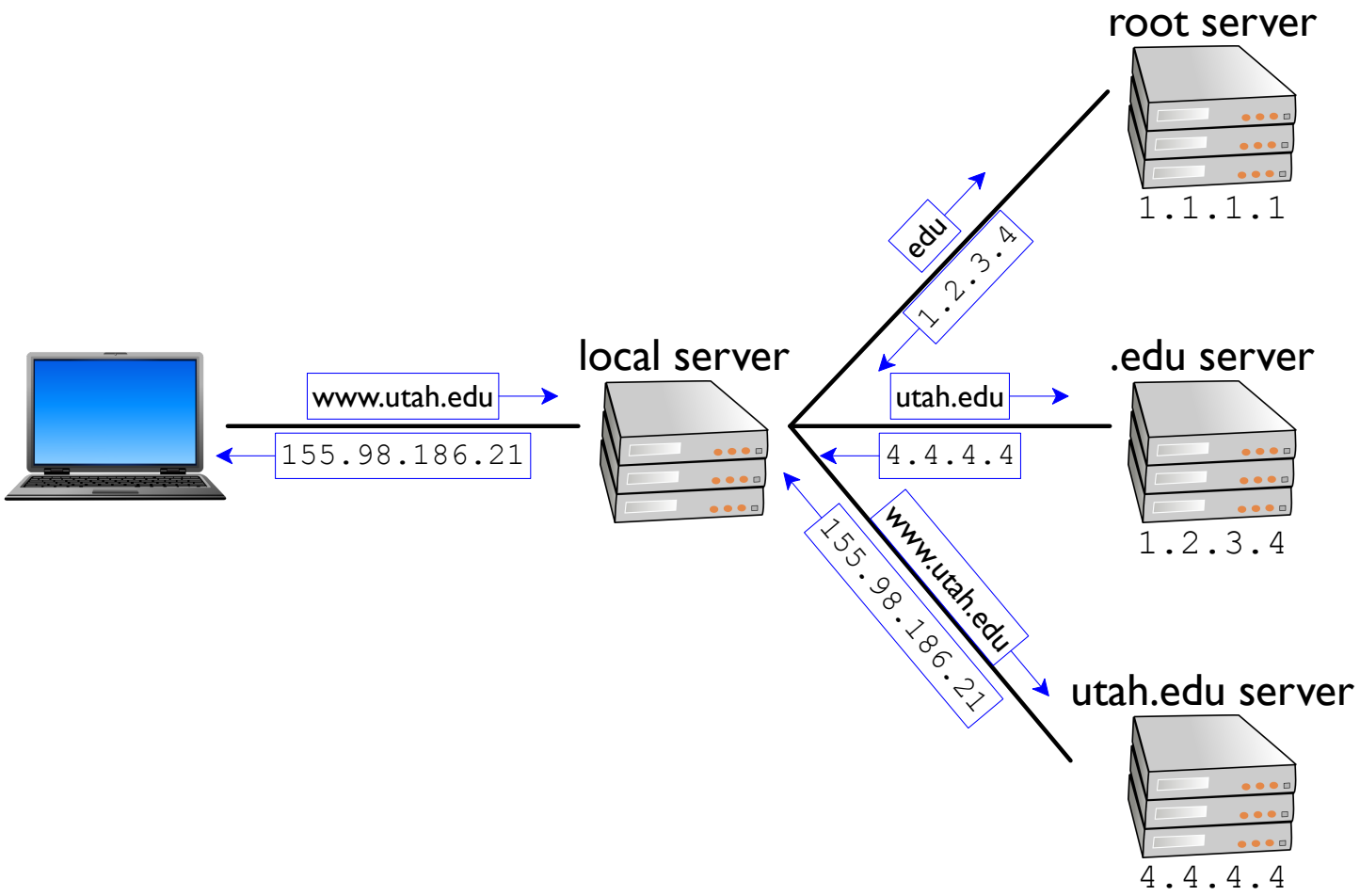
Name Resolution



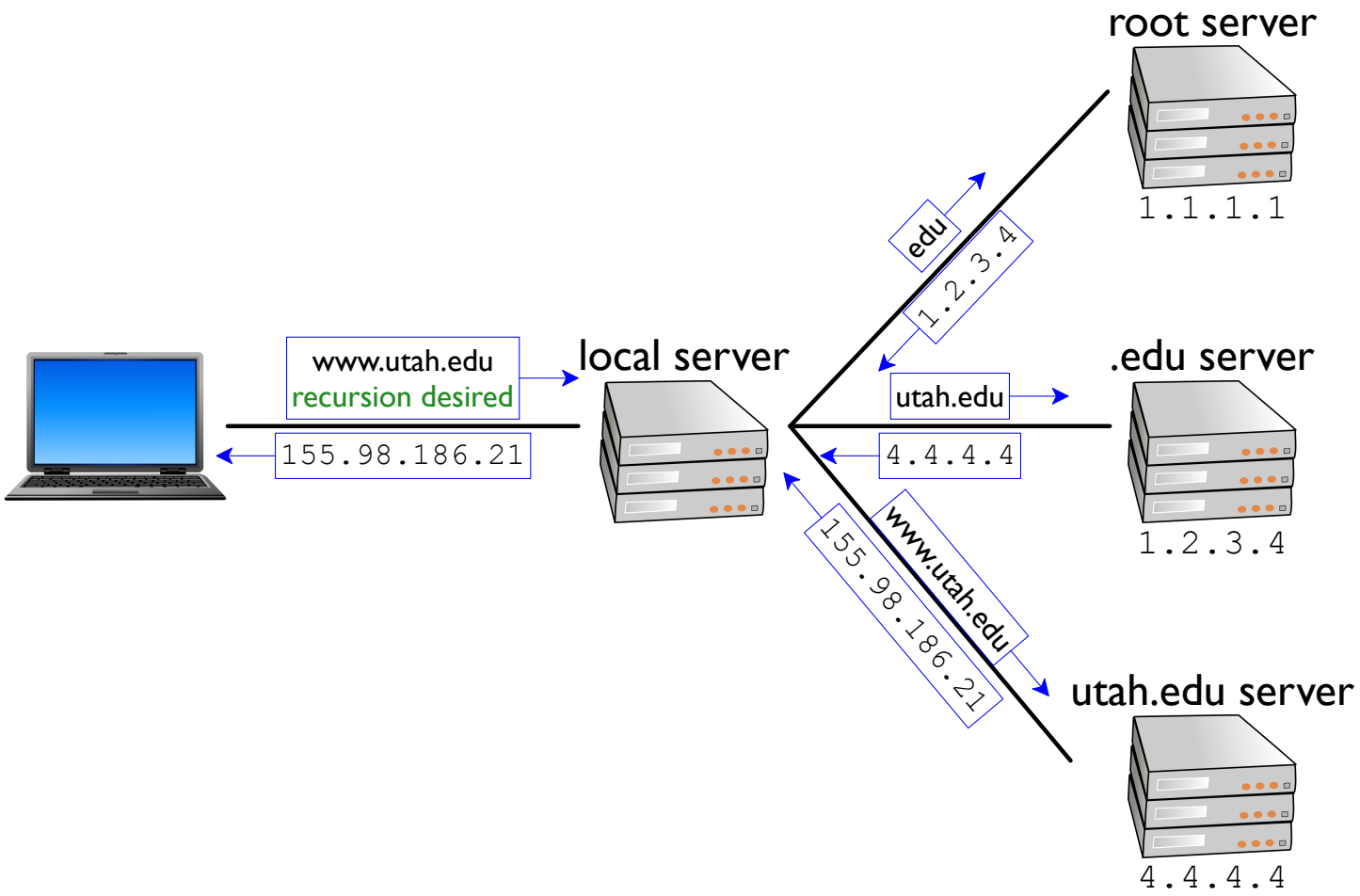
Name Resolution



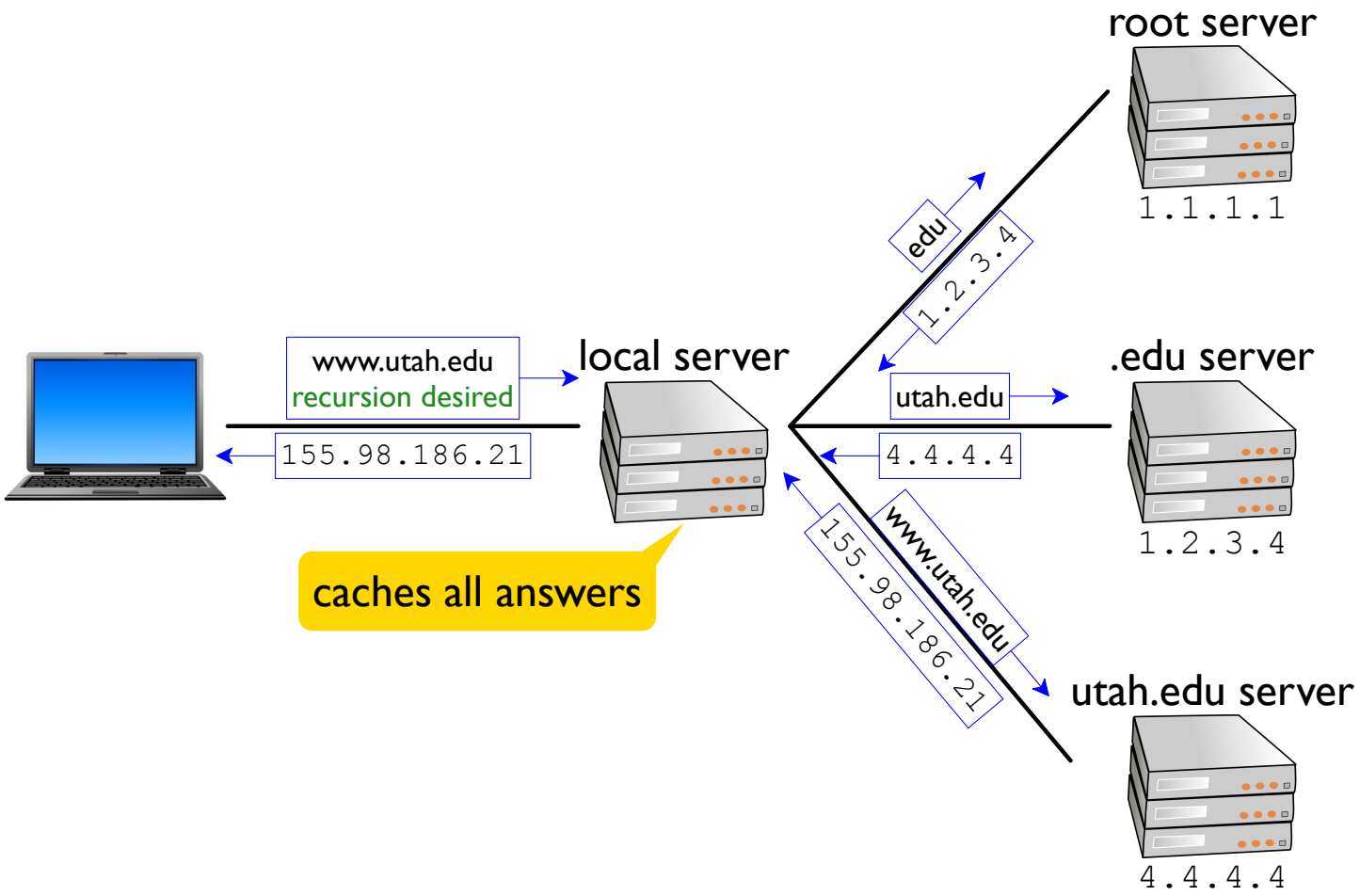
Name Resolution



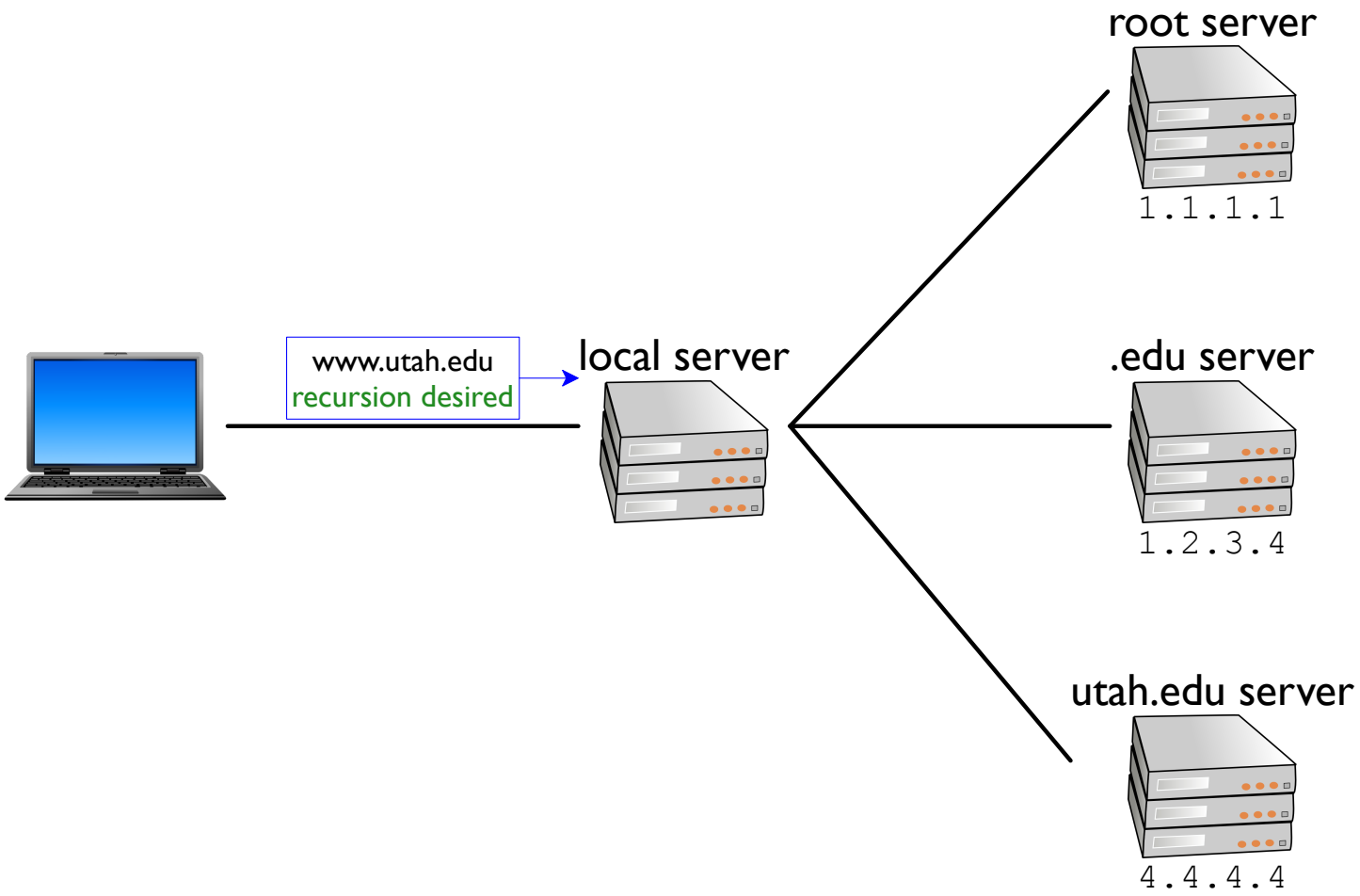
Name Resolution



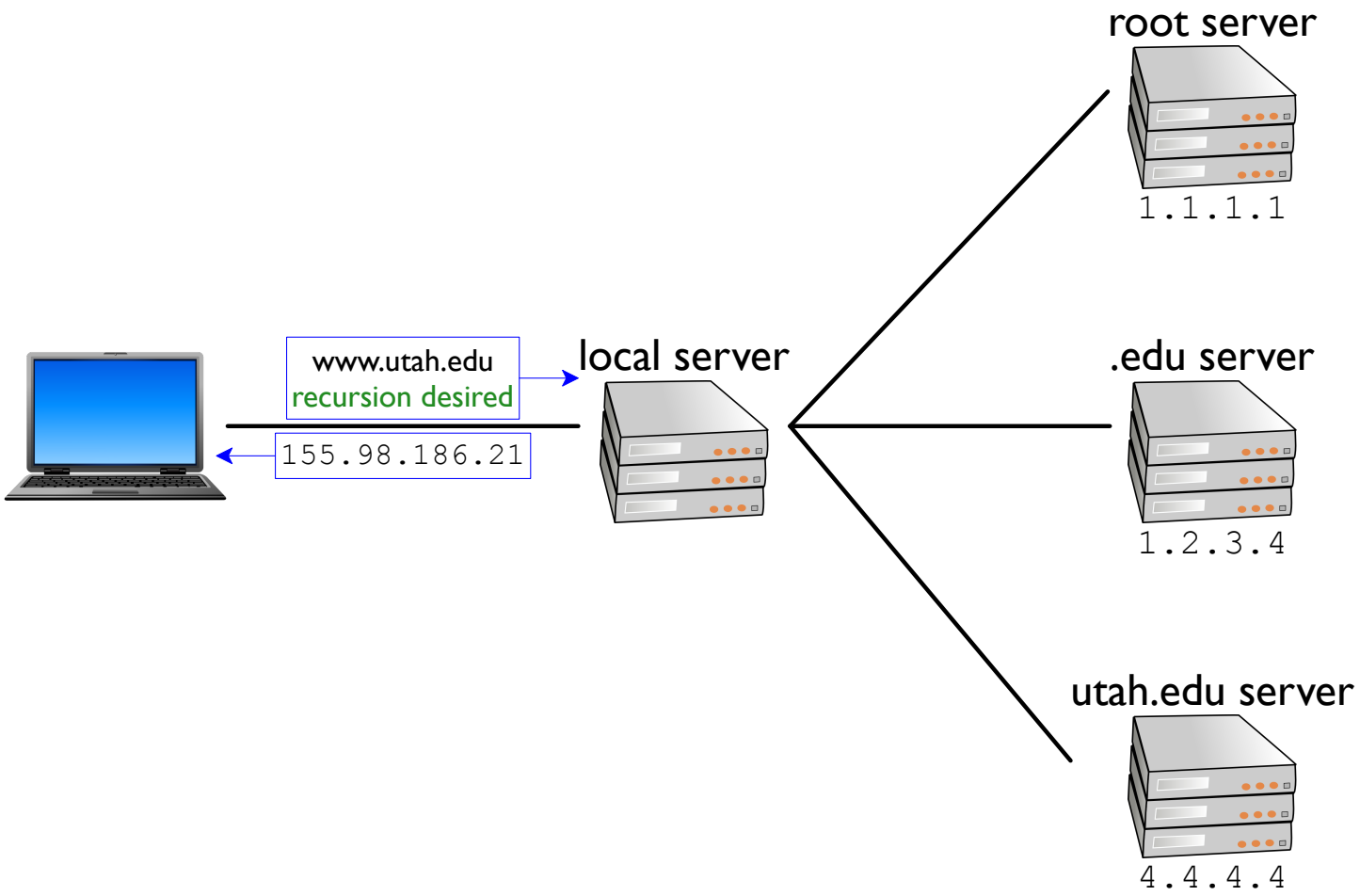
Name Resolution



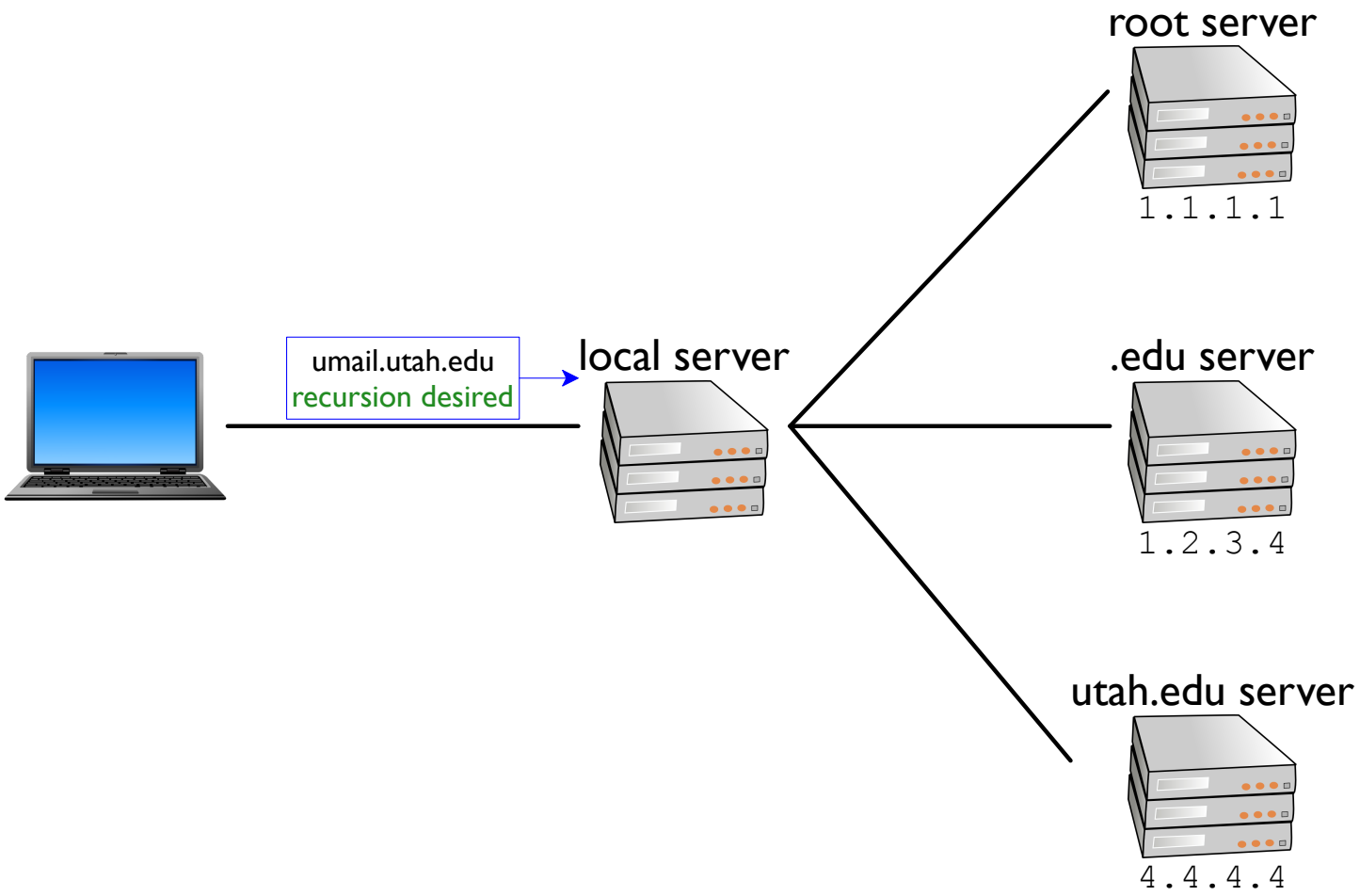
Name Resolution



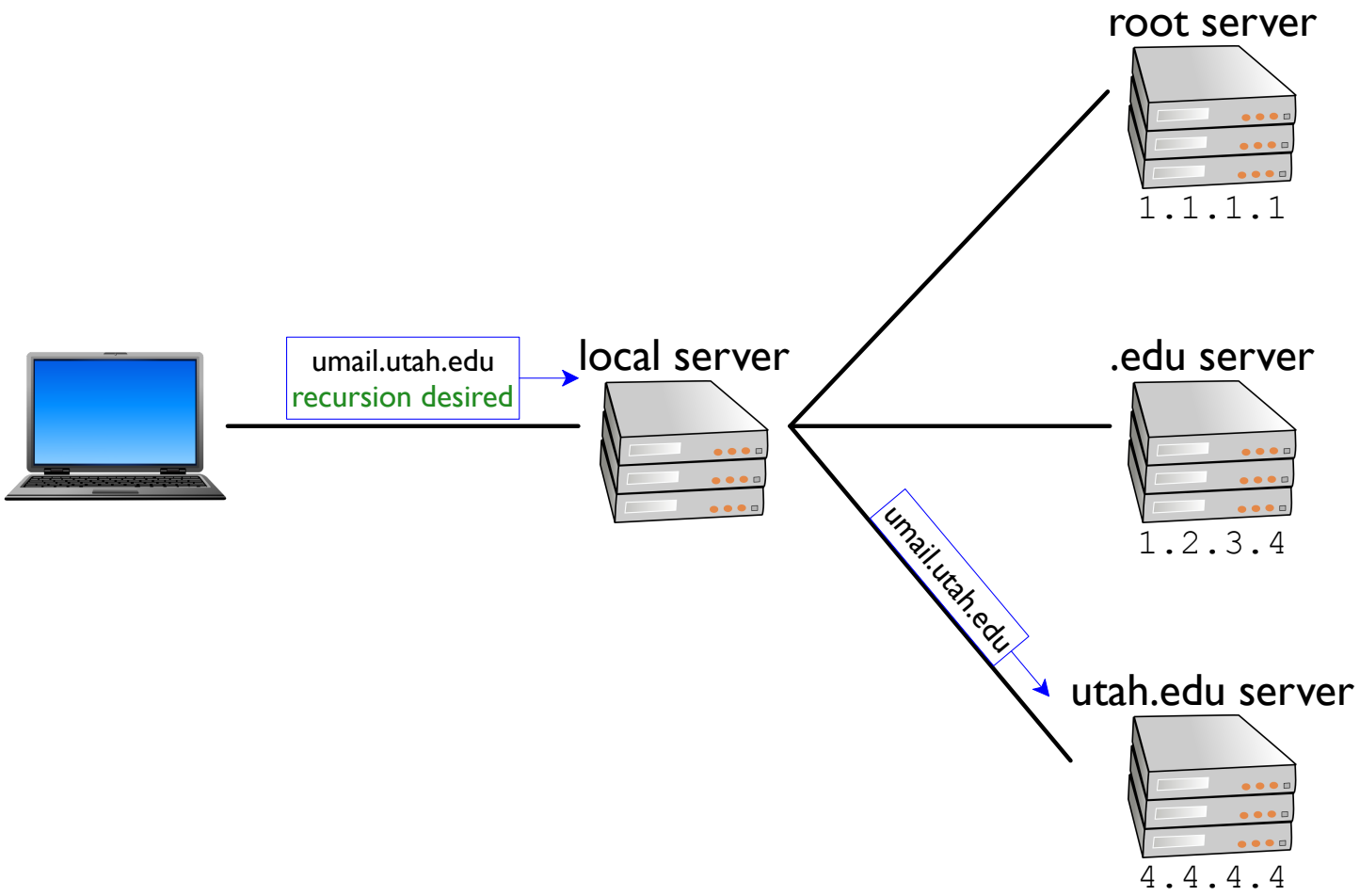
Name Resolution



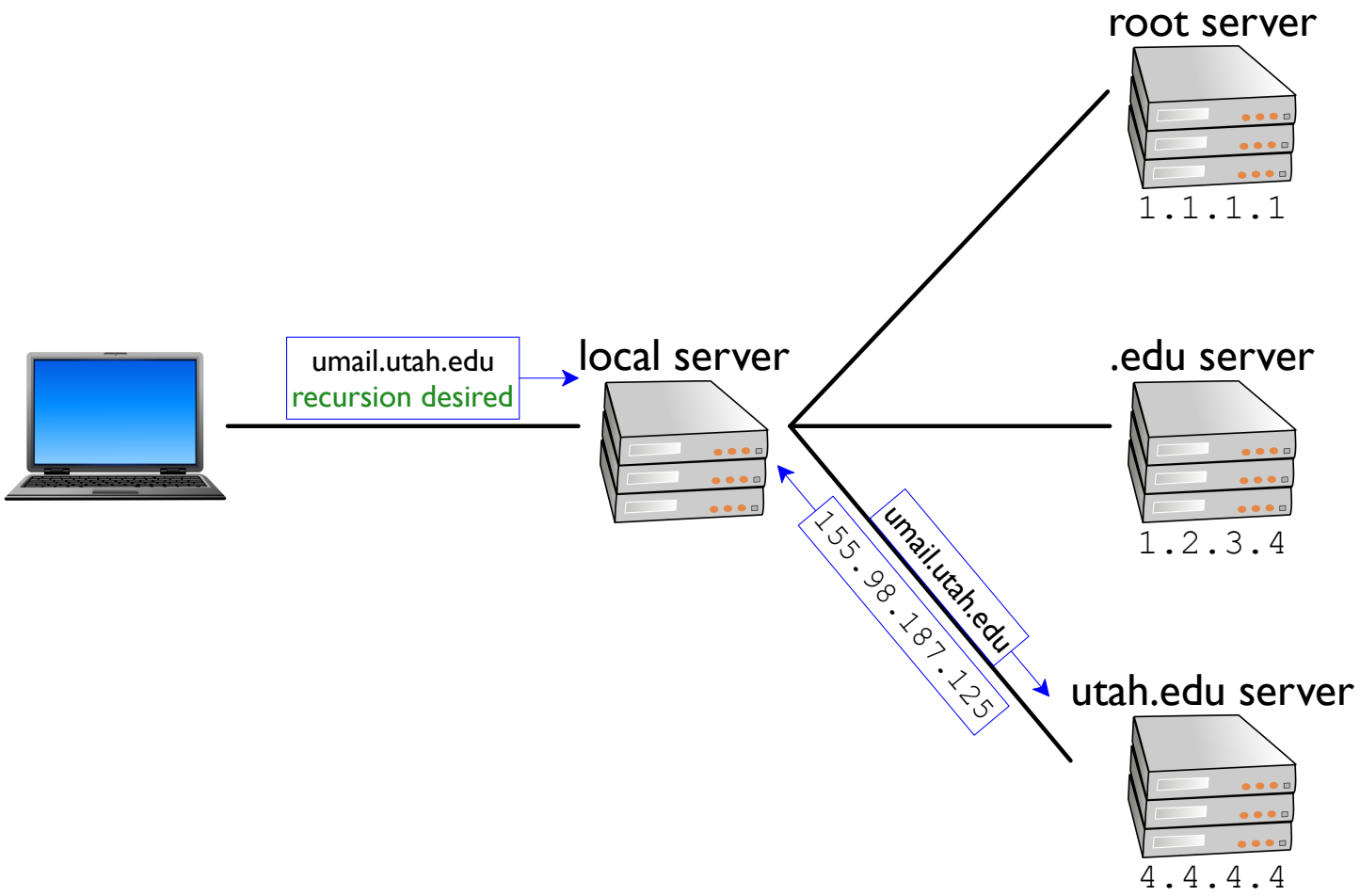
Name Resolution



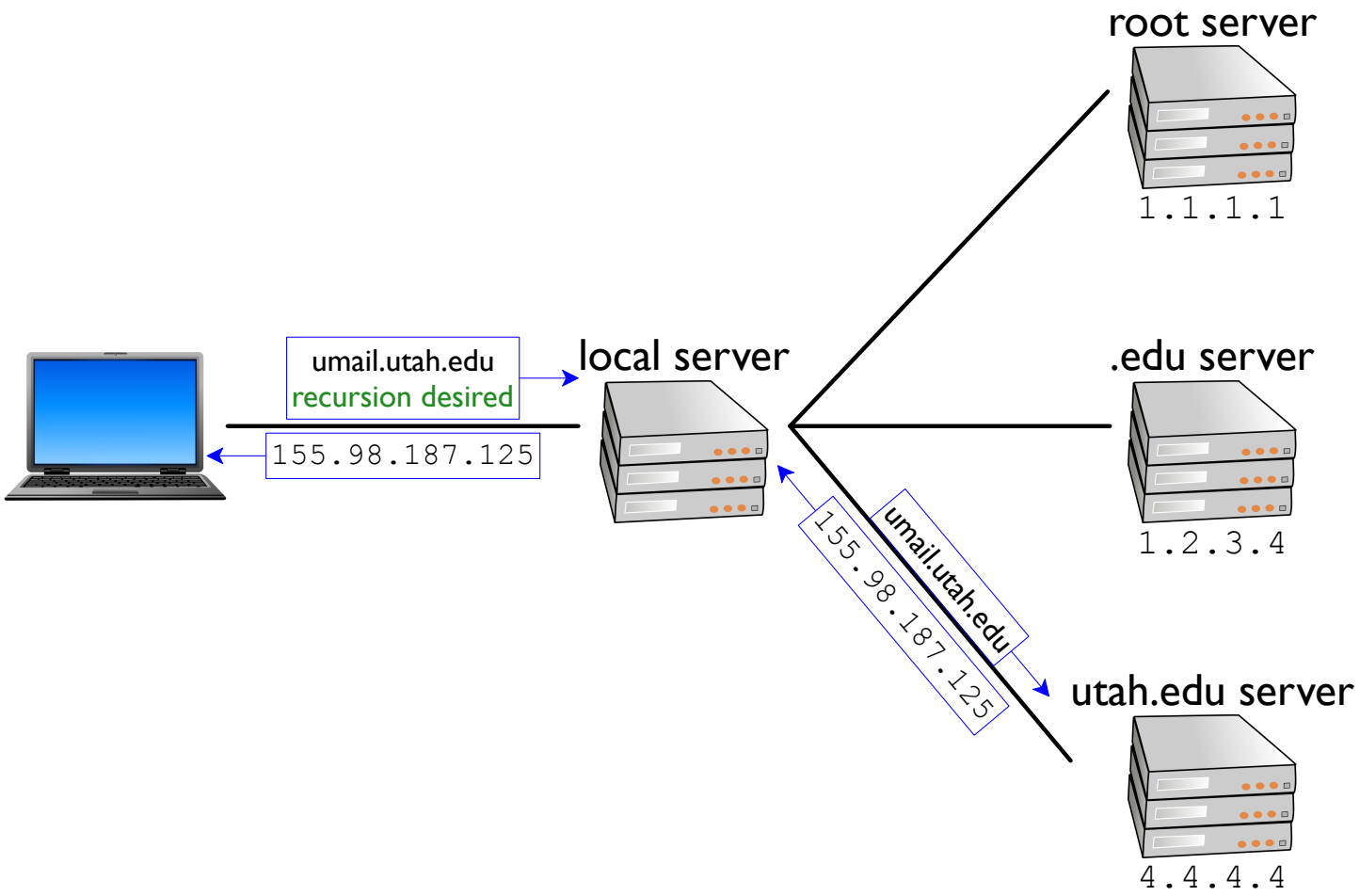
Name Resolution



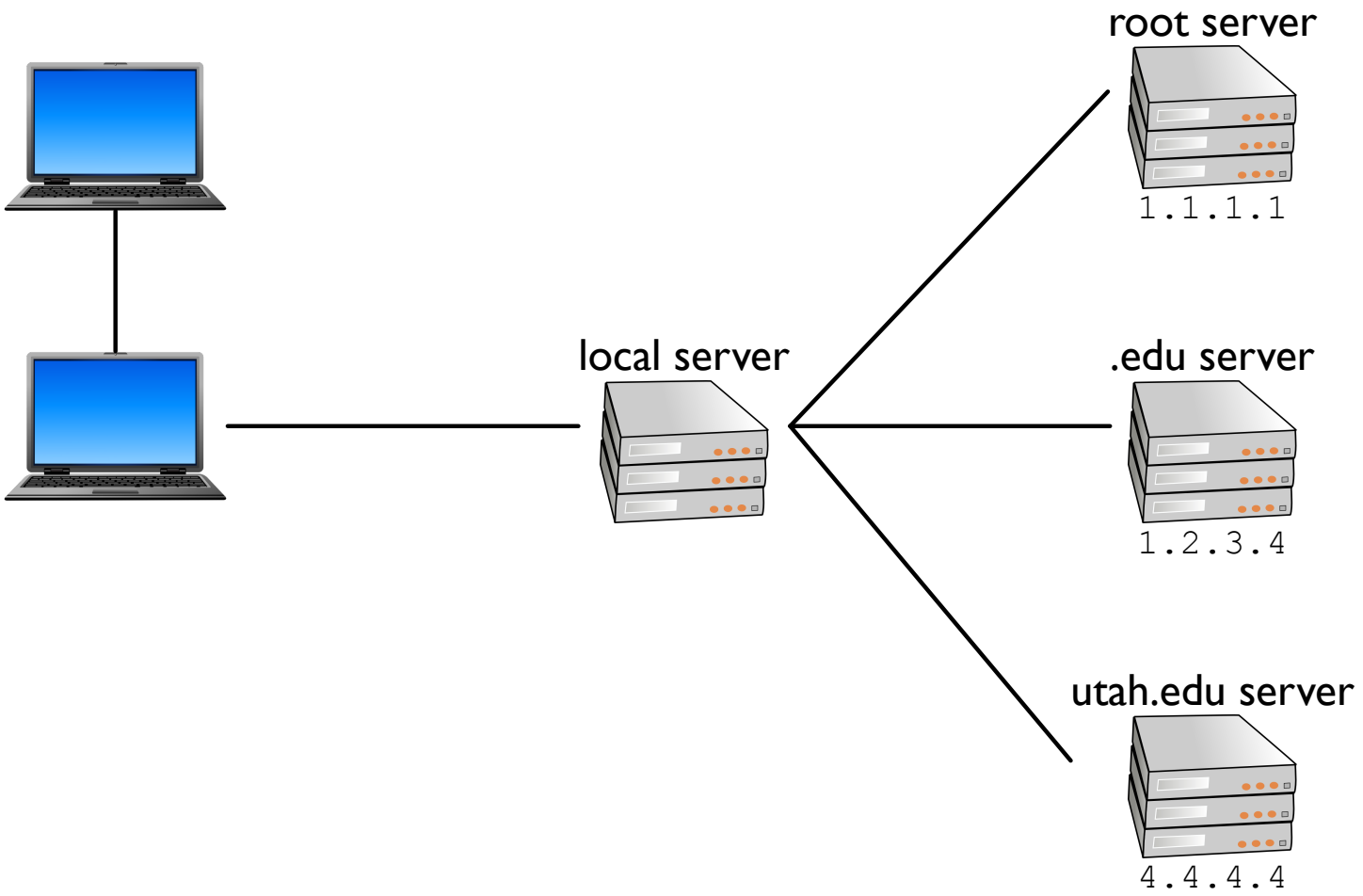
Name Resolution



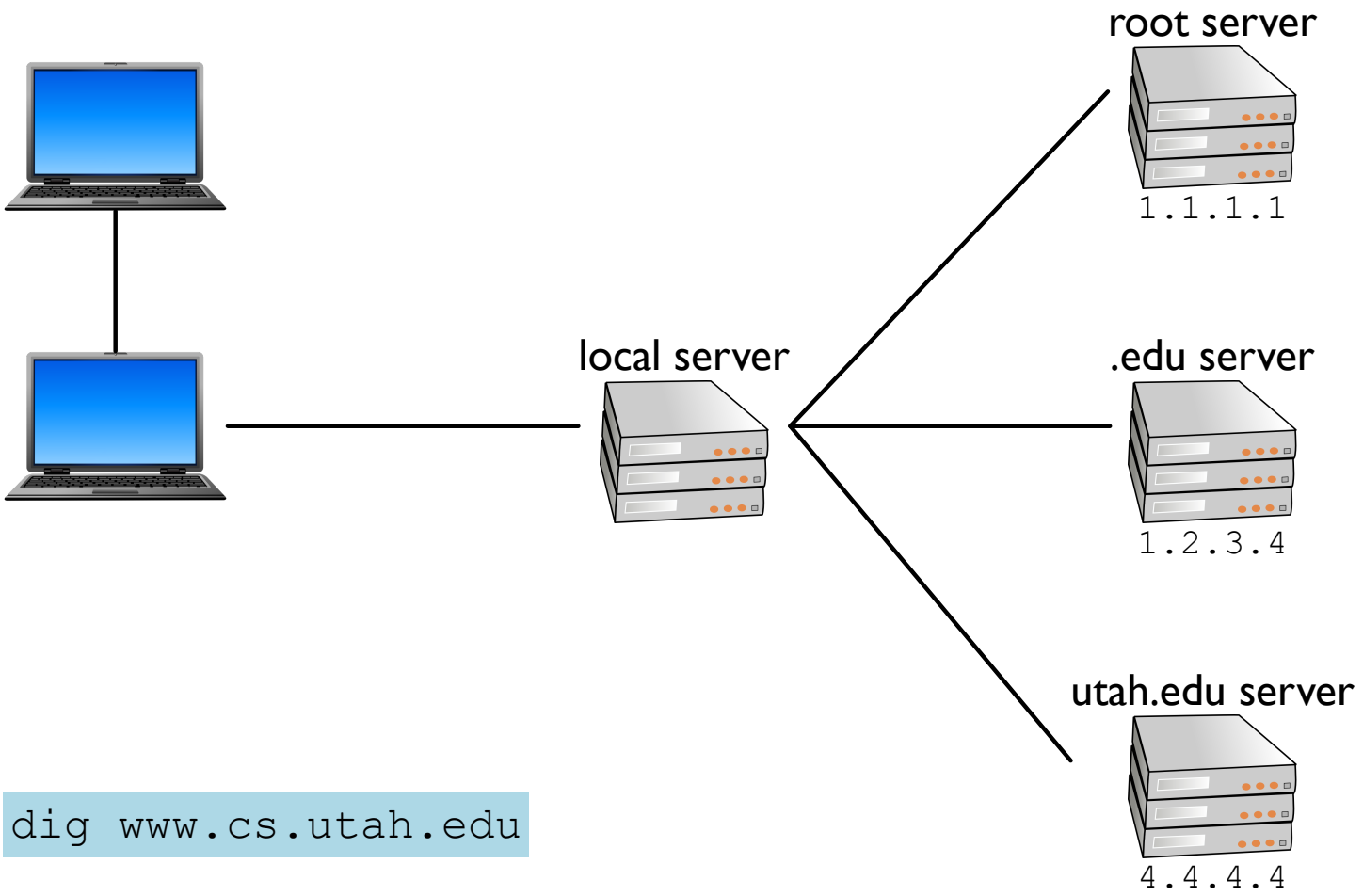
Name Resolution



Name Resolution

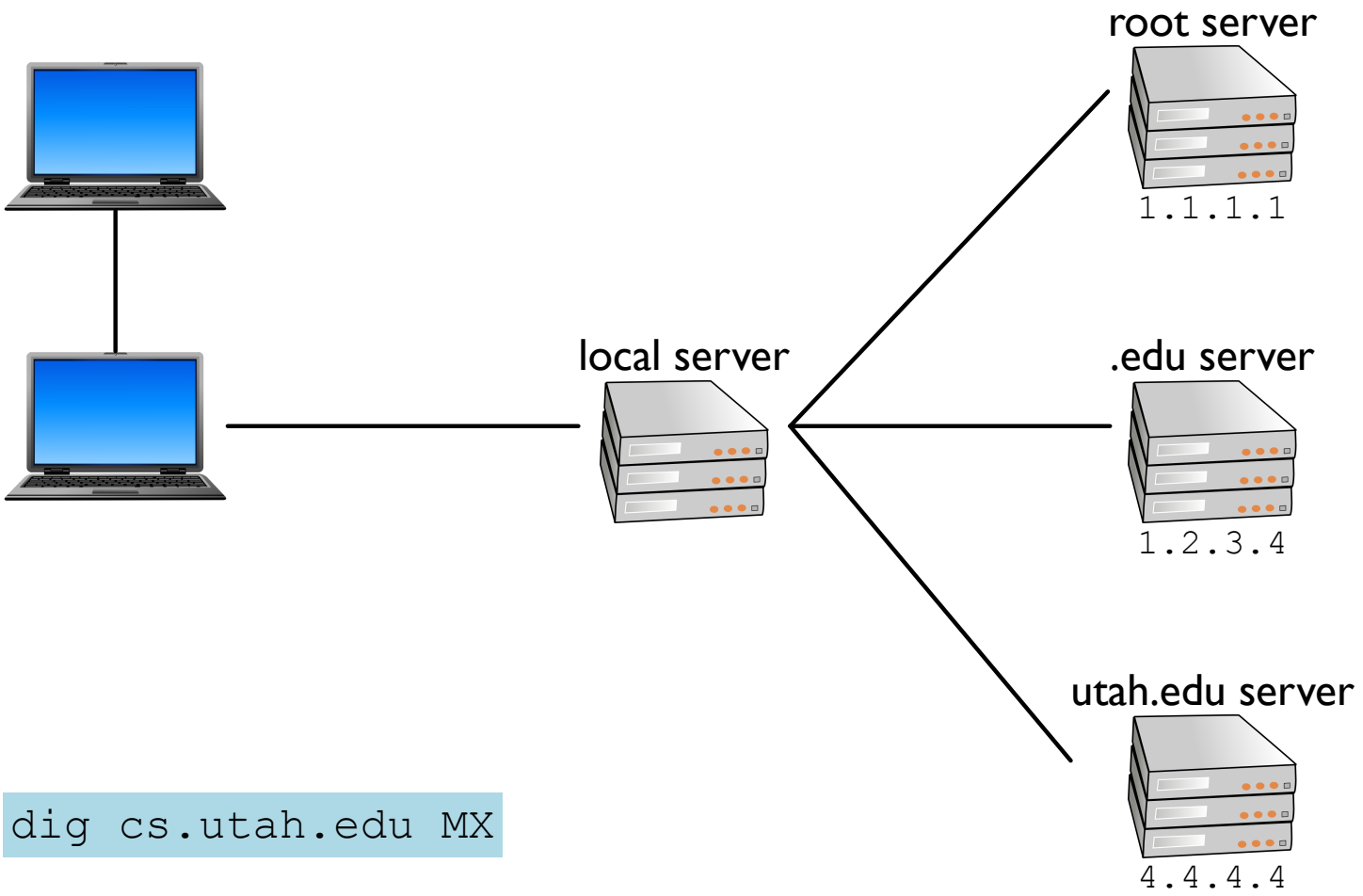


Name Resolution




```
dig www.cs.utah.edu
```

Name Resolution




```
dig cs.utah.edu MX
```

Summary

The **application layer** refers to a protocol implemented outside the operating system

- by a library
- by a program

Main building blocks provided by the next layer down:

- **TCP** — connection-oriented
- **UDP** — individual packets

Application-layer examples: HTTP, SMTP, **DNS**