layers

application	HTTP, SSH, SMTP, …	application-defined meanings
transport	TCP, UDP, …	reach correct program, reliablity/streams
network	IPv4, IPv6, …	reach correct machine (across networks)
link	Ethernet, Wi-Fi, …	coordinate shared wire/radio
physical	…	encode bits for wire/radio

names and addresses

name	address
logical identifier	location/how to locate
variable `counter`	memory address `0x7FFF9430`
DNS name `www.virginia.edu`	IPv4 address `128.143.22.36`
DNS name `mail.google.com`	IPv4 address `216.58.217.69`
DNS name `mail.google.com`	IPv6 address `2607:f8b0:4004:80b::2005`
DNS name `reiss-t3620.cs.virginia.edu`	IPv4 address `128.143.67.91`
DNS name `reiss-t3620.cs.virginia.edu`	MAC address `18:66:da:2e:7f:da`
service name `https`	port number `443`
service name `ssh`	port number `22`

layers

application	HTTP, SSH, SMTP, …	application-defined meanings
transport	TCP, UDP, …	reach correct program, reliablity/streams
network	IPv4, IPv6, …	reach correct machine (across networks)
link	Ethernet, Wi-Fi, …	coordinate shared wire/radio
physical	…	encode bits for wire/radio

port numbers

we run multiple programs on a machine

- IP addresses identifying machine — not enough

port numbers

- we run multiple programs on a machine
  - IP addresses identifying machine — not enough
- so, add 16-bit *port numbers*
  - think: multiple PO boxes at address

port numbers

we run multiple programs on a machine

- IP addresses identifying machine — not enough

so, add 16-bit *port numbers*

- think: multiple PO boxes at address

0–49151: typically assigned for particular services

- 80 = http, 443 = https, 22 = ssh, …

49152–65535: allocated on demand

- default "return address" for client connecting to server

UDP v TCP

UDP: messages sent to program, but no reliablity/streams

- get assigned port number
- SOCK_DGRAM with socket() instead of SOCK_STREAM
- can sendto()/recvfrom() multiple other programs with one socket
  - (but don't have to)
- send messages which are limited in size, unreliable

TCP: stream to other program

- need to bind() + listen() + accept() or connect() to setup connection
- one socket per connection
- read/write bytes — divided into messages automatically
- reliable — acknowledgments/resending handled for you

UDP sockets on IPv4

```
int fd = socket(AF_INET, SOCK_DGRAM, 0);
struct sockaddr_in my_addr= ...;
bind(fd, &my_addr, sizeof(my_addr))
...
struct sockaddr_in to_addr = ...;
sendto(fd, data, data_size, 0 /* flags */,
    &to_addr, sizeof(to_addr));
struct sockaddr_in from_addr = ...;
recvfrom(fd, &buffer[0], buffer_size, 0,
    &from_addr, sizeof(from_addr));
...
/* or connect() to set default sendto address
```

connections in TCP/IP

connection identified by *5-tuple*

  used by OS to lookup "where is the socket?"

(protocol=TCP/UDP, local IP addr., local port, remote IP addr., remote port)

local IP address, port number can be set with `bind()` function

  typically always done for servers, not done for clients

  system will choose default if you don't

connections on my desktop

```
cr4bd@reiss-t3620>/u/cr4bd
$ netstat --inet --inet6 --numeric
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 128.143.67.91:49202     128.143.63.34:22        ESTABLISHI
tcp        0      0 128.143.67.91:803       128.143.67.236:2049     ESTABLISHI
tcp        0      0 128.143.67.91:50292     128.143.67.226:22       TIME_WAIT
tcp        0      0 128.143.67.91:54722     128.143.67.236:2049     TIME_WAIT
tcp        0      0 128.143.67.91:52002     128.143.67.236:111      TIME_WAIT
tcp        0      0 128.143.67.91:732       128.143.67.236:63439    TIME_WAIT
tcp        0      0 128.143.67.91:40664     128.143.67.236:2049     TIME_WAIT
tcp        0      0 128.143.67.91:54098     128.143.67.236:111      TIME_WAIT
tcp        0      0 128.143.67.91:49302     128.143.67.236:63439    TIME_WAIT
tcp        0      0 128.143.67.91:50236     128.143.67.236:111      TIME_WAIT
tcp        0      0 128.143.67.91:22        172.27.98.20:49566      ESTABLISHI
tcp        0      0 128.143.67.91:51000     128.143.67.236:111      TIME_WAIT
tcp        0      0 127.0.0.1:50438         127.0.0.1:631           ESTABLISHI
tcp        0      0 127.0.0.1:631           127.0.0.1:50438         ESTABLISHI
```

non-connection sockets

TCP servers waiting for connections +
UDP sockets with no particular remote host

Linux: OS keeps 5-tuple with “wildcard” remote address

“listening” sockets on my desktop

```
cr4bd@reiss-t3620>/u/cr4bd
$ netstat --inet --inet6 --numeric --listen
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:38537         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:36777         0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:41099           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:45291           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:51949         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:41071         0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:32881         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:38673         0.0.0.0:*               LISTEN
....
tcp6       0      0 :::42689                :::*                    LISTEN
udp        0      0 128.143.67.91:60001     0.0.0.0:*
udp        0      0 128.143.67.91:60002     0.0.0.0:*
...
udp6       0      0 :::59938                :::*
```

TCP state machine

TIME_WAIT, ESTABLISHED, ...?

OS tracks "state" of TCP connection

- am I just starting the connection?
- is other end ready to get data?
- am I trying to close the connection?
- do I need to resend something?

standardized set of state names

TIME_WAIT

remember delayed messages?

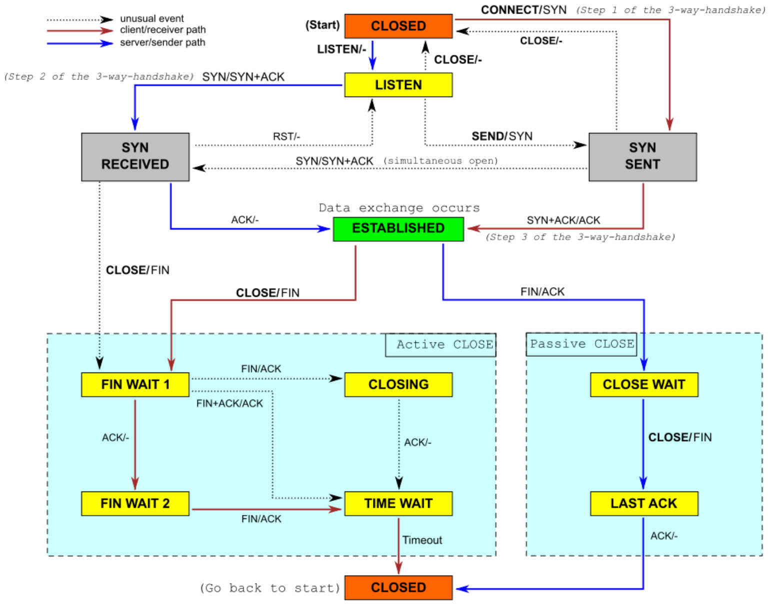
problem for TCP ports

if I reuse port number, I can get message from old connection

solution: TIME_WAIT to make sure connection really done

- done after sending last message in connection

TCP state machine picture

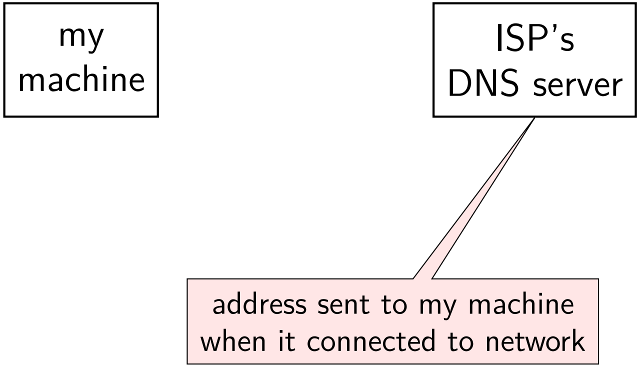


via Wikimedia/User:Scil100; CC-BY-SA

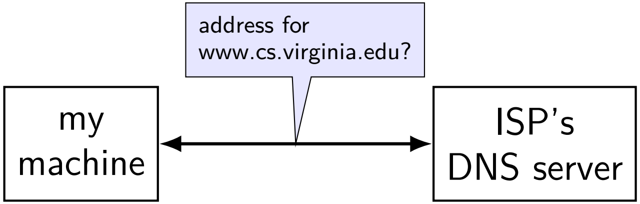
names and addresses

name	address
logical identifier	location/how to locate
variable `counter`	memory address `0x7FFF9430`
DNS name `www.virginia.edu`	IPv4 address `128.143.22.36`
DNS name `mail.google.com`	IPv4 address `216.58.217.69`
DNS name `mail.google.com`	IPv6 address `2607:f8b0:4004:80b::2005`
DNS name `reiss-t3620.cs.virginia.edu`	IPv4 address `128.143.67.91`
DNS name `reiss-t3620.cs.virginia.edu`	MAC address `18:66:da:2e:7f:da`
service name `https`	port number `443`
service name `ssh`	port number `22`

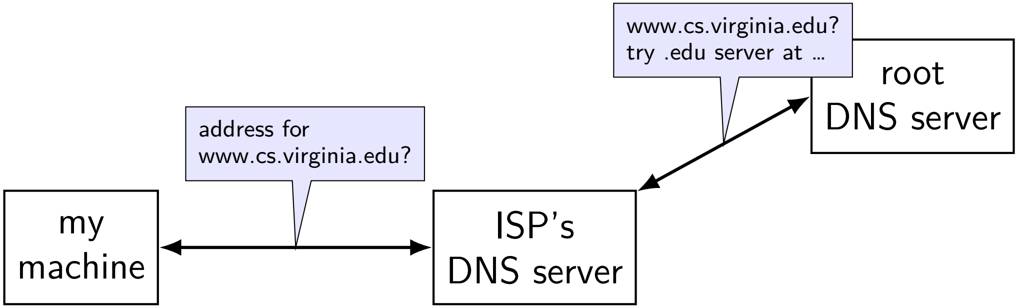
DNS: distributed database



DNS: distributed database

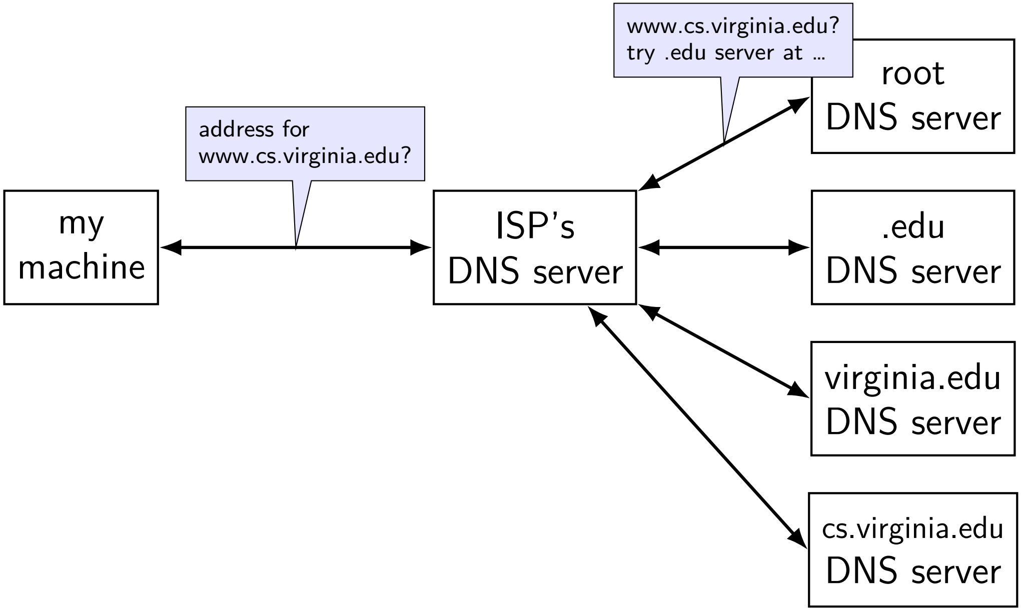


DNS: distributed database

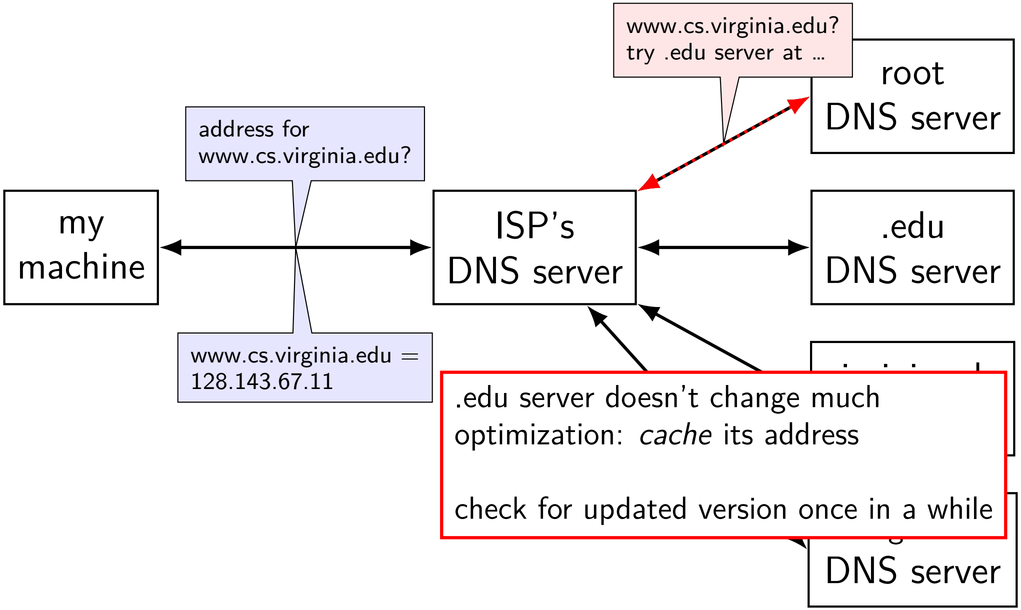


address for www.cs.virginia.edu?
www.cs.virginia.edu? try .edu server at …
my machine
ISP's DNS server
root DNS server

DNS: distributed database



DNS: distributed database



.edu server doesn't change much
optimization: *cache* its address

check for updated version once in a while

querying the root

```
$ dig @a.root-servers.net www.cs.virginia.edu
...
edu.                        172800      IN       NS       b.edu-servers.net.
edu.                        172800      IN       NS       f.edu-servers.net.
edu.                        172800      IN       NS       i.edu-servers.net.
edu.                        172800      IN       NS       a.edu-servers.net.
...
b.edu-servers.net.        172800      IN       A       192.33.14.30
b.edu-servers.net.        172800      IN       AAAA       2001:503:231d::2:30
f.edu-servers.net.        172800      IN       A       192.35.51.30
f.edu-servers.net.        172800      IN       AAAA       2001:503:d414::30
...
```

querying the edu

```
$ dig @b.edu-servers.net www.cs.virginia.edu
...
;; AUTHORITY SECTION:
virginia.edu.                172800      IN        NS        nom.virginia.edu.
virginia.edu.                172800      IN        NS        uvaarpa.virginia.edu.
virginia.edu.                172800      IN        NS        eip-01-aws.net.virginia.edu.

;; ADDITIONAL SECTION:
nom.virginia.edu.        172800      IN        A        128.143.107.101
uvaarpa.virginia.edu.        172800      IN        A        128.143.107.117
eip-01-aws.net.virginia.edu. 172800 IN        A        44.234.207.10
```

querying virginia.edu

```
$ dig @nom.virginia.edu www.cs.virginia.edu
...
;; AUTHORITY SECTION:
cs.virginia.edu.        3600        IN        NS        coresrv01.cs.virginia.edu.

;; ADDITIONAL SECTION:
coresrv01.cs.virginia.edu. 3600        IN        A        128.143.67.11
```

querying cs.virginia.edu

```
$ dig @coresrv01.cs.virginia.edu
...
;; ANSWER SECTION:
www.cs.Virginia.EDU.        172800      IN      A       128.143.67.11

;; AUTHORITY SECTION:
cs.Virginia.EDU.        172800      IN      NS      coresrv01.cs.Virginia.EDU.
...
```

querying typical ISP’s resolver

```
$ dig www.cs.virginia.edu
...
;; ANSWER SECTION:
www.cs.Virginia.EDU.          7183        IN        A        128.143.67.11
..
```

cached response

valid for 7183 more seconds

after that everyone needs to check again

names and addresses

name	address
logical identifier	location/how to locate
variable `counter`	memory address `0x7FFF9430`
DNS name `www.virginia.edu`	IPv4 address `128.143.22.36`
DNS name `mail.google.com`	IPv4 address `216.58.217.69`
DNS name `mail.google.com`	IPv6 address `2607:f8b0:4004:80b::2005`
DNS name `reiss-t3620.cs.virginia.edu`	IPv4 address `128.143.67.91`
DNS name `reiss-t3620.cs.virginia.edu`	MAC address `18:66:da:2e:7f:da`
service name `https`	port number `443`
service name `ssh`	port number `22`

two types of addresses?

MAC addreses: on link layer

IP addresses: on network layer

how do we know which MAC address to use?

a table on my desktop

my desktop:

```
$ arp -an
? (128.143.67.140) at 3c:e1:a1:18:bd:5f [ether] on enp0s31f6
? (128.143.67.236) at <incomplete> on enp0s31f6
? (128.143.67.11) at 30:e1:71:5f:39:10 [ether] on enp0s31f6
? (128.143.67.92) at <incomplete> on enp0s31f6
? (128.143.67.5) at d4:be:d9:b0:99:d1 [ether] on enp0s31f6
```

...

how is that table made?

ask machines on local network (same switch)

“Who has 128.148.67.140”

the correct one replies

what about non-local machines?

when configuring network specify:

range of addresses to expect on local network

- 128.148.67.0-128.148.67.255 on my desktop
- “netmask”

gateway machine to send to for things outside my local network

- 128.143.67.1 on my desktop
- my desktop looks up the corresponding MAC address

routes on my desktop

```
$ /sbin/route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         128.143.67.1    0.0.0.0         UG    100    0        0 enp0s31f6
128.143.67.0    0.0.0.0         255.255.255.0   U     100    0        0 enp0s31f6
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s31f6
```

URL / URIs

Uniform Resource Locators (URL)

tells how to find “resource” on network

Unifrom Resources Identifiers

superset of URLs

URI examples

```
https://kytos02.cs.virginia.edu:443/cs3130-spring2023/
                quizzes/quiz.php?qid=02#q2

https://kytos02.cs.virginia.edu/cs3130-spring2023/
                quizzes/quiz.php?qid=02

https://www.cs.virginia.edu/

sftp://cr4bd@portal.cs.virginia.edu/u/cr4bd/file.txt

tel:+1-434-982-2200
```

URI generally

`scheme://authority/path?query#fragment`

scheme: — what protocol

//authority/

authoirty = user@host:port OR host:port OR user@host OR host

path

which resource

?query — usually key/value pairs

#fragment — place in resource

most components (sometimes) optional

URLs and HTTP (1)

```
http://www.foo.com:80/foo/bar?quux#q1
```

lookup IP address of www.foo.com

connect via TCP to port 80:

```
GET /foo/bar?quux HTTP/1.1
Host: www.foo.com:80
```

exercise: why include the Host there?

autoconfiguration

problem: how does my machine get IP address

otherwise:

- have sysadmin type one in?
- just choose one?
- ask someone on local network to assign it

autoconfiguration

problem: how does my machine get IP address

otherwise:

- have sysadmin type one in?
- just choose one?
- ask someone on local network to assign it

DHCP high-level

protocol done over UDP

but since we don't have IP address yet, use `0.0.0.0`

and since we don't know server address, use `255.255.255.255`
= "everyone on the local network"

local server replies to request with address + time limit

firewalls

don't want to expose network service to everyone?

solutions:

- service picky about who it accepts connections from
- filters in OS on machine with services
- filters on router

later two called "firewalls"

firewall rules examples?

ALLOW tcp port 443 (https) FROM everyone

ALLOW tcp port 22 (ssh) FROM my desktop's IP address

BLOCK tcp port 22 (ssh) FROM everyone else

ALLOW from address X to address Y

...

spoofing

if I only allow connections from my desktop's IP addresses,
how would you attack this?

hint: how do we know what address messages come from?

backup slides

backup slides