Network Fundamentals

From Zero to HTTP

Original content : Tom nom nom
Translate to Persian : Chaos Nexus

فهرست

یک به یک (One To One)

- دو دستگاه می‌توانند با یکدیگر صحبت کنند
- هر دستگاه دارای یک رابط شبکه است
- رابط‌های شبکه[1] می‌توانند مستقیماً از طریق کابل شبکه به یکدیگر متصل شوند
- هر رابط شبکه دارای یک آدرس کنترل دسترسی به رسانه (MAC) (همچنین به عنوان آدرس سخت‌افزاری نیز شناخته می‌شود) است
- آدرس‌های MAC به این شکل هستند: 50:46:5d:54:94:23
- آدرس‌های MAC به صورت جهانی منحصر به فرد هستند (حداقل در تئوری)
- داده‌ها در قطعاتی به نام «فریم» ارسال می‌شوند
- هر فریم دارای یک آدرس MAC منبع و مقصد است

[1] Network interfaces

چگونه آدرس MAC مقصد را می‌فهمد؟

- آنها[۲] این کار را نمی‌کنند!
- آنها آدرس IP را می‌دانند (چون شما آن را به آنها می‌گویید)
- یک آدرس IPv4 به این شکل است: 192.168.0.1
- در عین حال IPv6 هم وجود دارد، اما ما در اینجا به آن نمی‌پردازیم
- یک ماشین می‌تواند از کل شبکه بپرسد که چه کسی یک IP خاص دارد.
- ماشین‌ها فریم‌هایی را که MAC آنها به عنوان مقصد نیست، نادیده می‌گیرند.
- اما یک MAC خاص «هرکسی» یا «پخش» وجود دارد: ff:ff:ff:ff:ff:ff

[2] منظور از آنها ماشین ها یا کامپیوتر ها در ارتباط با یکدیگر است

دریافت اطلاعات رابط شبکه شما

در لینوکس

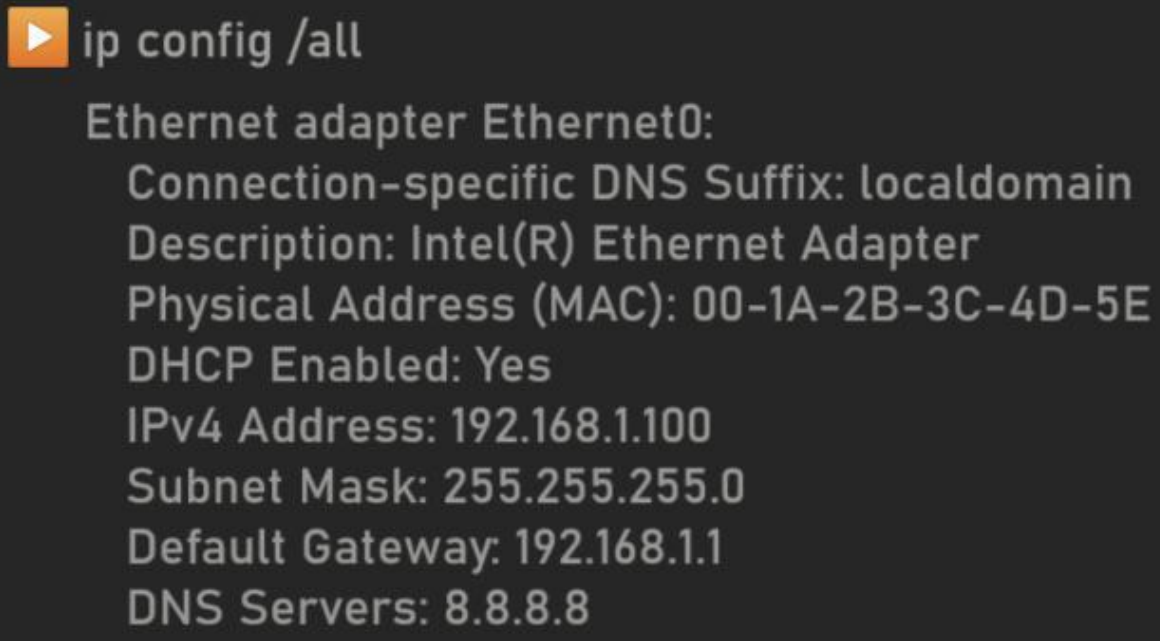
▶️ ifconfig enp3s0

```
enp3s0    Link encap:Ethernet  HWaddr 50:46:5d:54:94:23
          inet addr:192.168.1.30  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::5246:5dff:fe54:9423/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:48241295 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24083899 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:49741929087 (49.7 GB)  TX bytes:2925004440 (2.9 GB)
```

▶️ ip a show dev enp3s0

```
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 50:46:5d:54:94:23 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.30/24 brd 192.168.1.255 scope global enp3s0
       valid_lft forever preferred_lft forever
    inet6 fe80::5246:5dff:fe54:9423/64 scope link
       valid_lft forever preferred_lft forever
```

در ویندوز

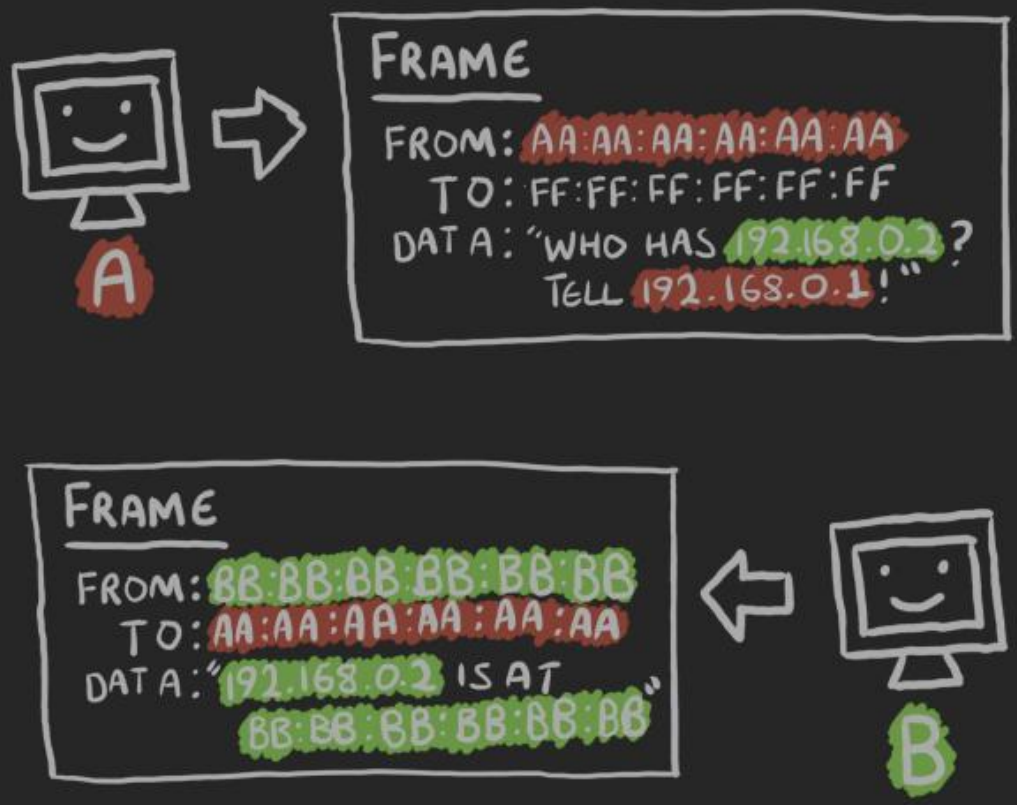


توجه داشته باشید که در سیستم شما این نتایج ممکن است متفاوت باشد

آیا کسی آنجاست؟

- ماشین A می‌خواهد با ماشین B صحبت کند
- ماشین A دارای IP 192.168.0.1 و MAC aa:aa:aa:aa:aa:aa است
- ماشین B دارای IP 192.168.0.2 و MAC bb:bb:bb:bb:bb:bb است
- ماشین A فریمی مانند این ارسال می‌کند:
  - MAC منبع: aa:aa:aa:aa:aa:aa
  - MAC مقصد: ff:ff:ff:ff:ff:ff
  - داده: "چه کسی 192.168.0.2 را دارد؟ به 192.168.0.1 بگویید!"
- ماشین B با فریمی مانند این پاسخ می‌دهد:
  - مک مبدا: bb:bb:bb:bb:bb:bb
  - مک مقصد: aa:aa:aa:aa:aa:aa
  - داده: "192.168.0.2 در bb:bb:bb:bb:bb:bb است!"
- هر دو ماشین IPها و MACهای مربوطه را در حافظه پنهان پروتکل تفکیک آدرس (ARP) خود برای استفاده‌های بعدی ذخیره می‌کنند.

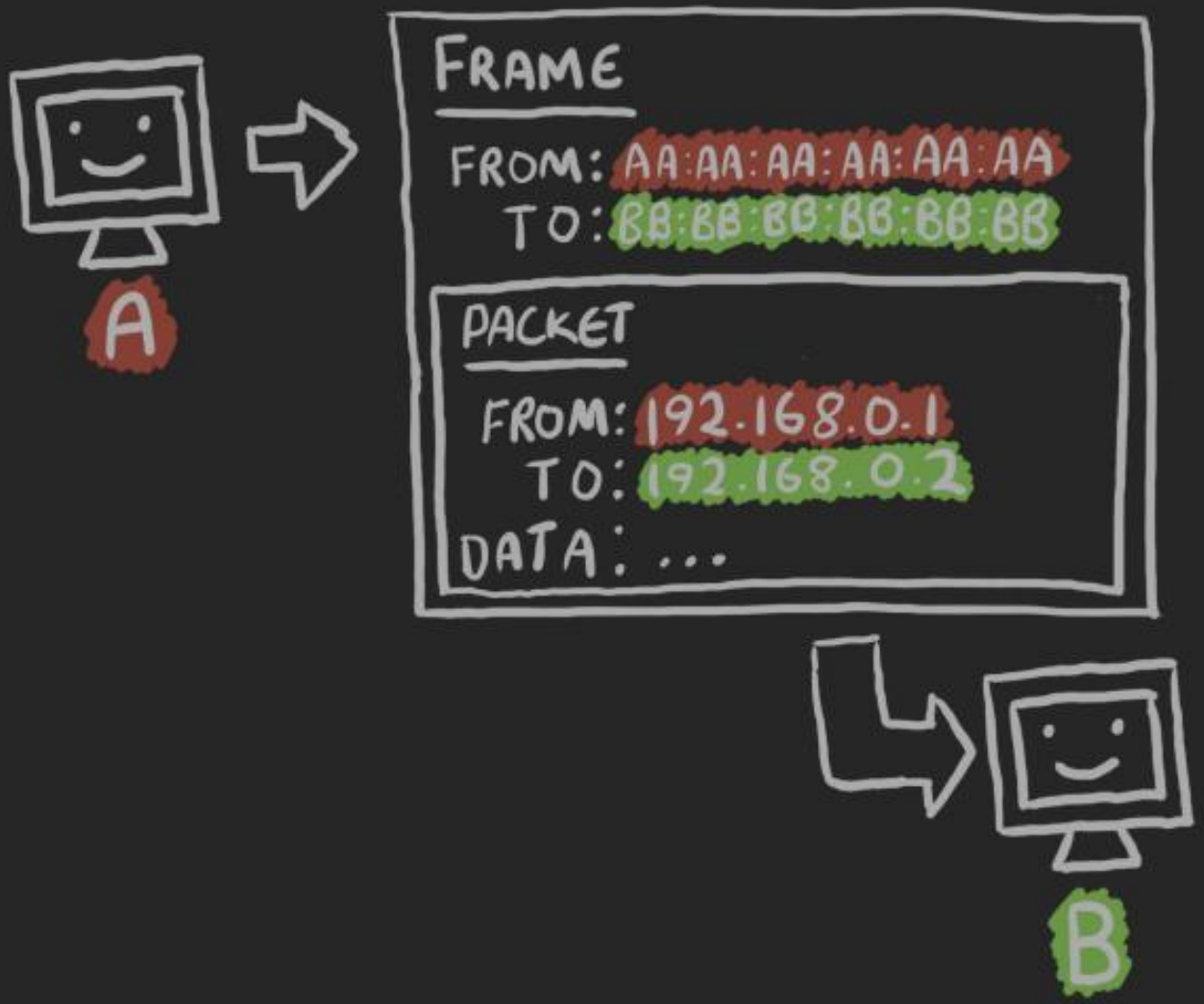
پروتکل تحلیل آدرس (Address Resolution Protocol)



پیام بعدی

- ماشین A می‌خواهد دوباره با ماشین B صحبت کند
- این بار ماشین A می‌تواند آدرس MAC را در حافظه پنهان ARP خود پیدا کند
- ماشین A فریمی ارسال می‌کند که به این شکل است:
  - مک مبدا: aa:aa:aa:aa:aa:aa:aa
  - مک مقصد: bb:bb:bb:bb:bb:bb
  - داده: ...
- درون داده‌ها یک «بسته» IP وجود دارد که به این شکل است:
  - آی‌پی مبدا: 192.168.0.1
  - آی‌پی مقصد: 192.168.0.2
  - داده: ...
- داده‌های اسلاید آخر در واقع یک بسته ARP بودند.
- چرا مک و آی‌پی؟
  - ماشین‌ها می‌توانند بیش از یک آدرس IP داشته باشند و دلایل دیگری نیز وجود دارد

حافظه نهان ARP (The ARP Cache)



مشاهده حافظه نهان ARP شما

در لینوکس

```
ip n
192.168.1.170 dev enp3s0 lladdr 00:17:88:49:a0:62 STALE
192.168.1.138 dev enp3s0 lladdr 94:44:44:ed:f5:c8 STALE
192.168.1.114 dev enp3s0 lladdr f4:5c:89:c1:ed:5f STALE
192.168.1.60  dev enp3s0 lladdr 00:18:a9:74:a5:88 STALE
192.168.1.1   dev enp3s0 lladdr 98:fc:11:85:74:6c REACHABLE
192.168.1.179 dev enp3s0 lladdr dc:3a:5e:5d:e0:9d STALE
192.168.1.163 dev enp3s0 lladdr 70:48:0f:c9:19:42 STALE
192.168.1.23  dev enp3s0 lladdr 38:ea:a7:a9:34:f3 STALE
192.168.1.134 dev enp3s0 lladdr 8c:f5:a3:30:af:a7 STALE
192.168.1.10  dev enp3s0 lladdr 44:d9:e7:62:ab:cc REACHABLE
```

در ویندوز

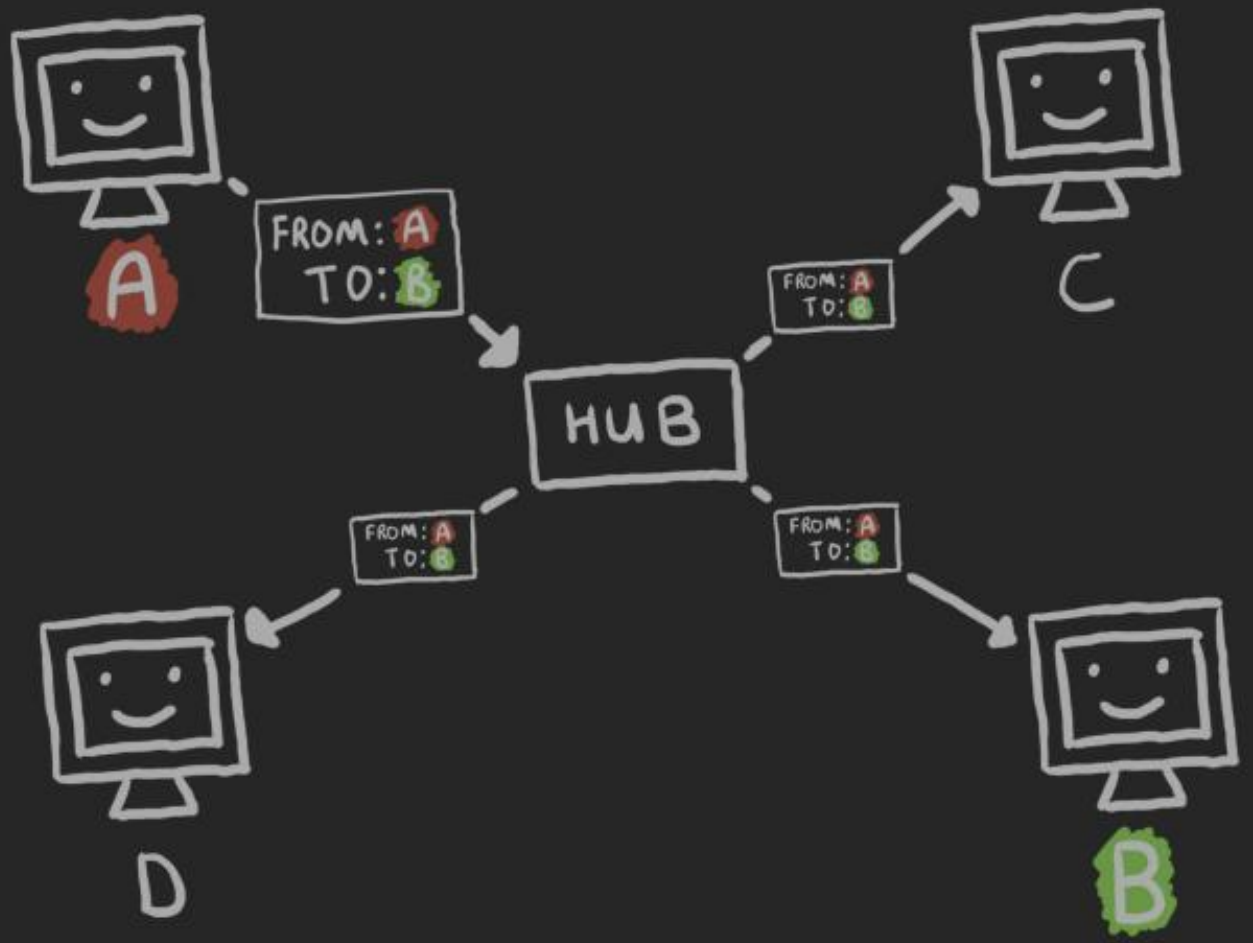
```
arp -a

Interface: 192.168.1.100 --- 0x10
  Internet Address      Physical Address      Type
  192.168.1.1           00-11-22-33-44-55     dynamic
  192.168.1.101         aa-bb-cc-dd-ee-ff     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
```


بیش از دو دستگاه

- بیش از دو ماشین می‌توانند به یک هاب متصل شوند
- هاب خیلی بی‌معنی است
- فقط هر چیزی را که دریافت می‌کند به تمام پورت‌ها ارسال می‌کند
- ماشین‌ها فریم‌هایی را که برای آنها در نظر گرفته نشده است نادیده می‌گیرند، بنابراین همه چیز (عمدتاً) خوب است
- هر چیزی که برای دو ماشین کار می‌کرد، دقیقاً به یک شکل کار می‌کند
- اما:
  - کند است (۱۰ مگابیت، اگر خوش شانس باشید ۱۰۰ مگابیت)
  - با تصادم مواجه می‌شوید (ماشین‌ها سعی می‌کنند با یکدیگر صحبت کنند)
  - می‌توانیم بهتر عمل کنیم

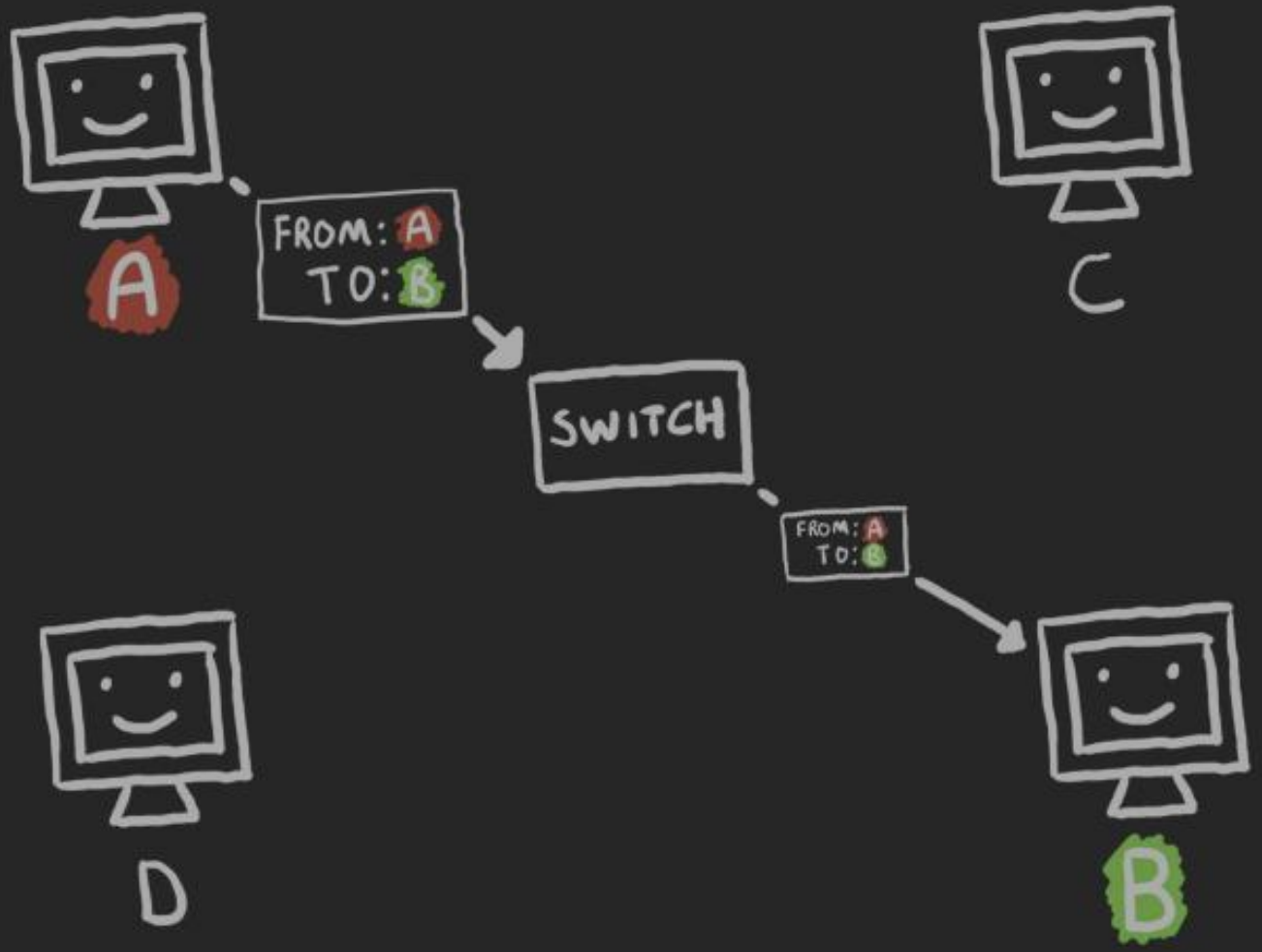
Hubs



Switching

- سوئیچ‌های شبکه هوشمندتر و کارآمدتر هستند
- سوئیچ‌ها MACهای منبعی را که در هر پورت دیده‌اند به خاطر می‌سپارند.
- فریم‌ها فقط به پورتی ارسال می‌شوند که MAC به آن متصل است.
- اگر سوئیچ نداند MAC کجاست: به همه پورت‌ها ارسال می‌کند.
  - هرگز نمی‌داند ff:ff:ff:ff:ff:ff کجاست، بنابراین همیشه به همه پورت‌ها ارسال می‌شود!
- تصادم کمتر!
- بسیار سریع‌تر!
  - سرعت 10 گیگابیت در شبکه‌های سوئیچینگ نسبتاً رایج است.

Switches



Subnets

- ماشین‌ها فقط می‌توانند بسته‌های IP را مستقیماً به ماشین‌های همان شبکه ارسال کنند.
- خب... چگونه یک شبکه (از نظر فنی یک زیرشبکه) را تعریف کنیم؟
- علاوه بر IP، هر ماشین یک ماسک زیرشبکه نیز دارد.
  - آنها به این شکل هستند: 255.255.255.0
- Subnet mask در ترکیب با IP منبع و مقصد استفاده می‌شود تا مشخص شود که آیا آنها در یک زیرشبکه هستند یا خیر.
- در واقع درک آن به صورت دودویی بسیار آسان‌تر است!

Subnet Masks

- دو ماشین در یک زیرشبکه هستند اگر بیت‌های موجود در IP آنها با بیت متناظر در subnet mask که ۱ است، مطابقت داشته باشد.

These two are on the *same* subnet:

```
Source:      192.168.0.1    11000000.10101000.00000000.00000001
Destination: 192.168.0.2    11000000.10101000.00000000.00000010
Subnet Mask: 255.255.255.0  11111111.11111111.11111111.00000000
```

These two are on *different* subnets:

```
Source:      192.168.0.1    11000000.10101000.00000000.00000001
Destination: 192.168.31.2   11000000.10101000.00011111.00000010
Subnet Mask: 255.255.255.0  11111111.11111111.11111111.00000000
```


نمادگذاری CIDR (CIDR Notation)

- مشخص کردن IP و ماسک زیرشبکه کمی خسته‌کننده می‌شود.
- می‌توانید به جای آن از نمادگذاری مسیریابی بین دامنه‌ای بدون کلاس استفاده کنید.
- تعداد ۱ها را در subnet mask بشمارید!

10.0.0.1/255.255.255.0

00001010.00000000.00000000.00000001/11111111.11111111.11111111.00000000

10.0.0.1/24

مسیریابی (Routing)

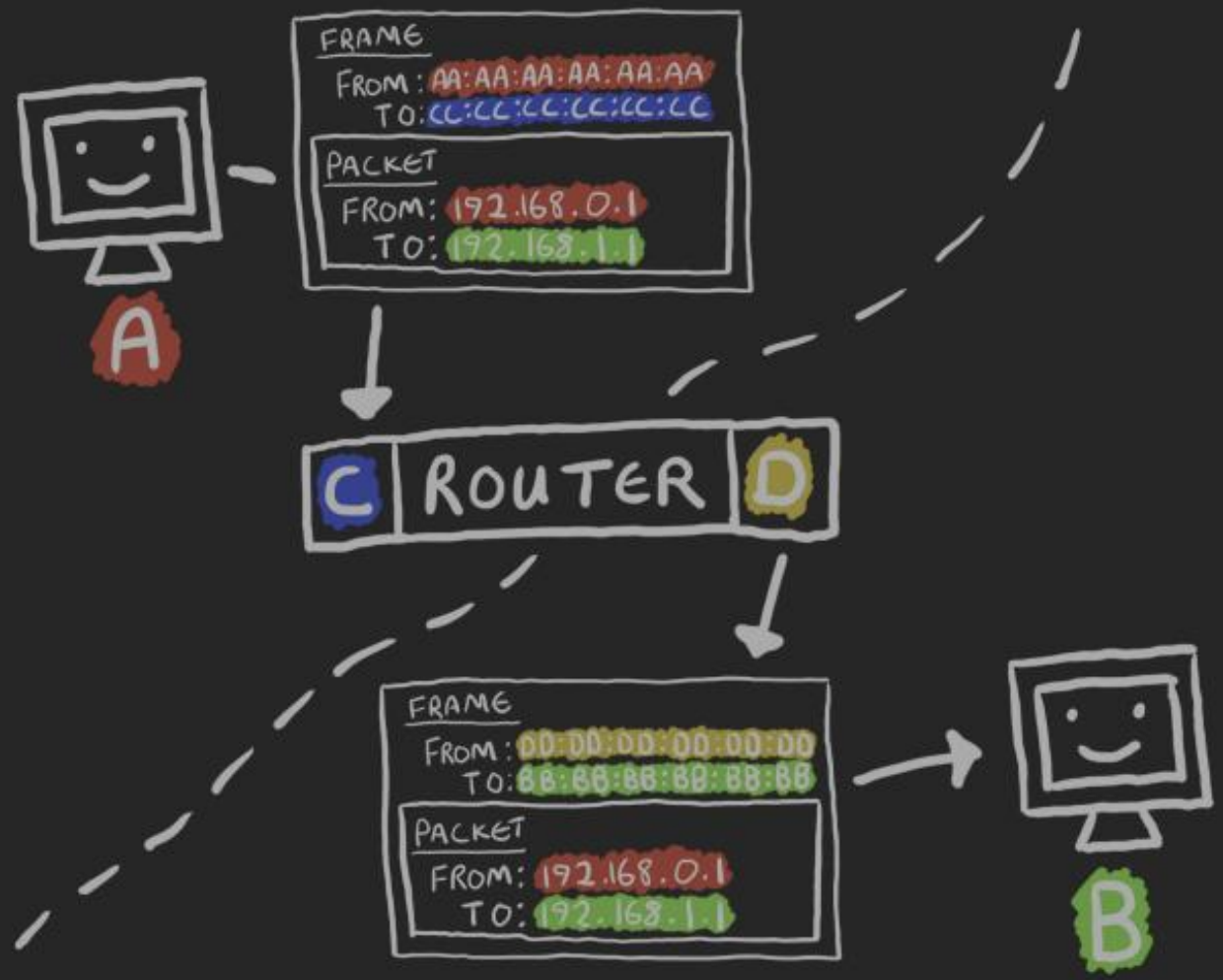
- برای ارسال یک بسته به دستگاهی در زیرشبکه دیگر، فریم به یک روتر ارسال می‌شود.
- یک روتر معمولاً بیش از یک رابط شبکه (و آدرس MAC) دارد.
- یک روتر همیشه بیش از یک آدرس IP دارد (حداقل یکی برای هر زیرشبکه)

- Machine A (subnet one):
  - MAC: aa:aa:aa:aa:aa:aa
  - IP: 192.168.0.1 / 255.255.255.0
- Machine B (subnet two):
  - MAC: bb:bb:bb:bb:bb:bb
  - IP: 192.168.1.1 / 255.255.255.0
- Router (both subnets):
  - MAC1: cc:cc:cc:cc:cc:cc
  - IP1: 192.168.0.254 / 255.255.255.0
  - MAC2: dd:dd:dd:dd:dd:dd
  - IP2: 192.168.1.254 / 255.255.255.0

مثالی از Hop

- ماشین A می‌خواهد با ماشین B صحبت کند، اما ماشین B در زیرشبکه متفاوتی قرار دارد.
- بنابراین، یک فریم با استفاده از MAC برای دروازه پیش‌فرض خود به عنوان مقصد ارسال می‌کند:
  - MAC منبع: aa:aa:aa:aa:aa:aa
  - MAC مقصد: cc:cc:cc:cc:cc:cc (اولین MAC روتر!)
  - IP منبع: 192.168.0.1
  - IP مقصد: 192.168.1.1 (IP دستگاه B!)
- روتر فریم را دریافت می‌کند و سپس موارد زیر را ارسال می‌کند:
  - MAC منبع: dd:dd:dd:dd:dd:dd (دومین MAC روتر)
  - MAC مقصد: bb:bb:bb:bb:bb:bb
  - IP منبع: 192.168.0.1
  - IP مقصد: 192.168.1.1
- روتر، MACهای مبدا و مقصد را تغییر داد و دستگاه B فریم را از روتر دریافت می‌کند :)

A Hop


FRAME
FROM: AA:AA:AA:AA:AA:AA
TO: CC:CC:CC:CC:CC:CC
PACKET
FROM: 192.168.0.1
TO: 192.168.1.1
A
C
ROUTER
D
FRAME
FROM: DD:DD:DD:DD:DD:DD
TO: BB:BB:BB:BB:BB:BB
PACKET
FROM: 192.168.0.1
TO: 192.168.1.1
B

انتخاب چندگانه

- ماشین A فریم را به عنوان آخرین راه چاره به دروازه پیش‌فرض خود ارسال کرد.
- ممکن است گزینه دیگری در جدول مسیریابی خود داشته باشد:

Network	Subnet Mask	Gateway
0.0.0.0	0.0.0.0	192.168.0.254
192.168.1.0	255.255.255.0	192.168.0.253
192.168.2.0	255.255.255.0	192.168.0.252

- با این جدول، MAC برای 192.168.0.253 مقصد خواهد بود.
- چندین شبکه متصل از طریق روترها چیزی را تشکیل می‌دهند که ما آن را اینترنت می‌نامیم :)

The OSI Model

#	Name	Unit	What?
7	Application	Data	HTTP, FTP etc
6	Presentation	Data	Encryption! TLS etc
5	Session	Data	PPTP, SOCKS
4	Transport	Segments	TCP, UDP
3	Network	Packets	IP and routing
2	Data-Link	Frames	MAC addresses and the like
1	Physical	Bits	Electricity on a wire

مجموعه پروتکل اینترنت (The Internet Protocol Suite)

- یک مدل جایگزین و بسیار ساده‌تر
- هنوز فقط یک مدل است؛ همه چیز به خوبی تعریف نشده است

#	Name	Unit	What?
4	Application	Data	HTTP, FTP etc
3	Transport	Segments	TCP, UDP
2	Internet	Packets	IP and routing
1	Link	Frames	MAC addresses and the like

کنترل حمل و نقل (Transport Control)

- تاکنون ما فقط به ارتباط یک طرفه پرداخته‌ایم.
- شبکه غیرقابل اعتماد است، اما ما به ارتباط قابل اعتماد نیاز داریم.
- چگونه متوجه می‌شوید که کسی نامه شما را دریافت کرده است؟
  - از آنها بخواهید که نامه‌ای برای شما ارسال کنند!
  - اگر پس از مدتی پاسخی دریافت نکردید، نامه دیگری ارسال کنید :)
- TCP قابلیت اطمینان را برای بسته‌های IP فراهم می‌کند.
- TCP پورت‌ها را اضافه می‌کند تا بتوانیم بیش از یک مکالمه بین دو IP داشته باشیم.
  - پورت‌ها فقط اعداد هستند. شما به یک پورت منبع و یک پورت مقصد نیاز دارید.
- اگر به قابلیت اطمینانی که TCP ارائه می‌دهد نیاز ندارید، می‌توانید از UDP استفاده کنید.

بیایید باهم با حالت TCP صحبت کنیم

Machine A		Machine B
Hey, can we talk?	→	...
...	←	Sure.
OK! Let's talk!	→	...
So, can you do this thing for me?	→	...
...	←	Yes, I hear you.
...	←	Here's the thing you wanted.
Got it!	→	...
...	←	I'm leaving.
Fine! Me too!	→	...
...	←	Good.

نسخه واقعی

192.168.0.1:56789		192.168.0.2:80
SYN	→	...
...	←	SYN, ACK
ACK	→	...
DATA	→	...
...	←	ACK
...	←	DATA
ACK	→	...
...	←	FIN
FIN, ACK	→	...
...	←	ACK

ارسال مجدد

- اگر فرستنده پس از مدتی ACK دریافت نکند، داده‌ها را دوباره ارسال خواهد کرد

192.168.0.1:56789		192.168.0.2:80
...handshake...		
DATA	→	...
...	←	ACK
...	←	DATA
...time passes...		
...	←	DATA
ACK	→	...
...termination...		

از چند لایه گذر میکنیم (حداقل برای OSI)[3]

- HTTP یک پروتکل لایه کاربرد است
- HTTP نسخه ۱.۱ فقط متن ساده است
  - آنقدر ساده که می‌توانید آن را با دست بنویسید!
- ممکن است مثلاً با TLS رمزگذاری شده باشد، اما فعلاً آن را نادیده می‌گیریم
- HTTP نسخه ۲ متن ساده نیست، اما آن را هم نادیده می‌گیریم
- وقتی در مورد یک پروتکل لایه کاربرد صحبت می‌کنیم، می‌توانیم (عمدتاً) لایه‌های پایین‌تر را نادیده بگیریم :)

[3] خودتان مطالعه کنید و یاد بگیرید !

بیایید باهم با حالت HTTP صحبت کنیم

192.168.0.1:56789		192.168.0.2:80
	...handshake...	

`GET /index.html HTTP/1.1`
`Host: example.com`
`Connection: close`
`User-Agent: slidedeck/0.3`
`Accept: */*`
	→	...

...	←	`HTTP/1.1 200 OK`
		`Content-Type: text/html`
		`Content-Length: 1337`
		
		`<!doctype html>`
		`<html>`
		`...`

	...termination...	

درخواست

- هر خط در درخواست توسط یک کاراکتر بازگشت به سطر و یک کاراکتر تغذیه سطر (دنباله CRLF) از هم جدا می‌شود.
- درخواست توسط دو دنباله CRLF خاتمه می‌یابد.
- هدرها به شکل کلید: مقدار ارسال می‌شوند.

What	What?
`GET /index.html HTTP/1.1`	Get me the file at /index.html; I'm using HTTP version 1.1
`Host: example.com`	The name of the host I'm connecting to is example.com
`Connection: close`	Please close the TCP connection when you've sent me the data
`User-Agent: slidedeck/0.3`	Just FYI, my client software is slidedeck 0.3
`Accept: */*`	I'll accept any kind of data in response!

پاسخ

- هدرهای پاسخ نیز توسط توالی‌های CRLF از هم جدا می‌شوند
- بدنه پاسخ توسط دو توالی CRLF از هدرها جدا می‌شود

What	What?
`HTTP/1.1 200 OK`	I'm using HTTP version 1.1; that request is OK!
`Content-Type: text/html`	I'm going to send you some text that happens to be HTML
`Content-Length: 1337`	You'll need to read 1337 bytes to get all of the response body

`<!doctype html>`
`<html>`
`...`
	The response body

اسم‌ات چیست؟

- ما تمام این مدت در مورد آدرس‌های IP صحبت می‌کردیم
- به خاطر سپردن «example.com» به جای «93.184.216.34» آسان‌تر است.
  - و خیلی آسان‌تر از به خاطر سپردن «2606:2800:220:1:248:1893:25c8:1946» :)
- Domain Name System (DNS) نام‌ها را به آدرس‌های IP تبدیل می‌کند.
- DNS (بیشتر اوقات) از UDP استفاده می‌کند.
- معمولاً به پورت ۵۳ گوش می‌دهد.
- کلاینت‌ها از سرورهای DNS درخواست رکورد می‌کنند.

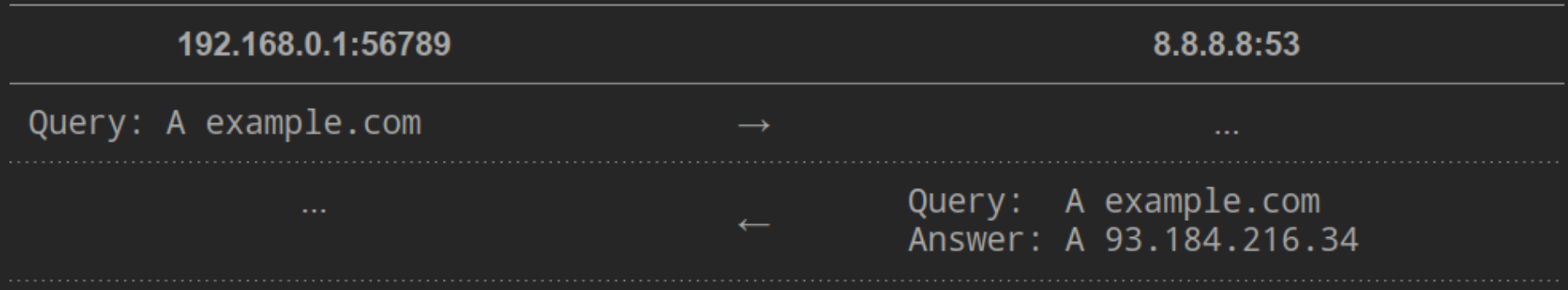
انواع رکورد (لیست ناقص است)

- انواع مختلفی از رکورد DNS وجود دارد؛ هر کدام هدف متفاوتی دارند.
- هر رکورد در مقابل نامی مانند «example.com» ذخیره می‌شود.

Type	Example	What?
A	93.184.216.34	An IPv4 Address
AAAA	2606:2800:220:1:248:1893:25c8:1946	An IPv6 Address
CNAME	origin.example.com	An alias for another name
MX	mail.example.com	A mail exchange handler
NS	ns1.webhost.com	An authoritative nameserver
TXT	Clacks-Overhead=GNU Terry Pratchett	Some human-readable text

بررسی یک مثال

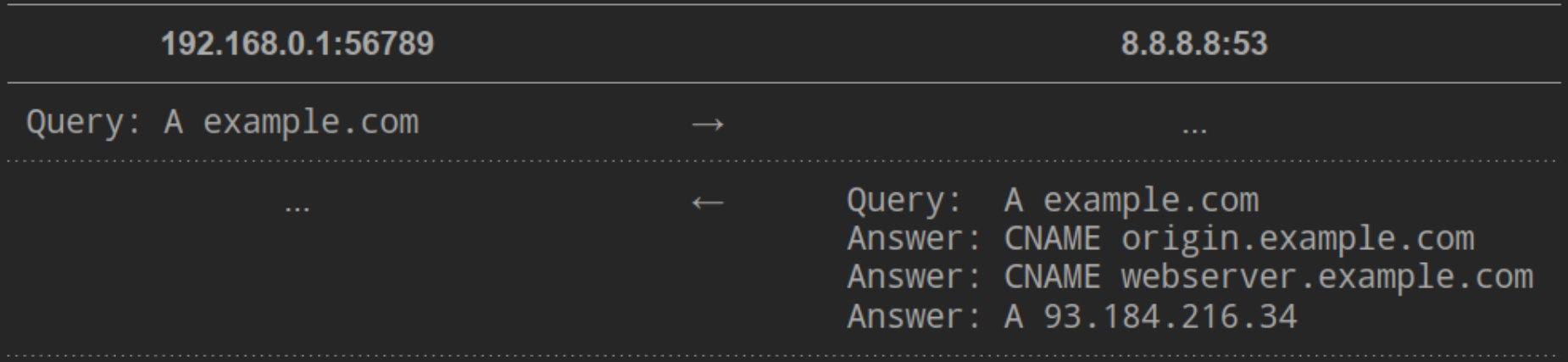
- کوئری‌های DNS[4] از UDP استفاده می‌کنند، بنابراین هیچ handshake وجود ندارد.
- این همچنین به این معنی است که مرتبط کردن درخواست‌ها و پاسخ‌ها می‌تواند دشوار باشد.
- پاسخ شامل کوئری است، بنابراین کلاینت می‌داند که به چه چیزی پاسخ می‌دهد.
- درخواست و پاسخ در واقع متن ساده نیستند، اما خواندن متن دودویی در مثال‌ها دشوار است.



[4] فرآیندی که در آن یک دستگاه (مانند رایانه یا تلفن هوشمند) از یک سرور سامانه نام دامنه (DNS) آدرس IP مرتبط با یک نام دامنه خاص را درخواست می‌کند.

CNAMEs

- برای برقراری ارتباط با یک میزبان به یک آدرس IP نیاز داریم.
- اگر هیچ رکورد A برای نام وجود نداشته باشد، اما یک رکورد CNAME وجود داشته باشد، سرور DNS با رکورد CNAME و در صورت وجود رکورد A برای آن نام، پاسخ خواهد داد.



Load Balancers

- یک سرور به ندرت برای مدیریت تمام ترافیک شما کافی است.
- Load balancer، درخواست‌های ورودی را بین چندین سرور تقسیم می‌کنند.
- برخی از Load balancer ها در لایه انتقال (TCP و غیره) کار می‌کنند.
- برخی دیگر در لایه کاربرد (HTTP و غیره) کار می‌کنند.
- لایه transport «آسان‌تر» است (یعنی به زمان CPU کمتری نیاز دارد).
- در لایه application قدرتمندتر است.
  - می‌توانید درخواست‌ها را مثلاً برای یک نقطه پایانی HTTP خاص به مجموعه‌ای متفاوت از سرورها ارسال کنید.
  - می‌توانید به برخی از درخواست‌ها بدون برخورد با سرور backend پاسخ دهید.
- هر دو نوع الگوریتم‌های Load balancer متعددی دارند.
  - Round Robin / Weighted Round Robin
  - Least Connections
  - Hashed on some property of connection (مثلاً Source IP)
  - Random

Transport Layer Load Balancers

- تمام بسته‌های مربوط به یک جلسه TCP به یک سرور backend ارسال و از آن دریافت می‌شوند.
- برای هر سرویس backend که از TCP استفاده می‌کند، کار می‌کند.
  - این شامل اکثر سرویس‌های backend می‌شود.
    - سرورهای وب
    - سرورهای پایگاه داده
    - Toasterهای فعال در اینترنت
- می‌توانند برای UDP نیز کار کنند، اما پروتکل لایه کاربرد در بالا باید بدون وضعیت باشد و/یا باید از یک الگوریتم Load balancer مبتنی بر هش استفاده کنید.
- نیازی به رمزگشایی ترافیک در حال انتقال نیست.
- درخواست‌ها را می‌توان فقط بر اساس جزئیات سطح انتقال یا اینترنت/شبکه مانند آدرس IP منبع تقسیم کرد.
- به طور کلی قابلیت‌های آنها نسبتاً محدود است.

Application Layer Load Balancers

- آنها در واقع پروتکل لایه برنامه (مثلاً HTTP) را درک می‌کنند.
- این به شما امکان می‌دهد کارهای مفیدی مانند موارد زیر انجام دهید:
  - تقسیم درخواست‌ها بر اساس جزئیات لایه برنامه (مثلاً مسیر HTTP، رشته پرس و جو، کوکی‌ها)
  - پاسخ به برخی درخواست‌ها بدون نیاز به سرور backend (مثلاً هدایت HTTP به HTTPS)
  - Edge Side Includes (یعنی فراخوانی بیش از یک سرور backend برای تشکیل یک پاسخ)
  - مسدود کردن درخواست‌هایی که مشکوک به مخرب بودن آنها هستید (مثلاً درخواست HTTP حاوی بار XSS احتمالی است)

قدرت پردازش بسیار بیشتری برای اجرا نیاز دارد

- معمولاً فقط برای یک پروتکل ساخته شده است (مثلاً یک Load Balancer HTTP واقعاً نمی‌تواند کاری با اتصالات MySQL انجام دهد)
- اگر از یک انتقال رمزگذاری شده مانند TLS استفاده می‌کنید، Load Balancer باید ترافیک ورودی را قبل از پردازش رمزگشایی کند.
  - این بدان معناست که باید کلیدهای خصوصی خود را در Load Balancers خود مستقر کنید.
  - گاهی اوقات ممکن است لازم باشد بعداً دوباره رمزگذاری کنید (مثلاً برای داده‌های دارنده کارت)

Network Address Translation (NAT)

- فضای IPv4 محدود است - آدرس‌های IPv4 اعداد صحیح بدون علامت ۳۲ بیتی هستند.
- حداکثر آدرس‌ها: ۴,۲۹۴,۹۶۷,۲۹۵
- با کم کردن محدوده‌های رزرو شده، در واقع فقط ۳,۷۰۲,۲۵۸,۴۳۰ می‌شود.
- بیش از ۷,۶۰۷,۰۰۰,۰۰۰ نفر در زمین تا مارس ۲۰۱۸
- چند دستگاه متصل به اینترنت دارید؟
    - آخرین باری که بررسی کردم، بیش از ۳۰ دستگاه در شبکه خانه ی من وجود داشت.
- NAT یک راه حل برای مشکل فضای IPv4 است، اما همچنین راهی خوب برای اطمینان از خصوصی بودن واقعی شبکه شما نیز هست.
    - به عنوان مثال، آدرس‌های خصوصی را نمی‌توان از اینترنت عمومی مسیریابی کرد، مگر اینکه صریحاً اجازه دهید.

نکته‌ی فرعی: فضای رزرو شده‌ی IPv۴ (Reserved IPvfour Space)

- محدوده‌های استفاده خصوصی (Private-use Ranges):
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
- محلی / حلقه برگشتی (Local / loopback):
  - 127.0.0.0/8
  - 0.0.0.0/8
- و حدود 10 محدوده یا بیشتر که به دلایل مختلف رزرو شده‌اند
  - به عنوان مثال مستندات، محدوده‌های پخش، «برای استفاده‌های آینده هستند»

نحوه به کار گیری NAT

- دستگاه A (192.168.0.10) در شبکه شما و پشت NAT قرار دارد.
- می‌خواهد به سرورهای DNS گوگل (پورت 53، 8.8.8.8) متصل شود.
- دستگاه A یک بسته ارسال می‌کند:
  - مک مبدا: aa:aa:aa:aa:aa:aa
  - مک مقصد: cc:cc:cc:cc:cc:cc (رابط داخلی روی دروازه پیش‌فرض)
  - آی‌پی/پورت مبدا: 192.168.0.10:34567 (آی‌پی دستگاه A)
  - آی‌پی/پورت مقصد: 8.8.8.8:53
- دروازه پیش‌فرض بسته را دریافت می‌کند و آی‌پی/پورت مبدا را قبل از ارسال به آدرس زیر بازنویسی می‌کند:
  - آی‌پی/پورت مبدا: 62.52.42.32:45678 (آی‌پی عمومی دروازه)
  - آی‌پی/پورت مقصد: 8.8.8.8:53
- ترجمه ضبط می‌شود تا ترافیک برگشتی بتواند IP مقصد و پورت خود را ترجمه کند.