Lecture 12: Open Problems in Transport Security

6.1600 – Fall 2023

Corrigan-Gibbs & Zeldovich
MIT

Plan

- What encryption leaks
  * Who's talking to whom
  * How much they're saying
- Availability → also not protected
- Why eliminating leakage isn't enough
  * Server compromise
  * Private Info Retrieval

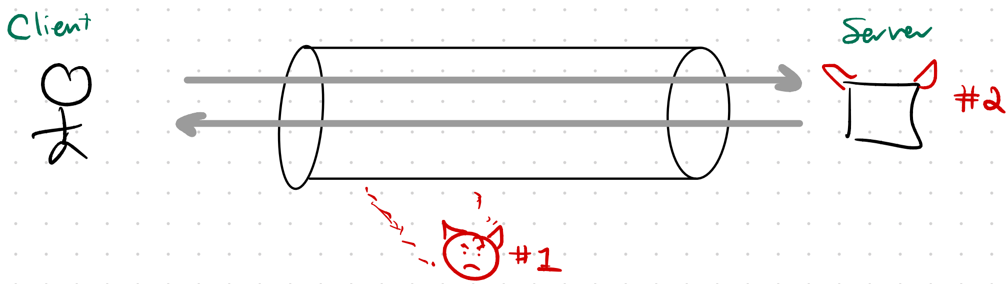
Logistics

* Midterm Wed - this room, this time
  Open laptop, no network
* Monday: Guest speaker from BU Tech Law Clinic

(Set up laptop for Mirai demo)

Recap: Encryption in practice

- We have encryption, we have authentication.
- Using TLS, we can get "encrypted & auth pipe"
  - ↳ Convenient! Can run your favorite TCP-based protocol (HTTP, POP, SMTP, FTP, ...) over TLS to get confidentiality & authenticity.



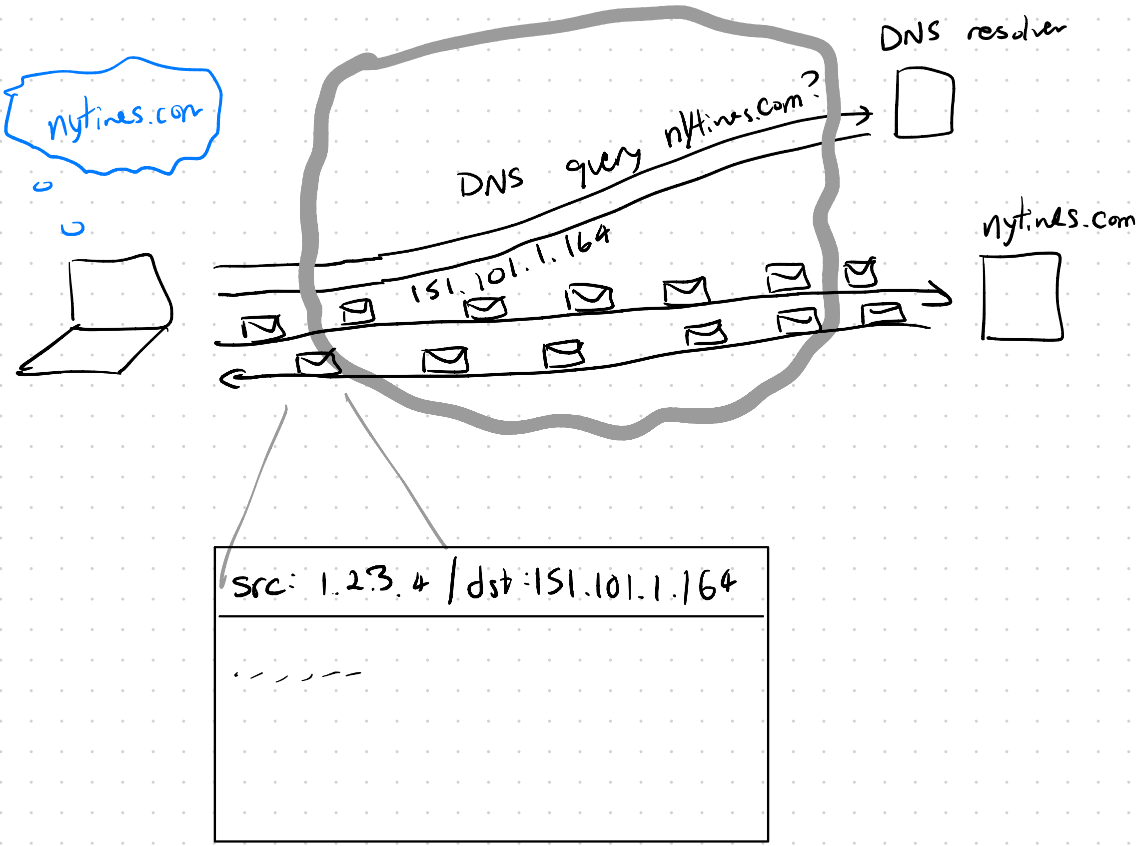
Today

1. Imperfections in the "encrypted pipe"
   ↳ What it doesn't protect
   - Metadata leakage
   - Denial of service

2. Even if we had a perfect encrypted pipe, why that's not good enough.
   - Server compromise
   ... more in next module

Problem: Net attacker sees who is talking to whom

- From this "metadata" can infer your political beliefs, religion, medical condition, travel plans, kids' school, ...
- Also allows attacker to selectively censor your traffic



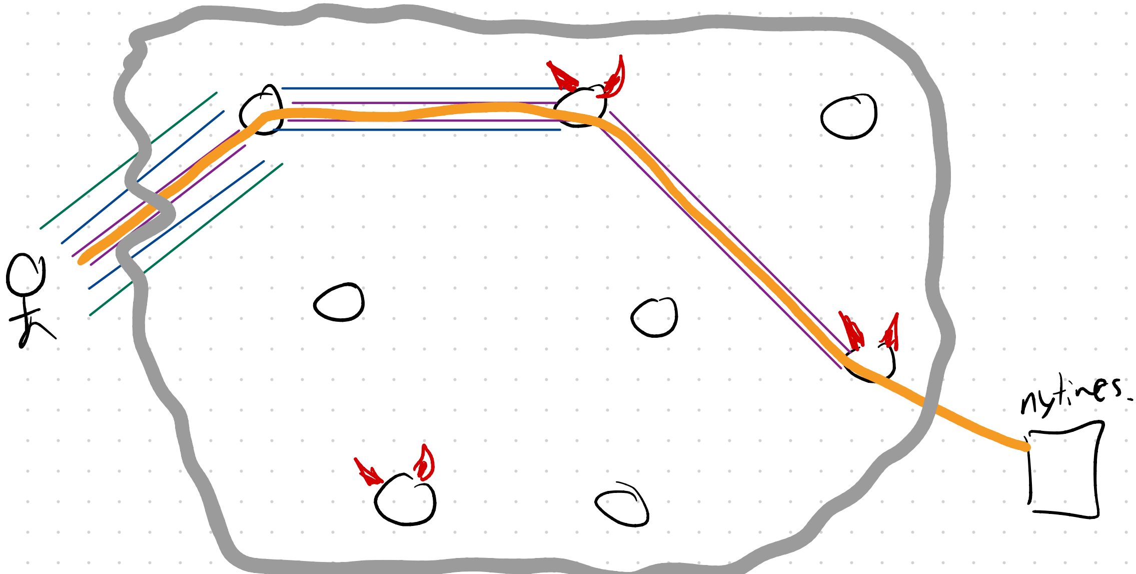
⇒ To route IP packets, routers need to know src/dst IP addresses (+ DNS + TLS leakage)

⇒ In-network attacker learns exactly who you're talking to (could talk to everyone all the time but...)

Attempt at a solution: Tor

Idea: * Bounce traffic arount internet
* Hope attacker can't see too much of it
* Give up on precise sec defn's

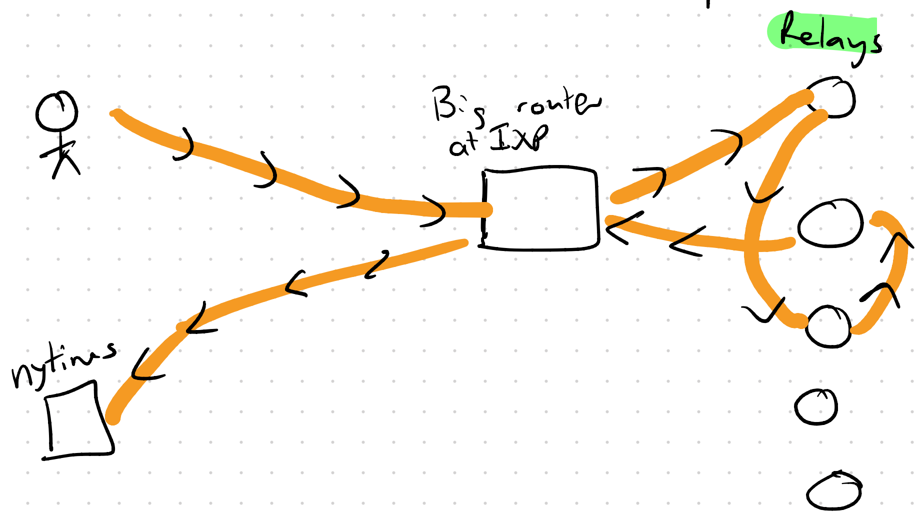
- Thousands of volunteer "relay" servers
- Build nested encrypted pipes (like TLS) through network. - 3 relays on path



Hope: If attacker is not too powerful, it will not be able to correlate input & output

* You can download & run Tor
* Millions of people use Tor daily (250 Gbps total)
* Even if security is imperfect, functionality is surprisingly good.

Unclear how much Tor helps...



Also, might worry about sending traffic through computers run by randoms on Internet.

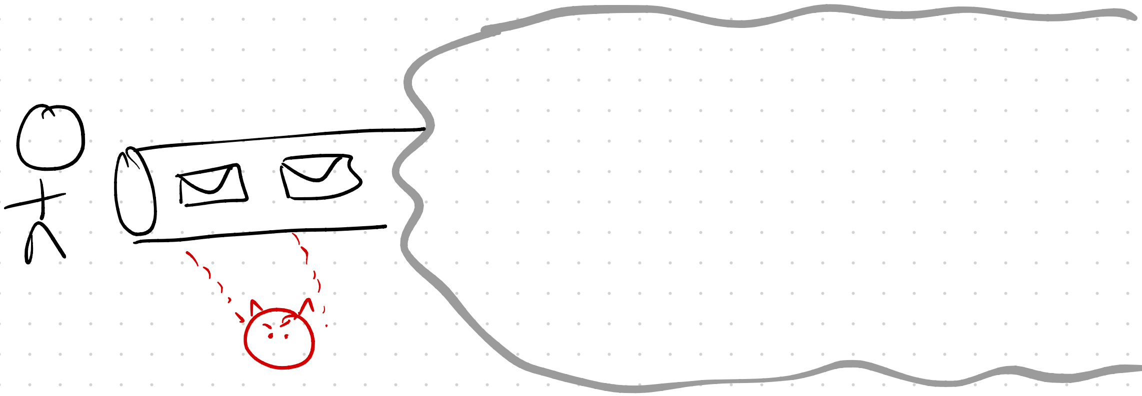
↳ Maybe no worse off?

But, it's also plausible that Tor gives you much better privacy against net attackers than anything else does... just hard to know.

↳ Frustrating state of affairs...

Maybe YOU will come up with a better solution???

Problem: Attacker sees packet sizes & timing



Can learn whether you are:

- streaming a movie (which movie)
- using SSH (which commands)
- downloading a file (which file)
- browsing the web (which page)

nytimes.com/index.html	nytimes.com/tips
1.56 MB	41.92 KB
76 asset reqs	15 asset reqs

↑ css, img, JS, fonts, ...

Attempts at a solution...

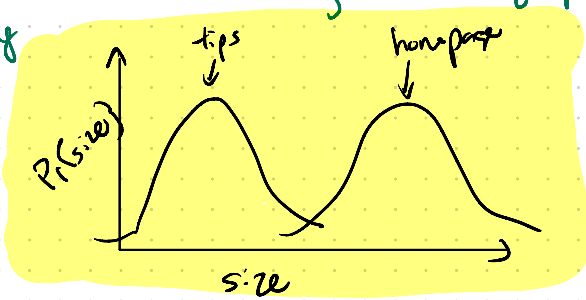
Padding: Make all nyt pages exactly 50 MB long and make 100 reqs for assets?

→ What about a page that needs 101 assets?
→ Overhead is obscene!

Sort-of secure, but too $$$

Random noise: Perturb length of each page & # of asset reqs
e.g. by adding random # of bytes.

→ Still can leak the length since long pages remain long



→ Given a handful of observations, can average out the noise.

Doesn't work at all... also $

We are a long way from understanding how to protect communications metadata

↳ In contrast enc & sigs do an excellent job at protecting comm data.

One promising (?) direction:

- Try to solve the easier problem of metadata-hiding messaging (Think: WhatsApp, iMessage, Signal)

+ Msgs are ≈ fixed length
+ Some latency OK
+ Total bytes sent per user is small
+ Few comm partners per user.

Potential for system that protects metadata w/ strong formal sec guarantee.

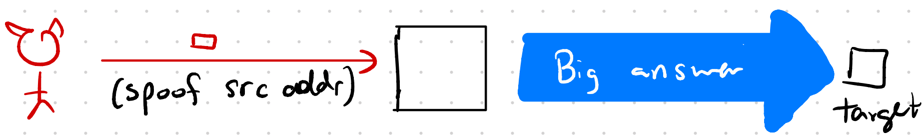
Many research papers, but still no deployed system...

Maybe you will figure it out!

ALSO

Encryption does not protect availability

- Denial of service
  - ↳ We already talked about censorship but there are other reasons: often $$$
- Basic idea: Swamp service w/ fake traffic
  - ↳ Real users can't get through
- Don't have to break enc/auth to cause damage!
- Old school: Traffic amplification

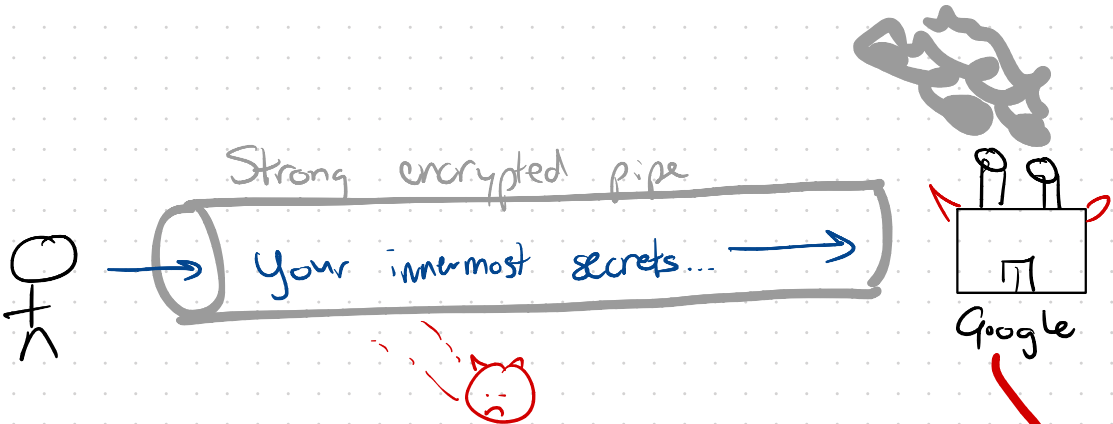


Mitigation: Detect spoofed src at network egress

- DDoS = distributed attack, often w/ hijacked machines

  e.g. Mirai botnet – IoT devices
  - ↳ Reach TB/s throughput (~400 M reqs/sec)
    - ↳ If can process 10k req/s on a machine ⇒ 40k machines
- No great plan: * Use CDNs to absorb traffic
  - ↳ economy of scale.
  - * Try to make devices harder to compromise

Encryption doesn't protect against server compromise



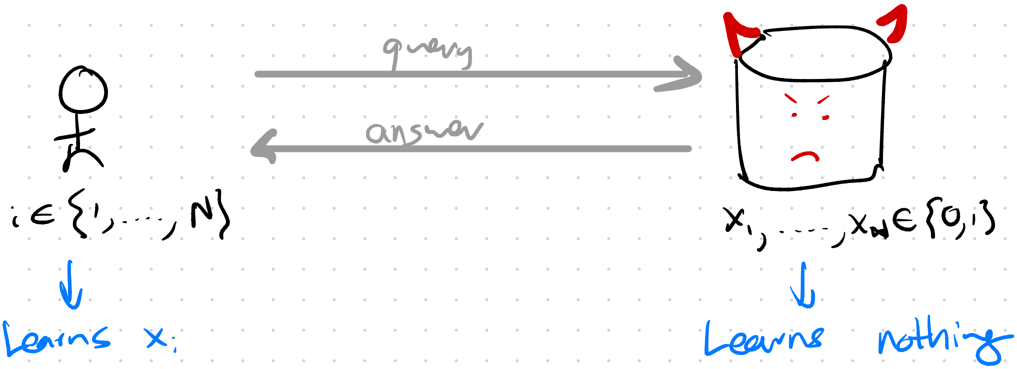
* Your search queries leak your med conditions, religion, interests, beliefs, ...
* Not to mention Gmail, files, etc.

- Lose it in breach/compromise
- Sell your data
- Be compelled to turn it over to LEA, etc.

Next module focuses on platform security...
↳ how to protect against server compromise by careful system design
↳ I'll give one example of how fancy crypto can help as well.

Private Info Retrieval

Goal: Read record from DB while hiding which record you read.
(Abstraction of private Google-search problem)



Correctness: Honest client interacting w/ honest server always outputs correct DB bit

Security: Client's query is a CPA-secure encryption of its index i.
(Why is CPA security good enough?)
Query "leaks nothing" about i.

↳ Client can fetch data from server w/o server learning what it fetched!

Surprise is that PIR is possible with $\ll N$ bits of communication.

We need one more tool:

Additively homomorphic encryption

- CPA-secure secret key enc scheme (Enc, Dec)
  - ↳ Cannot be CCA secure... why?
- Msg space is $\mathcal{M} = \mathbb{Z}_p$ ← ints mod p
- Extra property that for all $k \in \mathcal{K}$ ← keyspace

for all $m, \hat{m} \in \mathcal{M}$ (addition mod p)

$$\text{Enc}(k, m) * \text{Enc}(k, \hat{m}) = \text{Enc}(k, m + \hat{m})$$

↖ some op on ctexts

⇒ Can add msgs under encryption

⇒ Can multiply by constants: $\text{Enc}(k,m) * \text{Enc}(k,m) = \text{Enc}(k, 2m)$

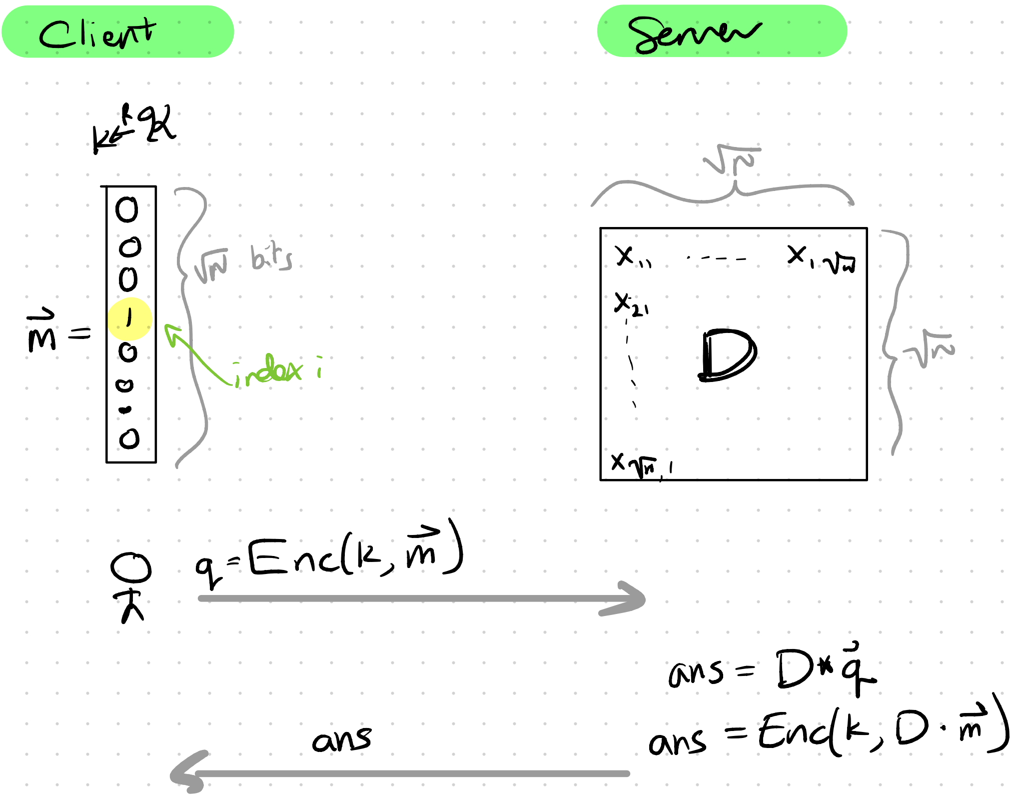
⇒ Can compute matrix-vector product of enc. vector & public matrix

$$D * \text{Enc}(\vec{m}) = \text{Enc}(D \cdot \vec{m})$$

(Can build from DDH assumption... essentially ElGamal)

Construction of PIR

- View N-bit DB as $\sqrt{N}$-by-$\sqrt{N}$ matrix
- Client wants a bit in column $i \in \{1, \ldots, \sqrt{N}\}$



$Dec(k, ans)$

$= D \cdot \vec{m}$

$= D \cdot [0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^T$

$=$ ith col of DB.

Correctness: By construction

Security: By sec of enc scheme

Comm cost: $\sqrt{N}$ cts $\ll N$

Server comp cost is high...
BUT promising!

Additively Homomorphic Encryption

Group $\mathbb{G}$, generator g { As used in ElGamal encryption
order q

Msg space $\{0, \ldots, M-1\}$ ← $M = poly(\lambda)$. "msgs are not too big"

$Gen() \to (sk, pk)$

- $x \xleftarrow{R} \{1, \ldots, q\}$
- $(sk, pk) \leftarrow (x, g^x)$

$Enc(pk, m) \to ct$

- $r \xleftarrow{R} \{1, \ldots, q\}$
- $ct \leftarrow (g^r, g^m \cdot (pk)^r)$

$Dec(sk, ct) \to m$

- $(R, T) \leftarrow ct$
- $V \leftarrow T^{sk} \cdot R^{-1} \in \mathbb{G}$
- find m s.t. $g^m = V$ by brute force
- output m

Same as CPA-secure ElGamal, with msg "in the exponent"

⇐ Msg cannot be too large...

Why this scheme is additively homomorphic:

$$\left(g^{r},\ g^{m}\cdot(pk)^{r}\right) \leftarrow Enc(m; r)$$

$$\left(g^{\hat{r}},\ g^{\hat{m}}(pk)^{\hat{r}}\right) \leftarrow Enc(\hat{m}; \hat{r})$$

$$\left(g^{r+\hat{r}},\ g^{m+\hat{m}}(pk)^{r+\hat{r}}\right) \leftarrow Enc(m+\hat{m}, r+\hat{r})$$


The
End