Lecture 8 - Transport Security

MIT - 6.5060

Fall 2021

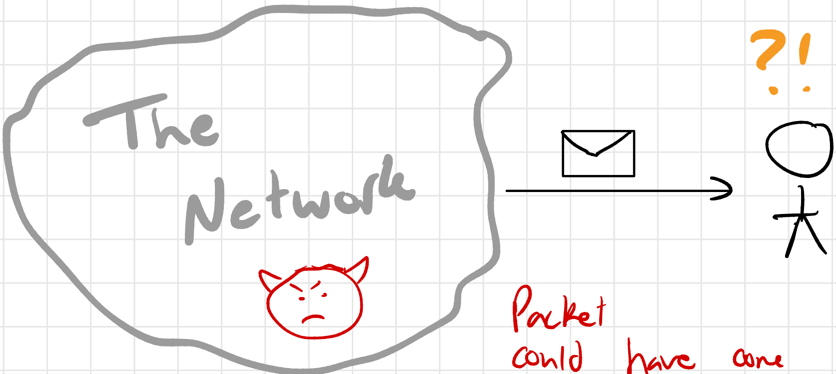
C.G, Devadas, Kalai, Zeldovich

Plan

- Network (in)security
- Encryption
  * Weak defn (CPA)
  * One-time pad
  * Encryption from PRF
- What's missing

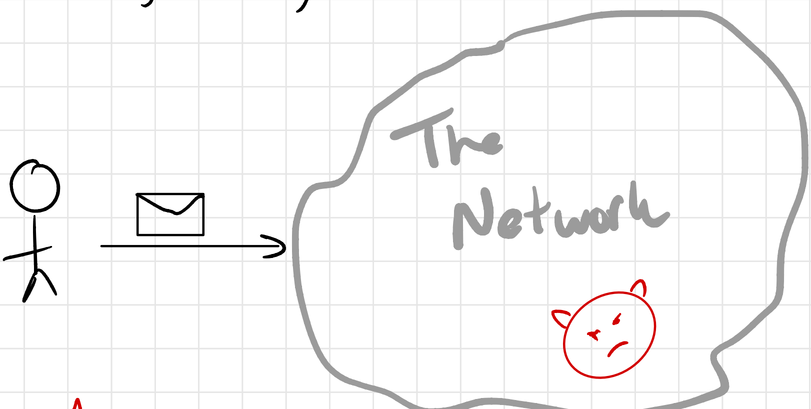
Background

Mental model for integrity...

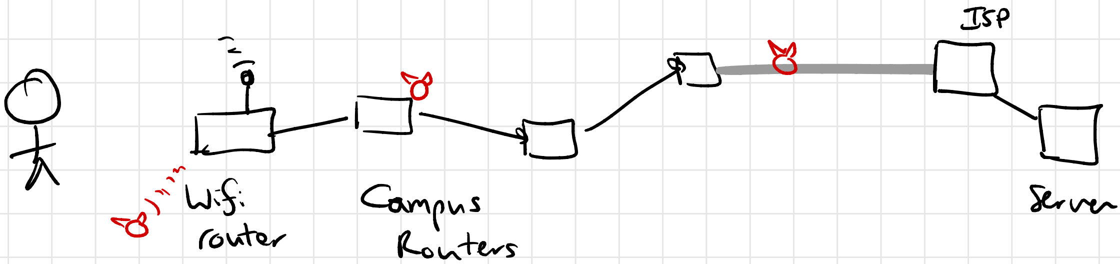


Packet could have come from anyone

For confidentiality...



Anyone can read the packets you send across a network.



Many places for an adversary to see your network traffic — every hop!

↳ Attacker doesn't need privilege — see tcpdump on LAN

Standard network protocols provide NO AUTH/ENC!

Ethernet — LAN
IP
DNS
email (SMTP, POP, IMAP)
HTTP — web content

⟹ When you query a DNS server.

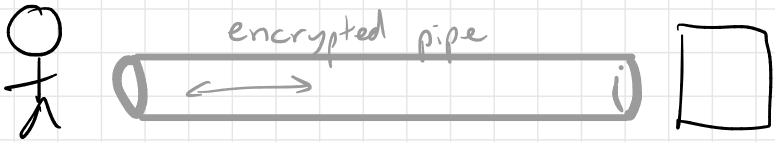
(a) Think of your query as being public
(b) Think of the answer as coming from an adversary.

Really?! Yes.

How can we get any integrity/privacy?
↳ Crypto: encryption & authentication.

Systems in which encryption appears...

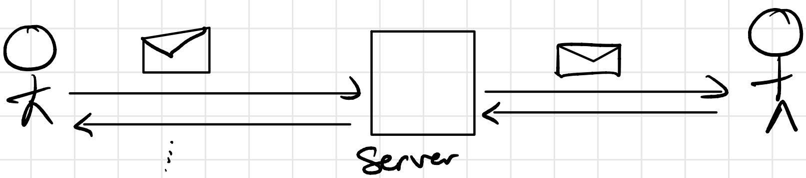
Encrypted interactive streams (web, SSH, email, ...)



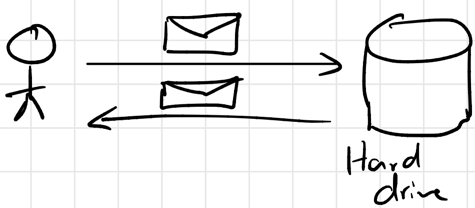
Like TCP (bidirectional stream)
+ encryption & auth... more later

google.com

High-latency encrypted (WhatsApp, Signal, iMsg, ...)



File encryption (PGP, pass mgr, ...)



Plan

* Begin with simplest form of encryption
* Build up to fancier / more powerful ones
* End module by seeing encryption in situ

Roadmap

Today: Weak encryption for fixed-len msgs with shared key

Next time: Strong (authenticated encryption) encryption for var-len msgs "

Next week: " " without shared key

In Two weeks: → " for streams "

↳ Encryption in applications (protocol-level attacks, extra properties)

Finally: Problems that encryption <u>doesn't</u> solve.

↳ e.g. hiding length of msg, recipient, ...

Note: You should almost never implement these things yourself! Better to use solid library when you can!

Encryption syntax

security parameter

key space $\mathcal{K}$ today: $\{0,1\}^n$ $(n = 128, 256)$

msg space $\mathcal{M}$ $\{0,1\}^n$

ciphertext space $\mathcal{C}$ $\{0,1\}^{2n}$

$$\mathrm{Enc}: \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{C}$$

$$\{0,1\}^n \times \{0,1\}^n \longrightarrow \{0,1\}^{2n}$$

$$\mathrm{Dec}: \mathcal{K} \times \mathcal{C} \longrightarrow \mathcal{M}$$

(We will see some schemes in which decrypt can also output "fail.")

What does it mean for an encryption scheme to be secure?

"Eavesdropper can't recover msg"

↳ Admits schemes that leak $1/2$ of msg bits.

"Eavesdropper can't recover any bit of msg"

↳ Admits schemes that leak whether two ctext bits encrypt same plaintext bits

"Eavesdropper can't distinguish ctext from random string"

↳ Maybe too strong? Seems ok to have first bits of ctext always be 000...

$\Rightarrow$ Not so easy to cook the right defn!

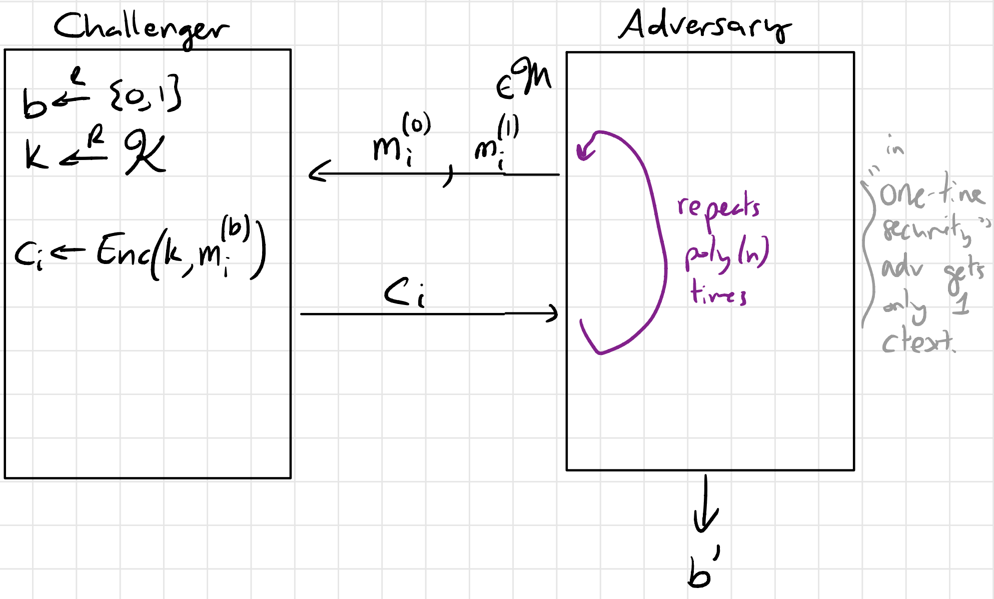
Weak security...

Indistinguishability under chosen plaintext attack (CPA)

also IND-CPA

Intuition: Scheme is CPA secure if attacker can't tell which of two chosen msgs are encrypted

Enc scheme (Enc, Dec) is CPA-secure iff $\forall$ eff advs A, A wins game w prob $\leq \frac{1}{2}$ + negl.



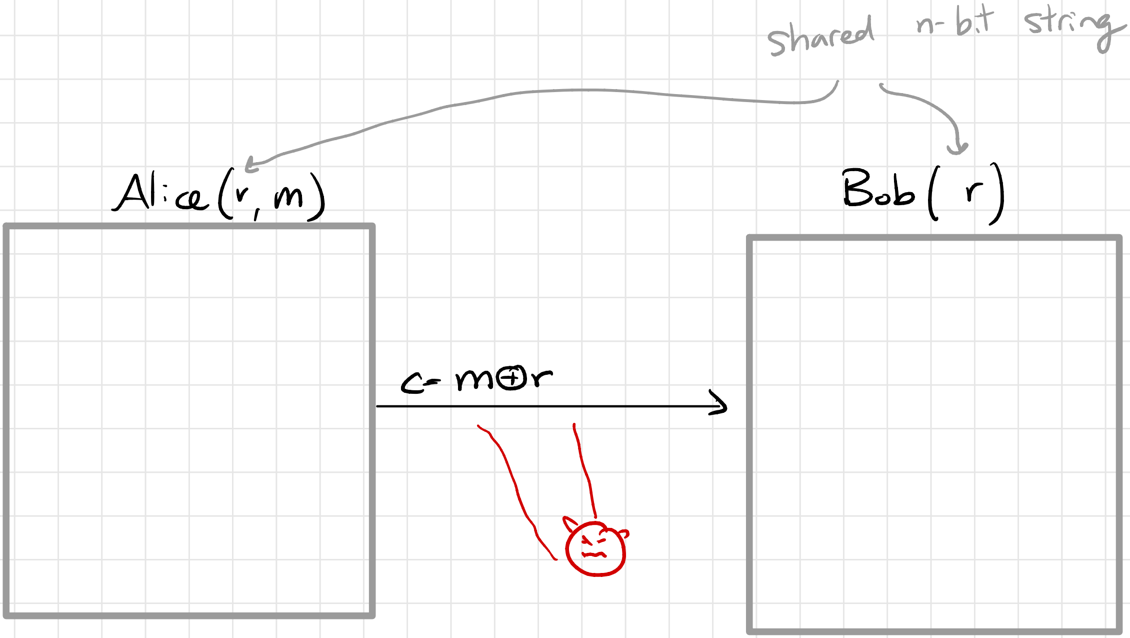
Attacker wins if $b = b'$

Even if attacker gets to choose msgs being encrypted, still can't learn distinguish one from another.

One-time pad

(Miller 1882, Vernam 1991, Mauborgne ....)

- The first encryption scheme with a strong theoretical foundation
- Widely used in practice through 1970s.



Benefit: * "Perfect" security — for any c, all m's are equally likely.
* Secure against comp. unbounded attacker

One-Time Pad

Problem: Need new r value for each msg.
↪ inherent for perfect info-theoretic security.
It's called the one-time pad for a reason.

TWO-TIME PAD ATTACK

$$c_1 = m_1 \oplus r$$
$$c_2 = m_2 \oplus r$$
$$c_1 \oplus c_2 = m_1 \oplus m_2$$

From: henrycg@mit.edu ...

Subject: ___ ...

If attacker knows bits of m_1, gets plaintext of m_2!

⇒ OTP is maybe ok for embassys, not for high-b/w computer systems

Historical aside: Venona (1943, ...)
- USSR used OTP for mil & diplomatic comms
- Duplicated pads shipped to a number of embassies
⇒ Two-time pad attack!
- US got copies of all telegrams (network is insecure!)
- Decryption continued through 1980. (!)

Idea: Use pseudorandomness (PRF) to generate many pads from short key.

Weak (CPA-secure) encryption for fixed-length msgs.

Uses PRF $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$

$\mathrm{Enc}(k, m)$:

$r \xleftarrow{\$} \{0,1\}^n$ ("nonce")

output $(r, F(k,r) \oplus m)$

$\mathrm{Dec}(k, (r,c))$:

output $c \oplus F(k,r)$

Alice $(k, m_1, \dots, m_T)$

$c_1 = \mathrm{Enc}(k, m_1)$

$c_2 = \mathrm{Enc}(k, m_2)$

$\vdots$

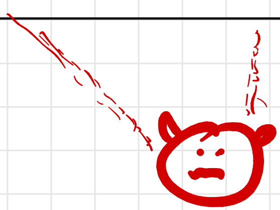
$r_1, c_1 = m_1 \oplus F(k, r_1)$

$r_2, c_2 = m_2 \oplus F(k, r_2)$

$\vdots$

$r_T, c_T = m_T \oplus F(k, r_T)$

Bob (k)



Notice: the block size n needs to be big enough to avoid repetitions of r values.

$\{r_1, \ldots, r_T\}$ should be distinct

What happens if not? Attacker sees:

$$(r, \ c_1 = m_1 \oplus F(k, r))$$
$$(r, \ c_2 = m_2 \oplus F(k, r))$$

$$\Rightarrow c_1 \oplus c_2 = m_1 \oplus m_2 \quad \Big\{ \text{"Two-time pad attack"}$$

By Birthday Paradox...

$$\boxed{\text{Need: } \quad \frac{T^2}{2^n} \ll 1}$$

AES has $n = 128 \Rightarrow$ After 2^{30} msgs or so, need to change keys. ("rekey")

Security intuition

Attacker sees pairs

where $k \xleftarrow{R} \mathcal{K}$ is a random secret key.

$$(r_1, \; m_1 \oplus F(k, r_1))$$
$$\vdots$$
$$(r_T, \; m_T \oplus F(k, r_T))$$
$$\Downarrow$$

By PRF security property (& provided that all r's distinct)

$$(r_1, \; m_1 \oplus \text{random}_1)$$
$$\vdots$$
$$(r_T, \; m_T \oplus \text{random}_T)$$
$$\Downarrow$$

One-time pad security. ✓

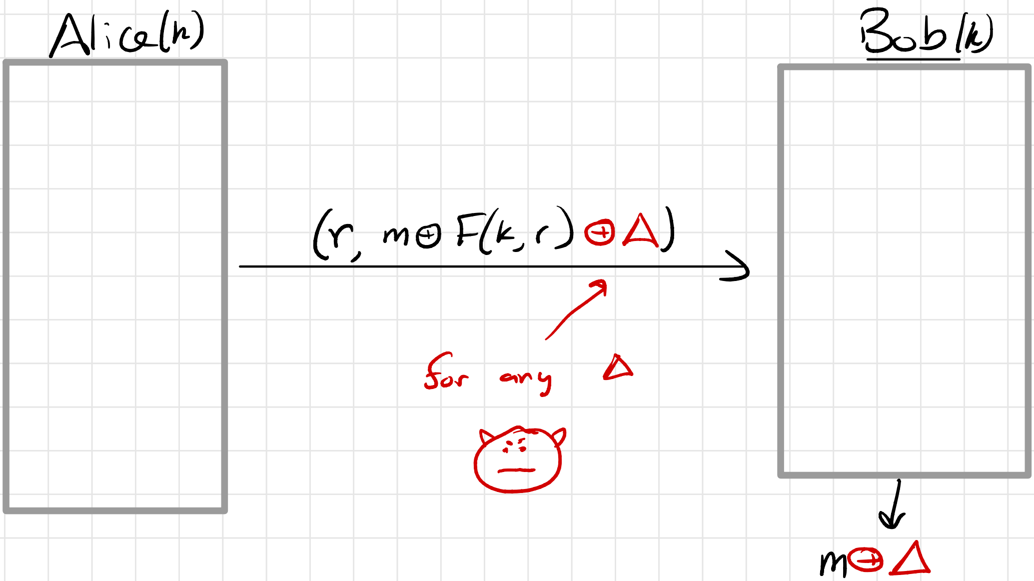
Note: * Security argument here only uses the fact that $(r_1, \dots, r_T)$ are distinct whp.

* If sender and receiver can have state, can set $r_1 = 1, r_2 = 2, r_3 = 3, \dots$

↳ Then, no need to send r values.

Why do we call CPA-secure encryption "weak"?

PROBLEM 1: CPA security definition guarantees <u>nothing</u> about integrity/authentication.



m = "Send \$100 to Srini"

$\Delta =$ [ ] ← "Srini" ⊕ "Yael!"

$m \oplus \Delta$ "Send \$100 to Yael!"

Why do we call CPA-secure encryption "weak"?

PROBLEM 2: When used in the context of a larger system, can create all sorts of security problems.

(More generally, security def'n says nothing about what happens if Bob decrypts an adv chosen ct.)

Might have an example on the next theory lab!