Lecture 8 - Transport Security

MIT - 6.1600

Fall 2023

Corrigan-Gibbs & Zeldovich

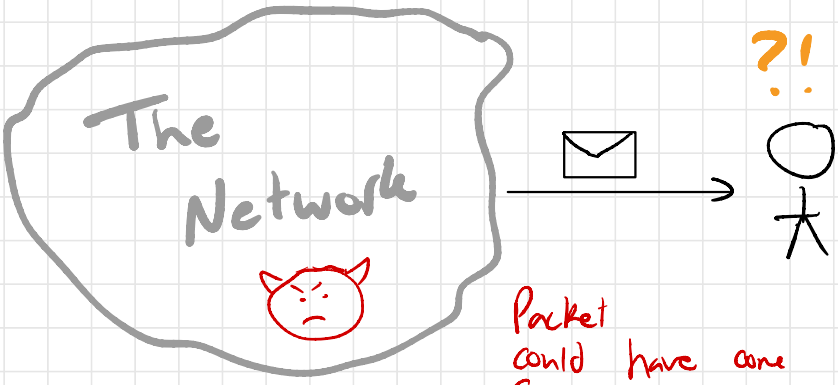
Plan

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

[Set up laptop]

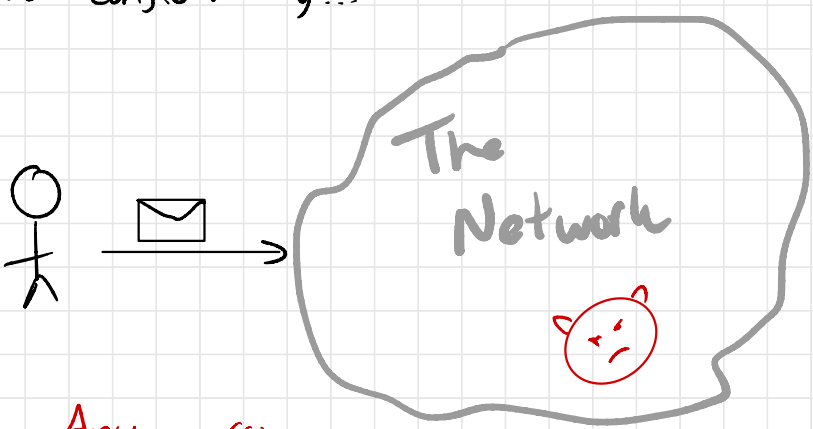
Background

Mental model for integrity...

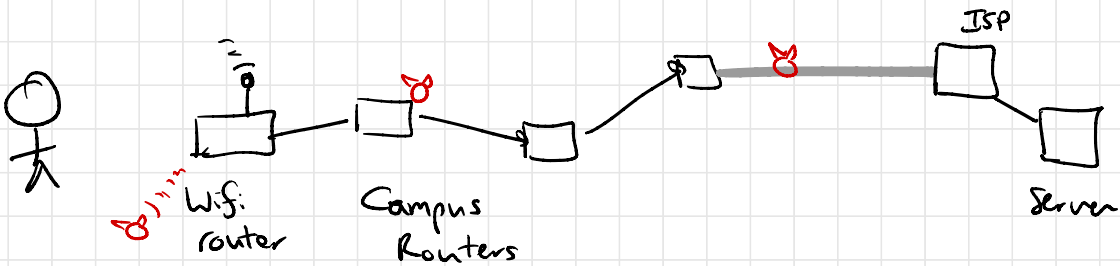


Packet could have come from anyone

For confidentiality...



Anyone can read the packets you send across a network.



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

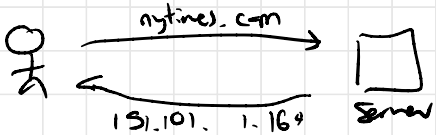
↳ Attacker doesn't need privilege — see tcpdump on LAN

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.

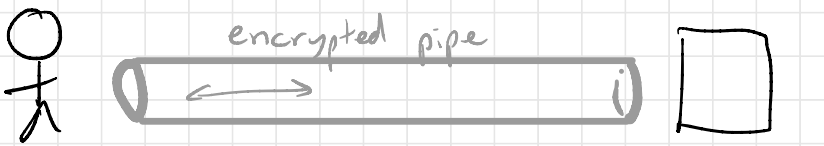


Really?! Yes.

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

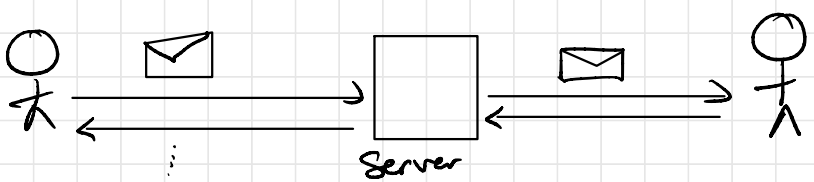
Systems in which encryption appears...

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

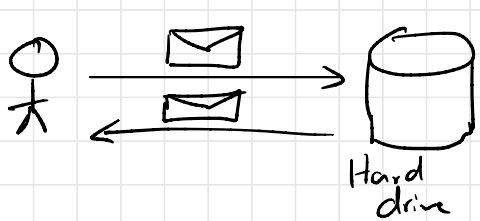


Like TCP (bidirectional stream)
+ encryption & auth... more later

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 s:th

Roadmap

Today: Weak (<PA-secure) encryption w/ shared key

Next time: **Strong** (CCA-secure) encryption w/ shared key

Next week: " " without shared key

In two weeks: Encryption in applications (protocol-level attacks, extra properties)

Finally: Problems that encryption doesn't 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

Key space $\mathcal{K}$ — today: $\{0,1\}^n$ (security parameter) ($n = 128, 256$)

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

ciphertext space $\mathcal{C}$ — $\{0,1\}^{n+\ell}$

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

$$\{0,1\}^n \times \{0,1\}^\ell \longrightarrow \{0,1\}^{n+\ell}$$

$$\text{Dec}: \mathcal{K} \times \mathcal{C} \rightarrow \mathcal{M}$$

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

Correctness: $\forall\ k \in \mathcal{K}\ \ \forall\ m \in \mathcal{M}$

$$\text{Dec}(k, \text{Enc}(k, m)) = m$$

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

"Eavesdropper can't recover msg"
↳ Admits schemes that leak ½ 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 0000...

⇒ Not so easy to cook the right defn!

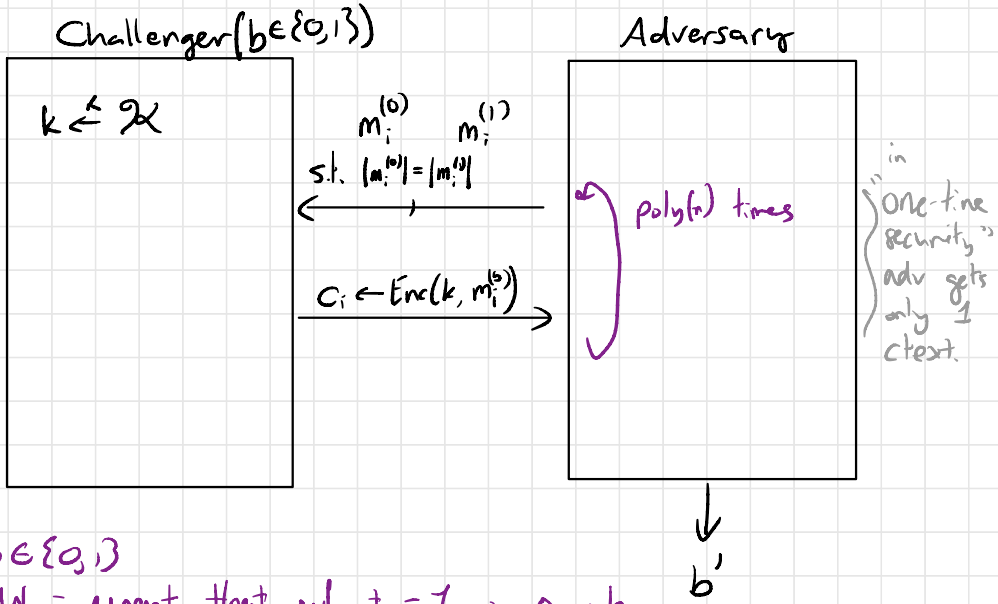
Weak security...

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

CPA-Security game.



For $b \in \{0,1\}$

Let W_b = event that output = 1 in Game b

CPA Security

WEAK!

Enc scheme $\mathcal{E} = (Enc, Dec)$ is CPA secure if $\forall$ eff advs A

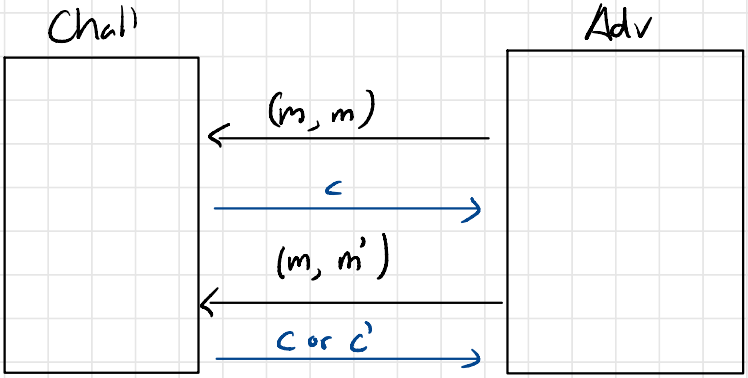
$$CPAAdv[A, \mathcal{E}] := |Pr[W_0] - Pr[W_1]| < negl.$$

Important: For an enc scheme to be even CPA (weakly) secure, must enc same msg → many ctexts
randomized, stateful, ...

If not, same msg ⟶ same ctext

↳ Attacker can detect when same msg was sent twice

↳ Enough to win in CPA game.



In practice, leaking duplicate msgs is often very problematic.

↳ Encrypt bytes of an image one by one...

[Show the ECB penguin]

↳ Attacker can inject traffic into stream

Secure encryption schemes must always use randomness...

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

One-time pad

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

$$\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$$

(key is as long as message)

$$Enc(k, m) := k \oplus m$$

$$Dec(k, c) := k \oplus c$$

Correctness $\forall\, k \in \mathcal{K},\ \forall\, m \in \mathcal{M}$

$$k \oplus (k \oplus m) = m \quad ✓$$

One-time security (sketch)

- Attacker chooses $m_0, m_1 \in \mathcal{M}$ s.t. $|m_0| = |m_1|$
- Attacker sees $Enc(k, m_0)$ or $Enc(k, m_1)$

$$\{Enc(k, m_0) : k \xleftarrow{R} \mathcal{K}\} \equiv \{Enc(k, m_1) : k \xleftarrow{R} \mathcal{K}\}$$

- Ctext leaks no info about msg

→ "Perfect" one-time security

One-Time Pad

Problem: Need new k 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: henryc@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 keys.

Randomized Counter Mode

CPA-secure encryption from PRF

Uses PRF $F: \mathcal{K} \times \{1, \ldots, N\} \to \{0,1\}^n$

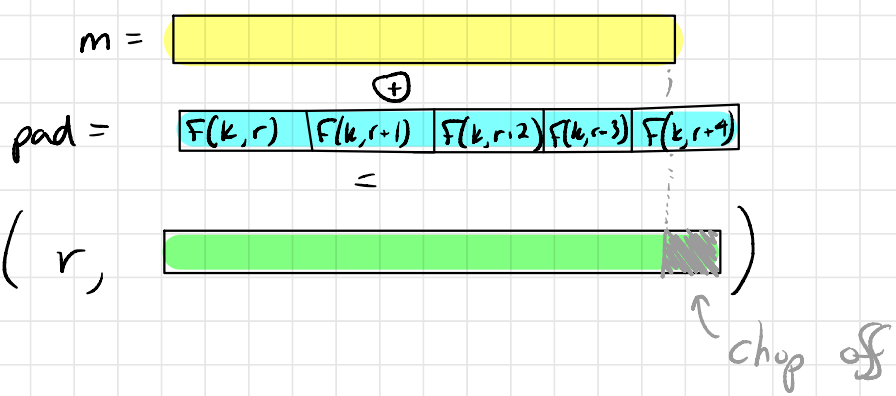
Keyspace: $\mathcal{K}$

Msg space: $\mathcal{M} = \{0,1\}^{\leq \ell}$ for $\ell = poly(n)$

Ct space: $\mathcal{C} = \{1, \ldots, N\} \times \{0,1\}^{\leq \ell}$

$Enc(k, m) :=$

$r \xleftarrow{R} \{1, \ldots, N\}$



$Dec(k, (r, c)) :=$

return msg

Randomized Counter Mode

Correctness: By construction

Security:

- If counter space size N is large enough, will never use $F(k, i)$ twice
- Appeal to PRF security
  - ↳ replace pseudo-random pad w/ true random string
- One-time pad security

If N is small ⇒ collisions ⇒ 2-time-pad attack

By Birthday Paradox... for T blocks

$$\text{Need: } \frac{T^2}{N} \ll 1$$

AES has $N = 2^{96}$ ⇒ Spec allows 2^{32} Enc calls per key

* Total data blocks 2^{64}

Note:

* If sender and receiver can have state, can set $r_1 = 1, r_2 = 2, r_3 = 3, \ldots$
  ↳ Then, no need to send r values.

* Maybe trickier to get right (power loss, VM reset, etc)

* Bad randomness?

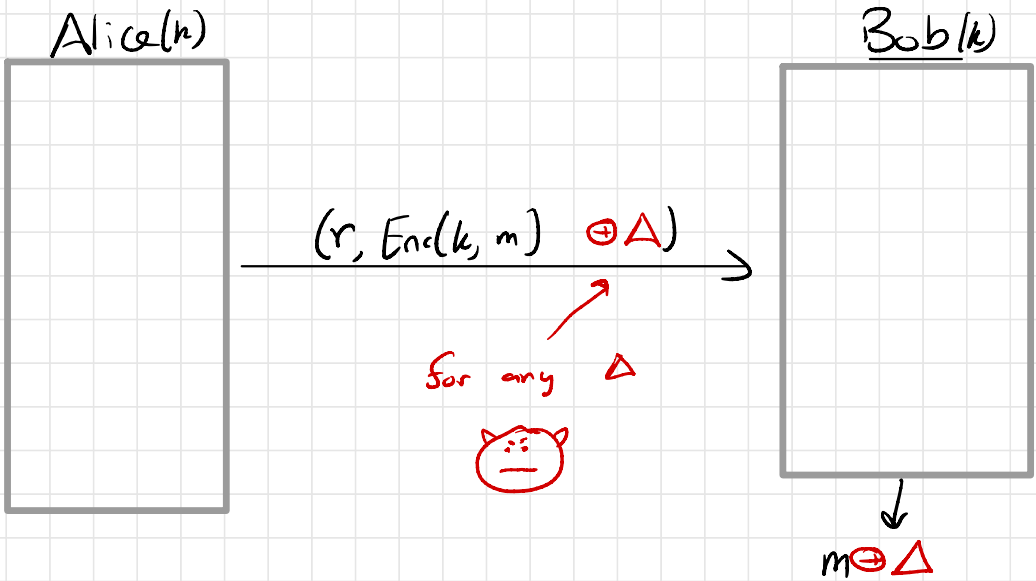
In practice, we use AES-128 as PRF... but it's a PRP.
If attacker sees encs of Q msgs of length ℓ blocks

$$\text{CPA Adv}[A, \mathcal{E}] \leq \frac{2Q^2\ell^2}{N} + (\text{advantage at breaking AES as PRP})$$

As $Q, \ell \to N^{1/4}$, security degrades
↑ See book Thm S.1.

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

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



m = "INSERT INTO USERS..."

Δ = carefully chosen string

m ⊕ Δ "DROP TABLES"

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 defn says nothing about what happens if Bob decrypts an adv chosen ct.)