Lecture 5: Back to Encryption

MIT - 6.5610
Spring 2023
Henry Corrigan-Gibbs

Plan

→ Review
  * Building blocks
  * CPA secure enc
  * MAC

→ GMAC analysis

→ Stretch break

→ CCA security & AE
  * Encrypt then MAC.

* Pset 1 due Friday 5pm!
* Reminder: Extensions

Primitives so far

One-way fn (OWF)

– "hard to invert"

$$f: \{0,1\}^n \to \{0,1\}^n$$

Pseudorand generator (PRG)

"small random str ⇒ Big PR string"

$$G: \{0,1\}^n \to \{0,1\}^{2n}$$

← any poly(n)

Pseudorand fn (PRF)

$$F: \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$$

← any poly(n)

"$F(k,\cdot)$ looks like a random fn" when key is random & hidden!

$$A^{F(k,\cdot)} \approx A^{R(\cdot)}$$

Pseudorand perm (PRP)

"block cipher"

$$F: \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$$

← must be n

$$F^{-1}: \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$$

"$F(k,\cdot)$ looks like a random perm"

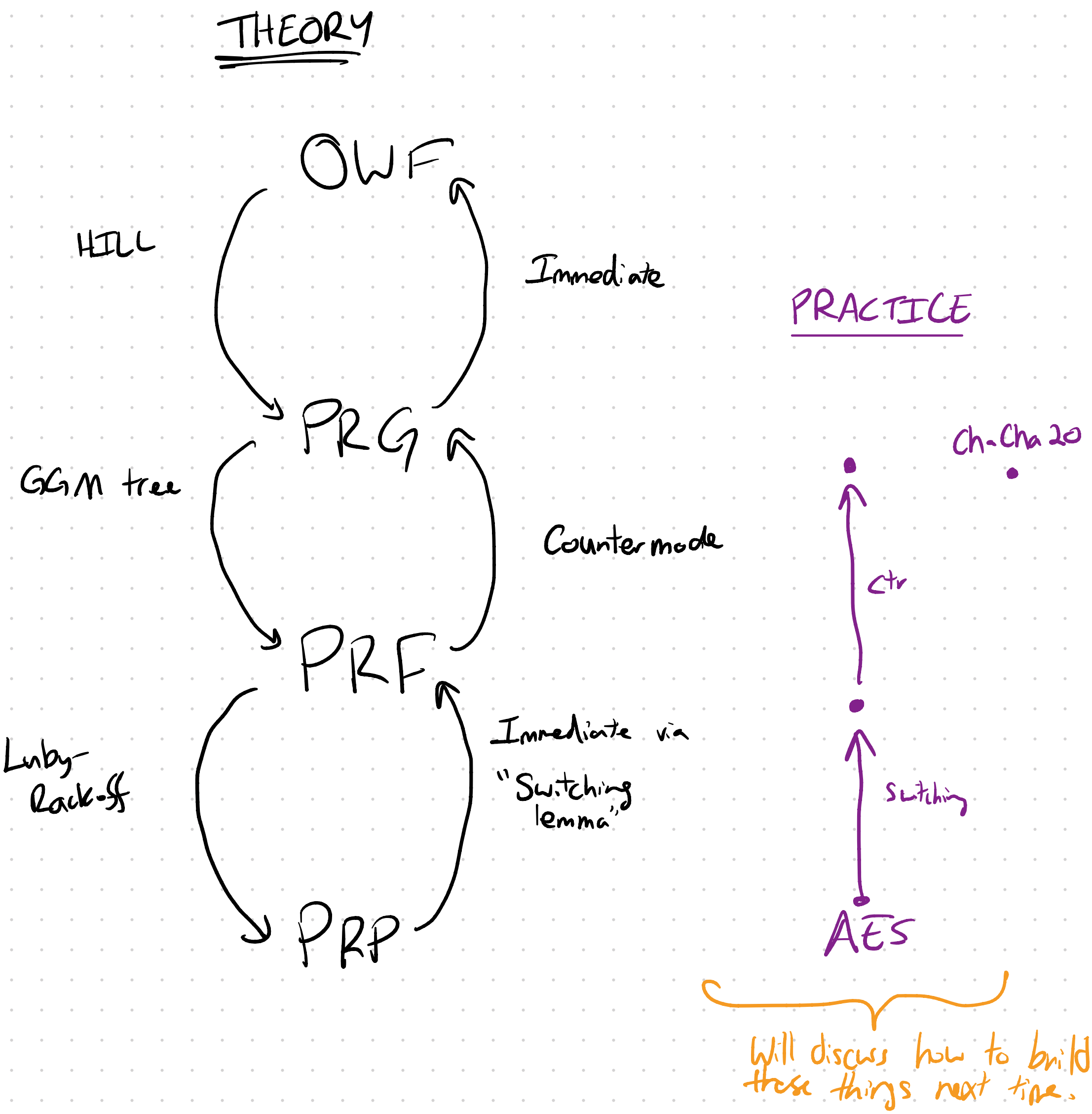
$$A^{F(k,\cdot),F^{-1}(k,\cdot)} \approx A^{R(\cdot),R^{-1}(\cdot)}$$

$\forall k \in \mathcal{K}\ \forall x \in \{0,1\}^n$:

$$F^{-1}(k, F(k,x)) = x$$

Make sure you know & understand the formal defn's (see past notes)

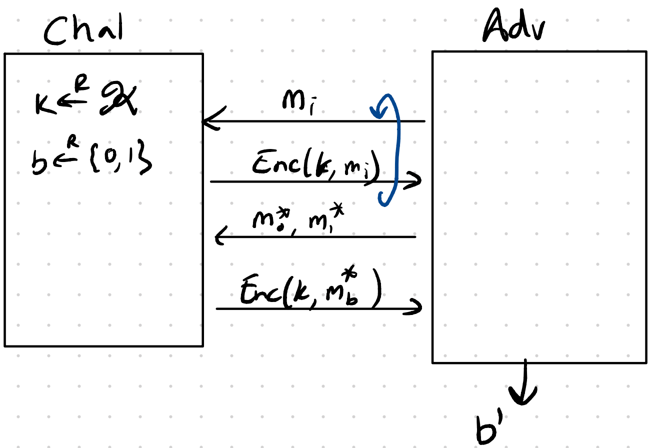
All equally powerful in theory terms...



Bigger tools

CPA-Secure encryption (WEAK / passive sec)

(Enc, Dec) over $\mathcal{K}$ is CPA secure if $\forall$ eff adv $\exists$ negl fn s.t. $|\Pr[A \text{ outputs } 1 \text{ when } b=0] - \Pr[A \text{ outputs } 1 \text{ when } b=1]| \leq \text{negl}$



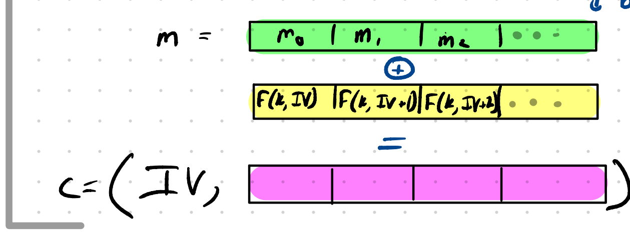
Even getting WEAK CPA security requires randomness!
↳ Eg: encrypting SSH comm P A S S W O R D
↳ obvious???

Counter mode using PRF $F: \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$

$\text{Enc}(k, m_0 \| m_1 \| \cdots \| m_{\ell-1}) :=$

$IV \xleftarrow{R} \{0,1\}^n$

Doesn't need to be multiple of block size

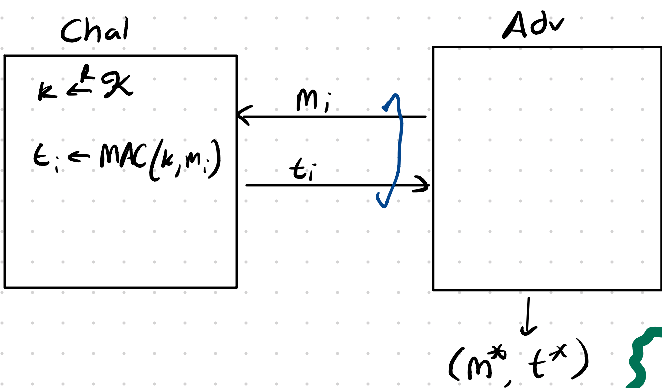


Message Authentication Code

Enc schemes give **NO** integrity protection! MAC does.

A MAC is a fn:

$$MAC : \mathcal{K} \times \mathcal{M} \rightarrow \{0,1\}^n$$



Adv wins if
$MAC(k, m^*) = t^*$
AND $m^* \notin \{m_1, \ldots, m_n\}$

Note: Some MAC schemes have a separate alg
$Ver(k, m, t) \rightarrow \{0,1\}$
to check tags. GMAC does.

GMAC [Simplified!]

First, define

Can compute in parallel. (See Horner's method)

128-bit blocks

$$GHASH(r, m_n \| \cdots \| m_1) := len(m) + m_1 r + m_2 r^2 + \cdots + m_n r^n \in GF(2^{128})$$

$$GMAC((k,r), m) := \begin{cases} IV \xleftarrow{R} \{0,1\}^{128} \\ (IV, F(k, IV) \oplus GHASH(r, m)) \end{cases}$$

$$GMACVer((k,r), m, (IV, t)) := \{ F(k, IV) \oplus GHASH(r,m) \stackrel{?}{=} t \}$$

Claim If $m \neq m'$

$$\Pr_{r \xleftarrow{R} \{0,1\}^{128}}[GHASH(r,m) = GHASH(r, m')] \leq \frac{n}{2^{128}}$$

Idea: $GHASH(r,m) = GHASH(r,m')$

$$\Rightarrow (m_1 - m_1')r + (m_2 - m_2')r^2 + \cdots + (m_n - m_n')r^n = 0$$

$$\Rightarrow P(r) = 0$$

Non-zero Poly of degree n. At most n roots!

Claim: GMAC is secure MAC.

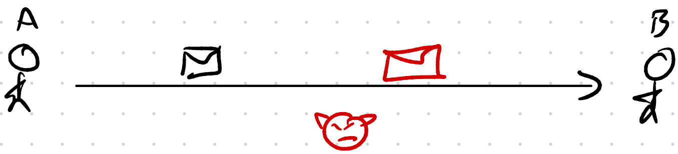
Idea: Adv has no info on r. So all adv m are indep of r. So adv forges w.p $\approx \frac{q^2 + n}{2^{128}}$ on q queries.

Stretch Break

CCA Security

CPA-secure: Adv can see encryption of msgs of its choice

↳ What if adv can see decryptions?



How could adv learn any info on decrypted msg?

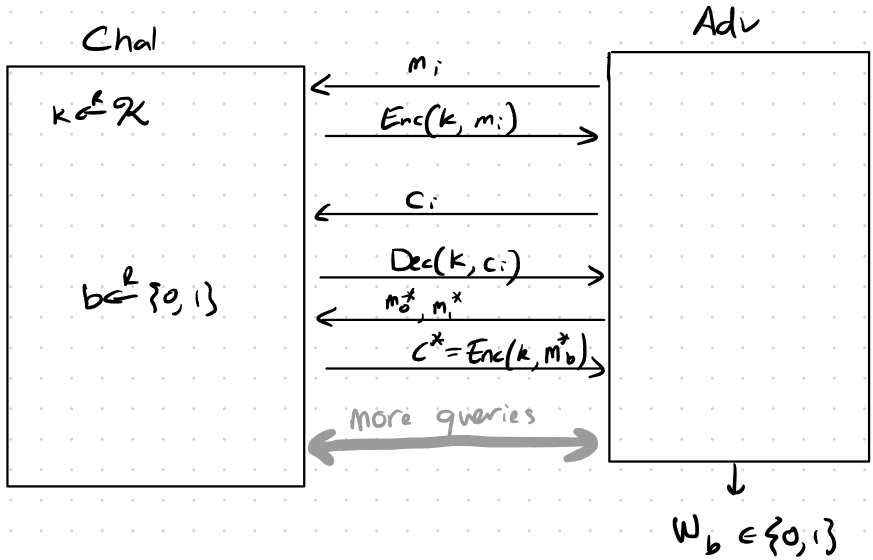
* B could reply w/ msg of varying len
* B could throw error
* B could reply in diff time
* B could perform other action

⟹ Two tasks

1. Stronger sec defn (CCA)
2. Stronger enc scheme.

Decryption routine for CCA scheme can output "fail"

CCA: Definition



↳ Adv may never ask for $Dec(k, c^*)$!

CCA Security Defn

(Enc, Dec) is CCA secure if $\forall$ eff adv A $\exists$ negl fn st. $\left|Pr[W_0 = 1] - Pr[W_1 = 1]\right| \leq$ negl.

Adv is very powerful here. AND adv's goal is very weak $\Longrightarrow$ Strong security

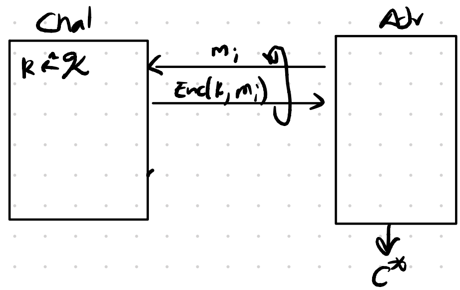
↳ Strongest possible??? No...

CCA Observations

* CCA sec ⇒ CPA sec ⇒ CCA must be rand.
* CCA cts cannot be "malleable" at all
  $c^* \curvearrowright \hat{c}^*$ ask for dec of $\hat{c}^*$
* CCA admits schemes that allow adv to cook up own cts

Authenticated encryption ("Gold standard" sec def)

(Enc, Dec) is AE if:
1) Is CPA secure and
2) Satisfies "ctext integrity"
   Adv wins if
   $c^* \notin \{c_1, \ldots, c_n\}$
   and $Dec(k, c^*) \stackrel{?}{=}$ reject



AE Security ⇒ CCA security.

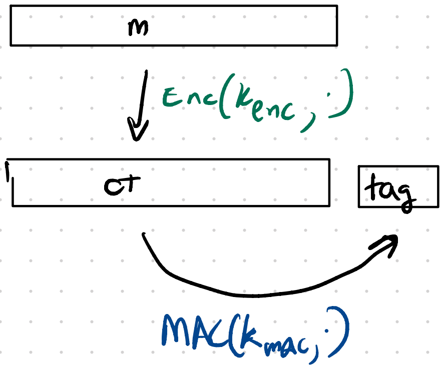
" ⇒ Msg integrity

AE is "gold standard" for enc security.
↳ AEAD = AE + associated (auth but not enc) data

Constructing AE schemes

"Encrypt then MAC" → As easy as it sounds

- Independent keys for both parts (PRF)
- AES-GCM is standard
  CTR mode + GMAC
- ChaCha-Poly1305 is another



To decrypt:
1) Check MAC on ct first. If bad, FAIL.
2) Then decrypt.

Encrypt-then-MAC is only safe way to combine enc & MAC

* AES-GCM ≃ AES-CTR then GMAC
* Also common = ChaCha20 + Poly1305 mac
* Well-designed crypto APIs handle this for you.

It's possible to construct AE directly from PRF (OCB)
↳ Less common. Why

Bad Ideas

MAC-then-encrypt

- ↳ Many many attacks (SSL)
- ↳ Basic idea: "padding oracle"

Encrypt-and-MAC

- ↳ Used in SSH (old versions)

Fundamental idea:

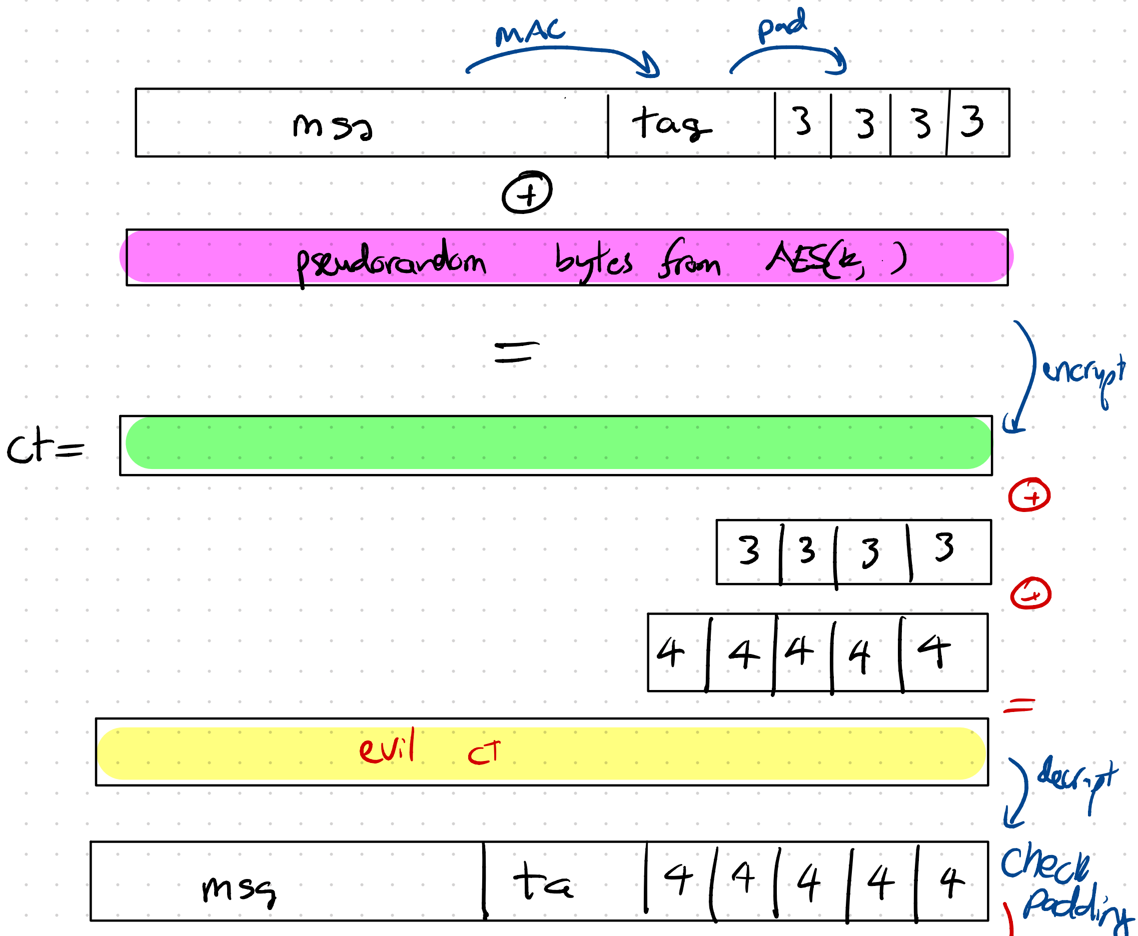
If enc scheme is only CPA secure, adv cannot learn <u>any</u> info on result of decrypting adv-chosen ct

MAC-then encrypt & encrypt-and-MAC violate this!

Why MAC-then-encrypt is bad:

* Some enc schemes (CBC mode) require plaintext be multiple of block size, e.g. 16 bytes
  ↳ Convenient & sometimes necessary

* Pad msg with n indicating "truncate n+1 bytes"



If adv can learn whether padding is valid, learns one byte of msg!