Applied Cryptography

thgoebel@ethz.ch

ETH Zürich, FS 2021

This documents is a **short** summary for the course *Applied Cryptography* at ETH Zurich. It is intended as a document for quick lookup, e.g. during revision, and as such does not replace reading the slides or a proper book.

We do not guarantee correctness or completeness, nor is this document endorsed by the lecturers. Feel free to point out any erratas.

Contents

List of Figures

1 Symmetric Cryptography

One-time pad Plaintext p, key k such that $|p| = |k|$. Ciphertext $c = p \oplus k$.

If k u.a.r. and only used once then the OTP is **perfectly secure**, i.e. $\Pr[P = p|C = c] = \Pr[P = p]$.

Note: keys can re-occur (as a result of random sampling) but they must not be re-used (i.e. the adversary must not be aware that the same key is used).

Issues: same lengths, key distribution, single use.

1.1 Block Ciphers

Block cipher A block cipher with key length k and block size n consists of two efficiently computable permutations[1]:

$$E : \{0,1\}^k \times \{0,1\}^n \mapsto \{0,1\}^n \quad D : \{0,1\}^k \times \{0,1\}^n \mapsto \{0,1\}^n$$

such that for all keys K D_K is the inverse of E_K (where we write E_K short for $E(K, \cdot)$).

Security notions Known plaintext attack, chosen plaintext attack, chosen ciphertext attack. Exhaustive key search on (P, C) pairs – no attack should be better, else we throw the cipher away.

Pseudo-randomness

- Adversary $\mathcal{A}$ interacts either with block cipher (E_K, D_K) or a truly random permutation (Π, Π^{-1}).
- A block cipher is called a **pseudo-random permutation PRP** if no efficient[2] $\mathcal{A}$ can tell the difference between E_K and Π (no access to the inverse).
- A block cipher is called a **strong-PRP** if no efficient $\mathcal{A}$ can tell the difference between (E_K, D_K) and (Π, Π^{-1}).

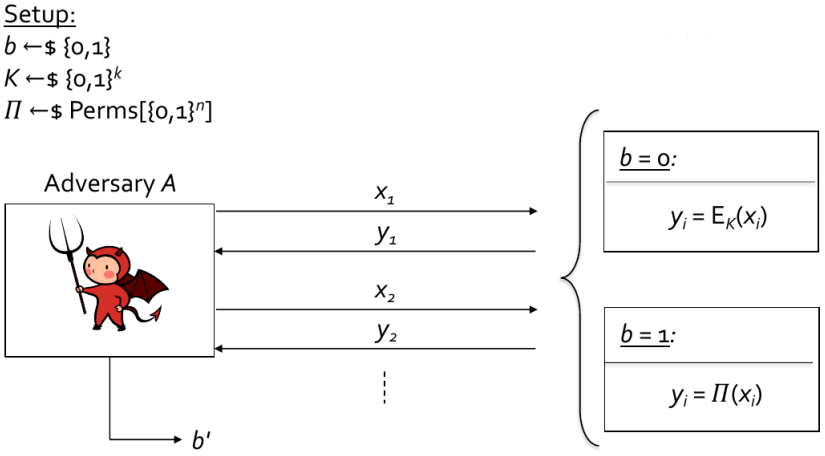

Figure 1: PRP game

[1] Encipher and decipher

[2] Quantified by runtime + number of oracle queries.

The advantage is defined as:

$$\mathbf{Adv}_E^{PRP}(\mathcal{A}) = 2 \cdot \left| \Pr[\text{Game } \mathbf{PRP}(\mathcal{A}, E) \Rightarrow \text{true}] - \frac{1}{2} \right|$$

where the probability is over the randomness of $b, K, \Pi, \mathcal{A}$. Note that:

$$\Pr[\text{Game } \mathbf{PRP}(\mathcal{A}, E) \Rightarrow \text{true}] = \Pr[b' = b]$$

Also see the Advantage Rewriting Lemma (1.2).

Constructing block ciphers In general: keyed round function that is repeated many times.

- Feistel cipher: halved blocks crossing back and forth, e.g. DES
- Substitution-permutation network: confusion + diffusion, e.g. AES

Electronic Code Book (ECB) mode Same plaintext always maps to the same ciphertext (deterministic). Thus serious leakage, don't use.

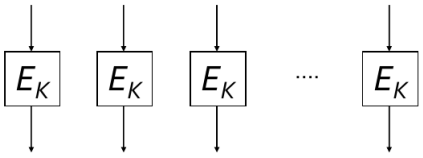

Figure 2: ECB mode

Cipher Block Chaining (CBC) mode Use u.a.r. IV/previous ciphertext block to randomise encryption.

A bit flip in C_i completely scrambles/randomises P_i and flips the same bit in P_{i+1}.

Caveats: non-random IV, padding oracle attack, ciphertext block collisions (after using the same key for $2^{n/2}$ blocks by the birthday bound).

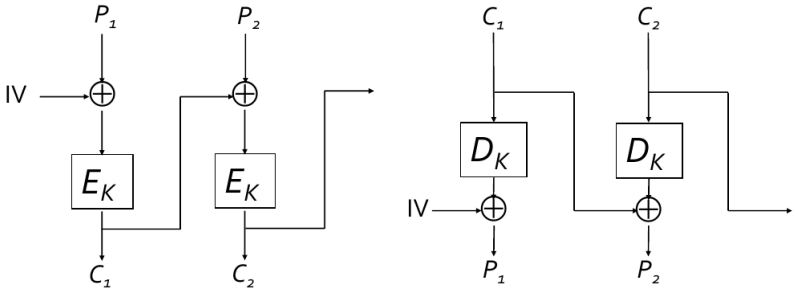

Figure 3: CBC mode (left: encipher, right: decipher)

Counter (CTR) mode Incrementing counter is encrypted with block cipher to produce a pseudorandom value to xor the plaintext block with.

Effectively a stream cipher producing OTP keys. E_K does not even need to be invertible. No padding needed, can just truncate the last block. A bit flip it C_i flips the same bit in P_i.

Caveats: counter must not repeat/wrap around (else xor of ciphertexts = xor of plaintexts).

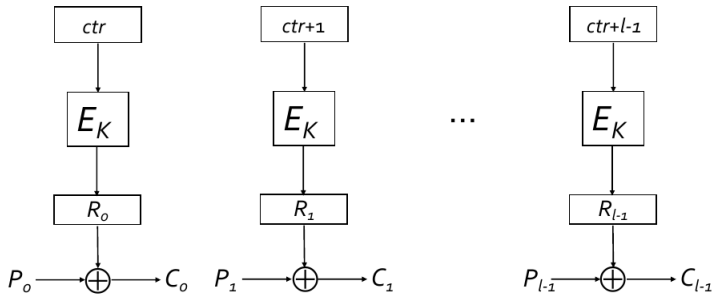

Figure 4: CTR mode

1.2 Symmetric Encryption

Symmetric Encryption Scheme is a triple $SE = (KGen, Enc, Dec)$. We have key space $\mathcal{K} = \{0,1\}^k$, message space $\mathcal{M} = \{0,1\}^{*3}$ and ciphertext space $\mathcal{C} = \{0,1\}^*$. For correctness, we have $Dec_K(Enc_K(m)) = m$.

IND-CPA Security Informally: computational version of perfect security – an efficient adversary cannot compute anything useful from a ciphertext (e.g. hide every bit of the plaintext). Equivalent to *semantic security*.

Formally: For any efficient adversary $\mathcal{A}$, given the encryption of one of two equal-length messages of its choice, $\mathcal{A}$ is unable to distinguish which one of the two messages was encrypted.

In the security game, $\mathcal{A}$ gets access to a *Left-or-Right encryption oracle*. The advantage of $\mathcal{A}$ is:

$$\mathbf{Adv}_{SE}^{IND-CPA}(\mathcal{A}) = 2 \cdot \left| \Pr[\text{Game } \mathbf{IND\text{-}CPA}(\mathcal{A}, SE) \Rightarrow \text{true}] - \frac{1}{2} \right|$$

Notes: Deterministic schemes **cannot** be IND-CPA secure (why?). CBC and CTR mode (if used properly) can be proven to be IND-CPA secure (assuming that Enc is a PRP-secure block cipher).

Caveats: No integrity. Says nothing about messages of non-equal length. No chosen ciphertexts.

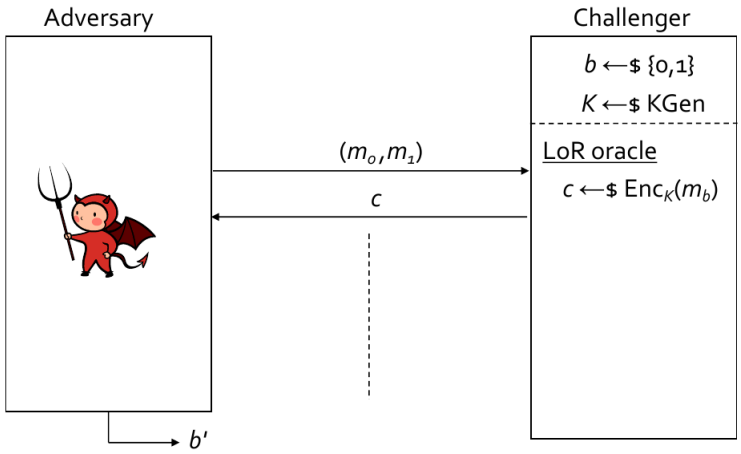

Figure 5: IND-CPA game

[3] In reality we might have a maximum plaintext length.

Advantage Rewriting Lemma Let b be a uniformly random bit and b' the output of some algorithm. Then:

$$\begin{aligned} 2\left|\Pr[b'=b]-\frac{1}{2}\right| &= \left|\Pr[b'=1|b=1]-\Pr[b'=1|b=0]\right| \\ &= \left|\Pr[b'=0|b=0]-\Pr[b'=0|b=1]\right| \end{aligned}$$

Difference Lemma Let Z, W_1, W_2 be events. If

$$(W_1 \wedge \neg Z) \text{ occurs if and only if } (W_2 \wedge \neg Z) \text{ occurs}$$

then

$$\left|\Pr[W_2]-\Pr[W_1]\right| \leq \Pr[Z]$$

In practice: Z is a bad event that rarely happens, W_1, W_2 are when $\mathcal{A}$ wins in security games G_1, G_2. Useful for *game hopping* proofs.

PRP-PRF Switching Lemma Let E be a block cipher. Then for any algorithm $\mathcal{A}$ making q queries:

$$\left|\mathbf{Adv}_E^{PRP}(\mathcal{A})-\mathbf{Adv}_E^{PRF}(\mathcal{A})\right| \leq \frac{q^2}{2^{n+1}}$$

1.3 Attacks

Padding Added before encryption to expand plaintext to a multiple of the block size, i.e. $pad(\cdot) : \{0,1\}^* \mapsto \{\{0,1\}^n\}^*$. Must be expanding (why?) and efficiently computable. May be either randomised or deterministic.
E.g. TLS padding: appends $t+1$ bytes of value t.

Problem: adversary can detect padding errors (explicit error messages, logs, timing differences).

Padding Oracle Given a ciphertext C, returns whether the padding of the decryption is valid or invalid. Leaks 1 bit of information. Kind of a chosen ciphertext attack, thus not covered by IND-CPA security! Main problem: no ciphertext integrity.

In practice: $\mathcal{A}$ needs 128 oracle queries on average to recover one plaintext byte. Repeat for all bytes and all blocks for full plaintext recovery. Additional practical details to consider. For TLS attacks, see Lucky 13 and POODLE.

Predictable IVs Random IVs are not just theoretically relevant for IND-CPA security of CBC mode. Consider the following steps (see Figure 7):

1. $\mathcal{A}$ queries LoR oracle with (P_0, P_1)
2. Oracle responds with $C = C_0||C_1$ where $C_1 = E_K(P_b \oplus C_0)$
3. $\mathcal{A}$ predicts next $IV = C_0'$
4. $\mathcal{A}$ queries $P_0 \oplus C_0 \oplus C_0'$
5. $\implies b = 0 \iff P_b = P_0 \iff C_1 = C_1' \implies$ breaks IND-CPA security

In practice: systems may use *IV chaining* (use the last ciphertext as the next IV, in order to avoid having to sample new randomness). E.g.: SSLv3, TLS 1.0, SSH in CBC mode. See also the BEAST attack on TLS.

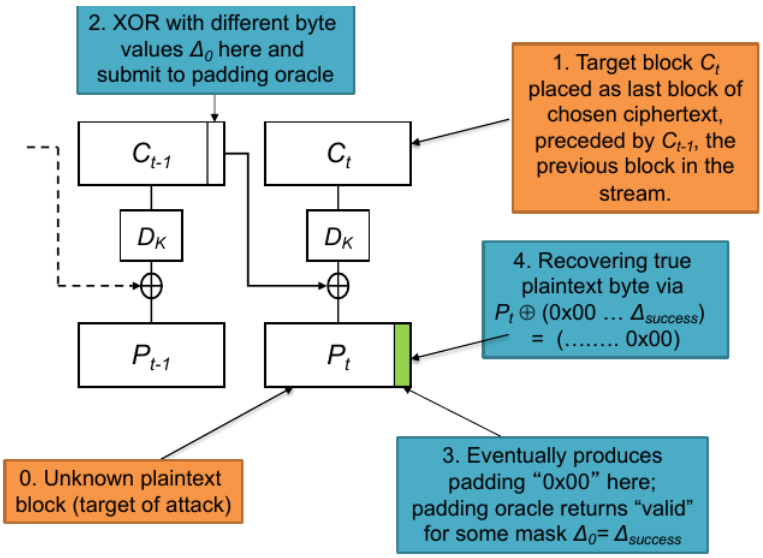

Figure 6: Padding oracle attack on CBC mode

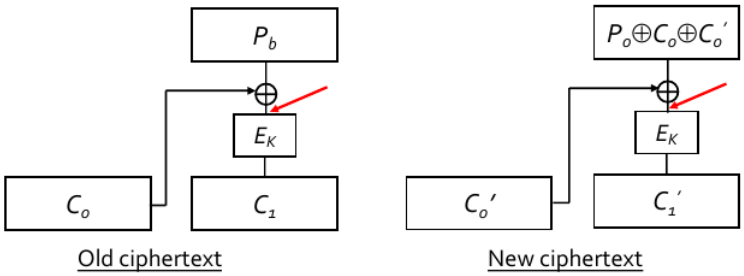

Figure 7: Attack on predictable IVs in CBC mode

1.4 Hash Functions

Cryptographic Hash Function An efficiently computable function $H : \{0,1\}^* \mapsto \{0,1\}^n$ that maps arbitrary-length inputs to fixed-length outputs ("*message digests*"). Not keyed!

Applications: MACs, signatures, key derivation, commitments.

Random oracle model Model under which hash functions are assumed to produced outputs that are uniformly random distributed. I.e. a hash function could be modelled by a random oracle. Very strong assumption!

Birthday paradox When drawing elements at random from a set of size s then after $\sqrt{s}$ trials we expect a collision with 39% probability. We quickly get to 99% with an additional constant factor.

Security goals Primary goals:

- *Pre-image resistance* (one-wayness): Given h, it is infeasible to find an m such that $H(m) = h$.
- *Second pre-image resistance*: Given m_1, it is infeasible to find an m_2 such that $H(m_1) = H(m_2)$.
- *Collision resistance*: It is infeasible to find any $m_1 \neq m_2$ such that $H(m_1) = H(m_2)$.

Secondary goals:

- *Near-collision resistance*: It is infeasible to find any $m_1 \neq m_2$ such that $H(m_1) \approx H(m_2)$ (e.g. hashes agree on most bits).
- *Partial pre-image resistance 1*: Given $H(m)$, it is infeasible to recover any partial information about m.
- *Partial pre-image resistance 2*: Given a target string t with $|t| = l$ it is infeasible to find an m such $H(m) = t||x$ (faster than with 2^l brute-force hash evaluations).

CR adversary:
Cannot quantify security over all efficient $\mathcal{A}$ (collisions exist, $\mathcal{A}$ can hardcode one). Instead, define "(t, ε)-CR adversaries", running in time t and with $\mathbf{Adv}_H^{CR}(\mathcal{A}) = \varepsilon$. Build reductions from there.

Relationships:

- Collision resistance $\implies$ second pre-image resistance
- Maybe: Collision resistance $\implies$ pre-image resistance – depending on how you define pre-image resistance (sampling from the domain or from the range?)

Generic attacks (in the ROM):
(Second) pre-image resistance: 2^n evaluations.
Collision resistance: $2^{n/2}$ evaluations! (by the birthday paradox)
$\implies$ e.g. SHA-1 with 160-bit outputs only achieves 80-bit security

Merkle-Damgård iterated hashing Constructs a hash function H from a *compression function* $h : \{0,1\}^k \times \{0,1\}^n \mapsto \{0,1\}^n$. Used e.g. in MD5, SHA-1, SHA-2.

Steps: pad message, split into k-bit chunks, repeatedly apply h to the chunks and IV/chaining values (see Figure 8).

Padding scheme:

$$m' = pad(m) = m||10^t||[len(m)]_k$$

where $[len(m)]_k$ is the k-bit encoding of the message length[4] and $0 \leq t < k$ is minimal.

Theorem: If h is collision resistant, then so is H.[5]

Length extension attack: Given $H(m)$ (but not m!) once can compute a valid hash $y = H(pad(m)||m'')$. This is problematic e.g. when construction MACs or KDFs from a hash function.

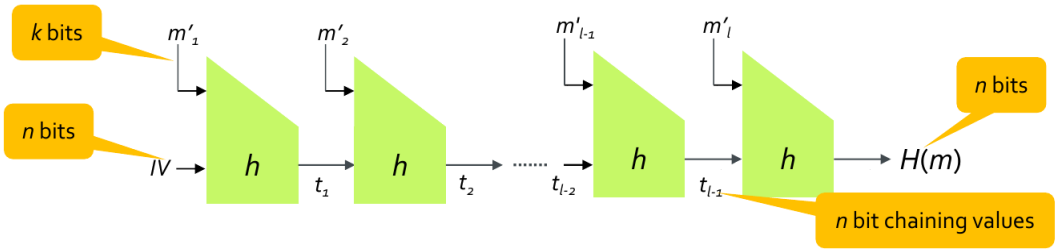

Figure 8: Merkle-Damgård transform

[4]This limits the maximum length of a message that can be hashed to $2^k - 1$.
[5]Note that the choice of padding scheme matters for the proof! Also note that the two IVs must be the same for a *full collision*. The much weaker form of two colliding messages for different IVs is called a *freestart collision*.

Constructing compression functions from block ciphers E.g. Davies-Meyer, Matyas-Meyer-Oseas, Miyaguchi-Preneel constructions. Use message as the key (need fast rekeying!) and (some variation of) chaining value as "plaintext" input.

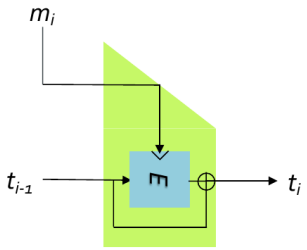

Figure 9: Davies-Meyer construction

Sponge construction Different design approach. Centered around 2 phases: absorbing + squeezing. Key ideas: giant bit permutation F, not inputting/outputting entire state. Variable length output.

E.g. used in SHA-3/Keccak: SHA-3$(m) = out_1||out_2||out_3||....$

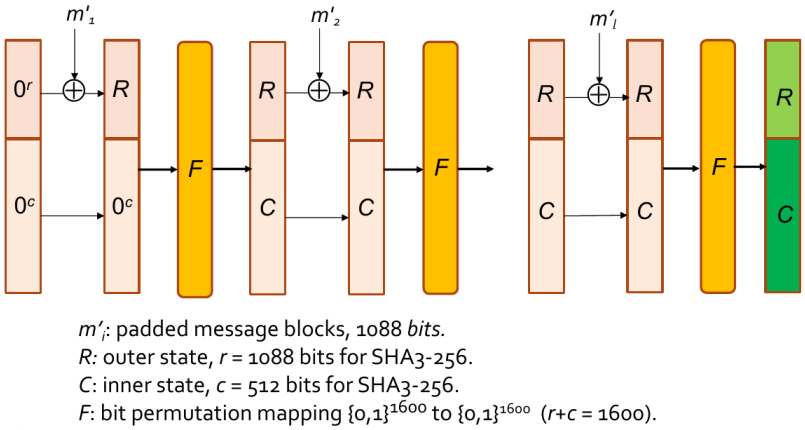

Figure 10: Sponge construction: absorbing

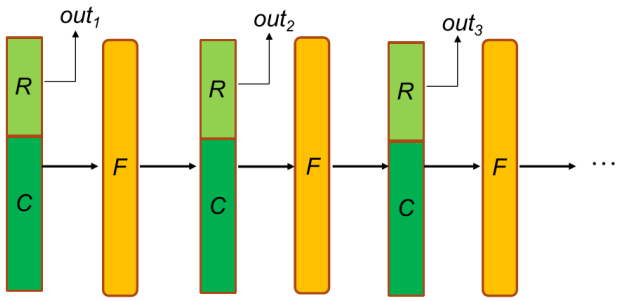

Figure 11: Sponge construction: squeezing

1.5 Message Authentication Codes MACs

1.6 Authenticated Encryption