Radbound University

Block cipher modes of use

Cryptography, Autumn 2021

Lecturers: J. Daemen, B. Mennink

September 28, 2021

Institute for Computing and Information Sciences
Radboud University

Outline

Overview of symmetric cryptography

Modes for encryption

Stream encryption with block ciphers

Provable security of modes

Authentication with block ciphers

Overview of symmetric cryptography

Symmetric cryptography operations

- Basic operations
  - encryption
  - MAC computation
  - authenticated encryption (including sessions)
- Require a key shared between sender and receiver
- Auxiliary operations
  - cryptographic hashing
  - deterministic random bit generation (DRBG)
  - ...
- Not really symmetric crypto but often categorized as such
  - true random generators
  - secret sharing for key management

Need for secret keys in symmetric cryptography

- Symmetric stands for
  - same key for encryption and decryption
  - same key for MAC generation and verification
- Basic operations achieve following:
  - reduce problem of securing (big) data
  - to problem of securing (small) keys
- A secure solution requires secrecy of keys
  - key generation requires qualitative random generator
  - key transfer between entities requires other keys
  - modules performing crypto shall not leak keys
  - many potential weaknesses

Limit to security: exhaustive key search

- Exhaustive key search
  - given some plaintext and corresponding ciphertext ($M = 1$) ...
  - trying all different keys (N)
- Single-target attack: one particular k-bit key K
  - success prob. after N trials: $N2^{-k}$
  - expected effort $N \approx 2^k$
  - (implicit) security claim: this should be best attack
  - so a k-bit key limits security strength to k bits
- Multi-target attack:
  - attacker is happy if she finds one key out of n keys K_i
  - relevant in many cases
  - e.g., if keys K_i are on badges giving access to a building

Limit to security: multi-target exhaustive key search

- Multi-target attack setting example
  - attacker knows $Z_i = \mathsf{SC}_{K_i}(D = 1, \ell)$ for n keys K_i
- Attack:
  - guess K' and compute $Z' = \mathsf{SC}_{K'}(D = 1, \ell)$
  - until $Z' \in \{Z_i\}$: success
  - success probability per trial: $\geq n2^{-k}$
  - expected effort $N \approx 2^k/n$,
- Security erosion: 128-bit key offers much less than 128-bit strength
  - Security strength decreases to $k - log_2(n)$
- Can be prevented with globally unique diversifier: *global nonce*
  - e.g., key ID_i plus message counter Nr: $Z_i = \mathsf{SC}_{K_i}(ID_i \| Nr, \ell)$
  - or, random string R of sufficient length $Z_i = \mathsf{SC}_{K_i}(R, \ell)$

Security erosion

Security strength is smaller than key if multi-target attacks are possible

Modes for encryption

Block cipher modes for encryption

- DES can encipher 8-byte messages, AES 16-byte messages
  - what about longer and shorter messages?
  - two approaches: block encryption and stream encryption
- Block encryption modes
  - split the message in blocks
  - after padding last *incomplete* block if needed
  - apply permutation B_K to blocks in some way
- Stream encryption modes
  - build a stream cipher with a block cipher as updating function F or output function f

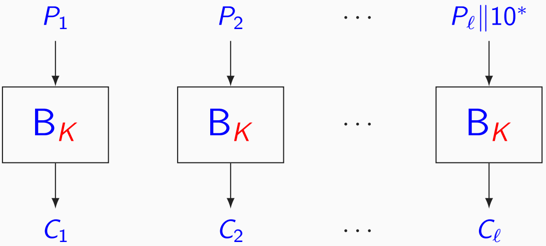
Block encryption modes

- Electronic Code Book (ECB) mode
  - *we consider only 16-byte messages*
  - longer messages are split in 16-byte blocks
  - shorter messages padded to 16 bytes
  - same for *last incomplete block*
- Cipher Block Chaining (CBC) mode
  - *ECB randomized with what's available*
  - requires also split in 16-byte blocks and padding
- Due to padding, cryptogram is longer than message

Intermezzo: padding

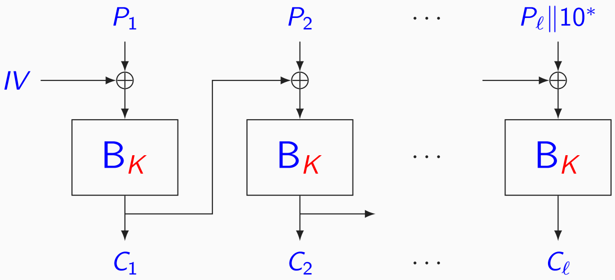
- Simplest padding: append zeroes
  - up to length multiple of block length (e.g., 16 bytes)
  - shortest possible padding
  - as such not usable for our purposes because it is not injective
- Decryption of cryptogram gives *padded* message
- Recovering message requires removing padding
  - send along message or padding length with cryptogram, or
  - impose padding is injective (or reversible)
- Simplest reversible padding: a single 1 and then zeroes
  - extends message in all cases
  - turns 16-byte message into 32-byte string
- Padding with exotic requirements
  - random-length padding: to hide message length
  - random padding: to add entropy
- Badly designed padding is often source of security problems

Electronic CodeBook Mode (ECB)



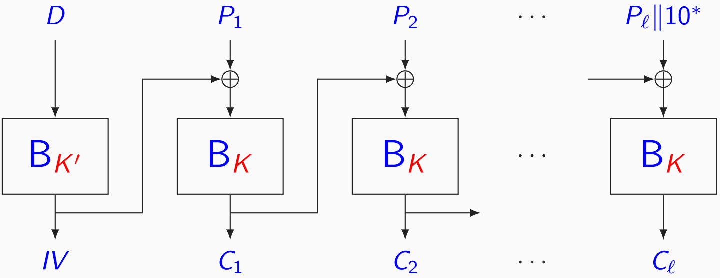
- Advantages
  - simple
  - parallelizable
- Limitation: equal plaintext blocks $\rightarrow$ equal ciphertext blocks:
  - likely to happen in low-entropy messages
  - problem in padded last block, that can be a single byte

Cipher Block Chaining mode (CBC)



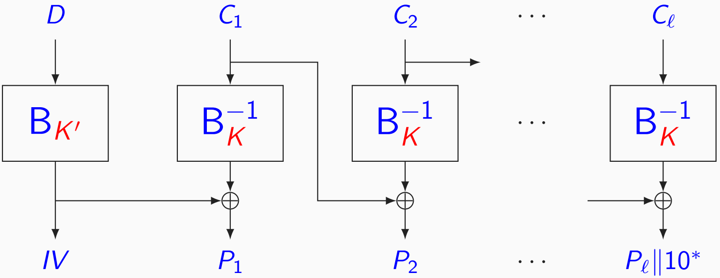
- *ECB with plaintext block randomized by previous ciphertext block*
- First plaintext block randomized with **random** Initial Value (IV)
- Solves leakage in ECB (partially):
  - equal plaintext blocks do not lead to equal ciphertext blocks
  - requires randomly generating and transferring IV

Cipher Block Chaining mode (cont'd)



- Replacing IV randomness by D nonce requirement: $IV = B_{K'}(D)$
  - with different key K' to avoid chosen-plaintext attacks
- CBC properties
  - encryption strictly serial
  - IV or diversifier D must be managed and transferred

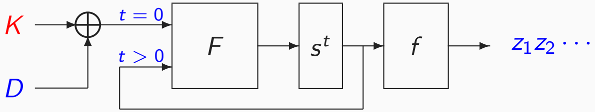
Cipher Block Chaining decryption



- Decryption can be done in parallel
- Bottom line
  - *we still need a nonce despite doing block encryption*
  - but ok, nonce re-use leaks less information

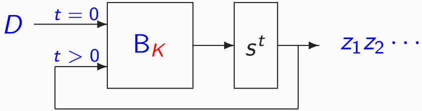
Stream encryption with block ciphers

Stream encryption with a block cipher



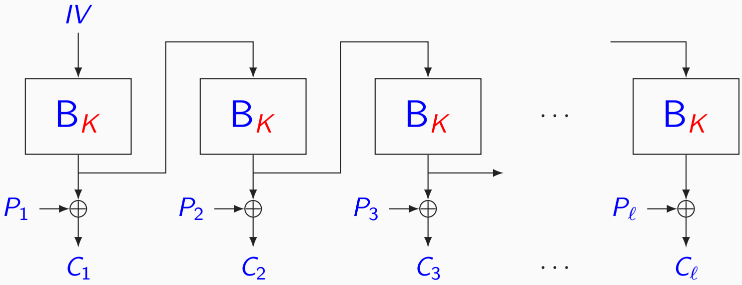
- Remember structure of iterative stream ciphers:
  - state update function $s^t = F(s^{t-1})$
  - output function $z_t = f(s^t)$
- Stream encryption modes of a block cipher:
  - use a block cipher for F or f

Output FeedBack mode (OFB)



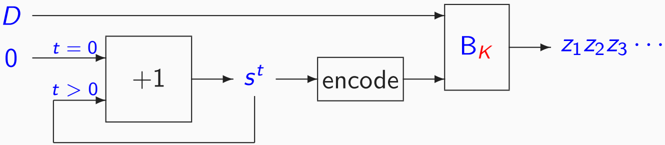
- $F = \mathsf{B}_K$, so $s_t \leftarrow \mathsf{B}_K(s_{t-1})$
- f is identity: $z_t \leftarrow s^t$
- Initialization: storage of K and $s_0 \leftarrow D$ (often called IV)
- Properties:
  - strictly serial
  - cycle lengths not known in advance
  - no need for B_K^{-1} (valid for all stream encryption)

OFB encryption presented in the classical way



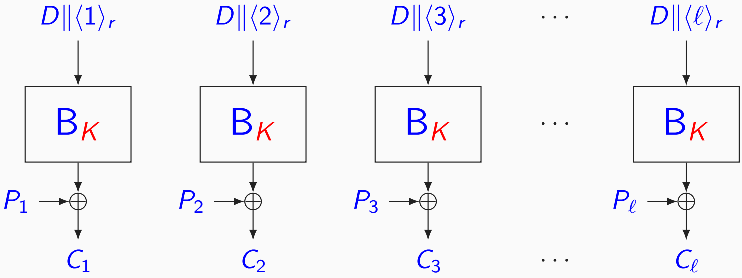
Note: the diversifier is often denoted as IV

Counter mode

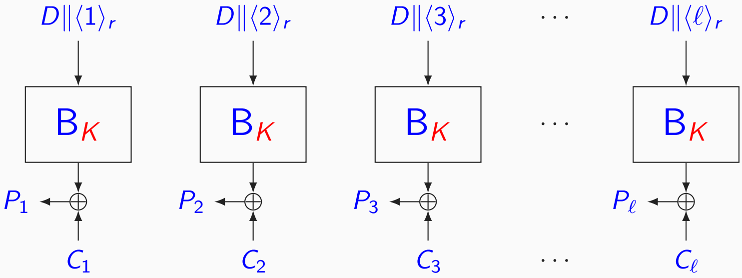


- F: interpret s^t as integer and add 1: $s^t = s^{t-1} + 1$
- $f = \mathsf{B}_K$, so $z_t = \mathsf{B}_K(D \|$ encoding of $s^t)$
- Initialization: storage of K and $s_0 \leftarrow 0$
- Properties:
  - fully parallelizable
  - number of blocks $\ell = |Z|$ is at most $2^{b-|D|}$
  - no risk of short cycles

Counter mode encryption presented in the usual way



Counter mode decryption presented in the usual way



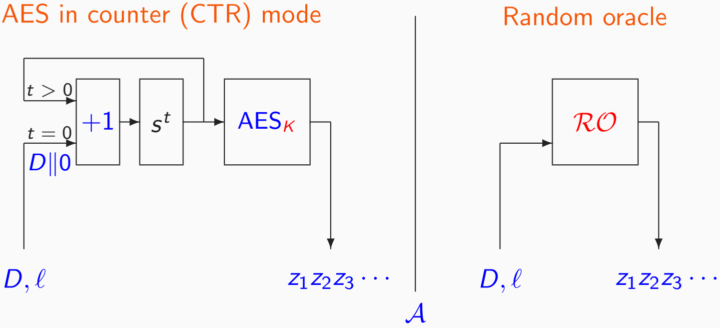
Encryption modes: overview

	ECB	CBC	OFB	CTR
parallel encryption	✓	—	—	✓
parallel decryption	✓	✓	—	✓
inverse free	—	—	✓	✓
absence of message expansion	—	—	✓	✓
tolerant to bit flips in $C \rightarrow P$	—	—	✓	✓
graceful degradation if nonce violation	n/a	✓	—	—

Provable security of modes

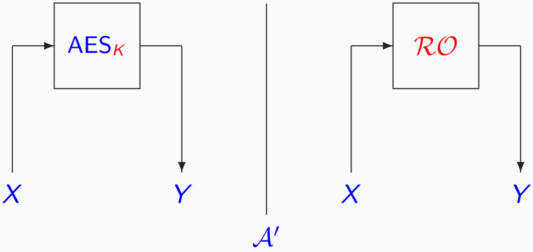
Provable security of a counter mode scheme

(counter mode depicted slightly differently for compactness)
(calls to internals symbolizing computational complexity omitted)



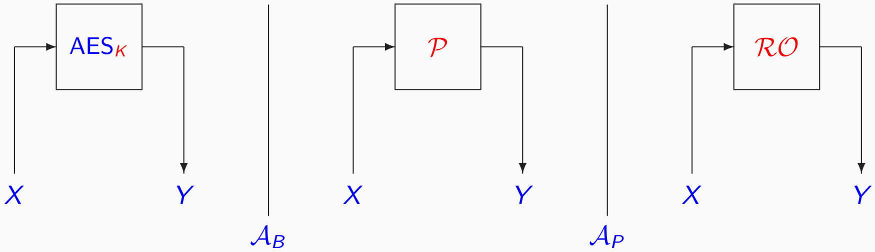
- Security of concrete scheme
  - advantage in distinguishing real and ideal world
  - denoted as $\mathrm{Adv}_{\mathcal{A}}(\mathsf{CTR}_{\mathsf{AES}_K}, \mathcal{RO})$
- Hard to analyze as such ...
  - we break this into simpler problems with some techniques
  - this set of techniques form the discipline of *provable security*

Provable security: simulation



- We replace $\mathcal{A}$ by an adversary $\mathcal{A}'$ that has more power
- $\mathcal{A}$ can be *simulated* by $\mathcal{A}'$
  - response to any query sent by $\mathcal{A}$ can be obtained by $\mathcal{A}'$
  - being asked to distinguish, $\mathcal{A}'$ can just ask $\mathcal{A}$
  - ... as she could do it herself
  - advantage of $\mathcal{A}'$ cannot be smaller than that of $\mathcal{A}$:

$$\mathrm{Adv}_{\mathcal{A}}(\mathsf{CTR}_{\mathsf{AES}_K}, \mathcal{RO}) \leq \mathrm{Adv}_{\mathcal{A}'}(\mathsf{AES}_K, \mathcal{RO})$$



- We add a step in between, here a random permutation $\mathcal{P}$
  - Adversary $\mathcal{A}_B$ distinguishes between AES_K and $\mathcal{P}$
  - Adversary $\mathcal{A}_P$ distinguishes between $\mathcal{P}$ and $\mathcal{RO}$
- Triangle inequality:

$$\mathrm{Adv}_{\mathcal{A}'}(\mathsf{AES}_K, \mathcal{RO}) \leq \mathrm{Adv}_{\mathcal{A}_B}(\mathsf{AES}_K, \mathcal{P}) + \mathrm{Adv}_{\mathcal{A}_P}(\mathcal{P}, \mathcal{RO})$$

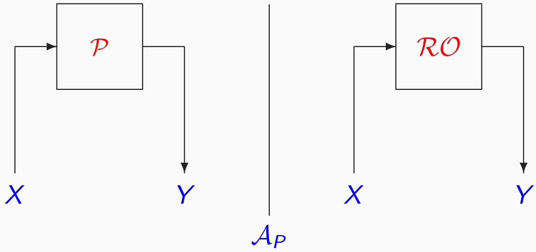
Separation of primitive cryptanalysis and mode security proofs

$$\mathrm{Adv}_{\mathcal{A}'}(\mathsf{AES}_K, \mathcal{RO}) \leq \mathrm{Adv}_{\mathcal{A}_B}(\mathsf{AES}_K, \mathcal{P}) + \mathrm{Adv}_{\mathcal{A}_P}(\mathcal{P}, \mathcal{RO})$$

The advantage has two components:

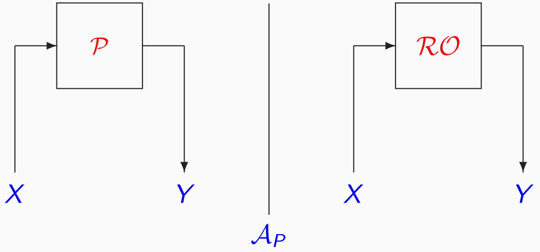
- Advantage of the primitive
  - here: PRP-security of AES
  - domain of *cryptanalysis*
  - cannot be proven, only assumed, claimed and challenged
- Advantage of the mode assuming ideal component
  - here: (CTR mode of a) random permutation $\mathcal{P}$
  - domain of *provable security*
  - bounds can be proven using probability theory

Provable security of counter mode as such



- Difference in behaviour between $\mathcal{P}$ and $\mathcal{RO}$
  - $\mathcal{P}$ returns uniformly random responses, with restriction that they don't collide
  - $\mathcal{RO}$ returns uniformly random responses
- This implies that $\mathcal{A}_P$ can distinguish $\mathcal{P}$ from $\mathcal{RO}$ if and only if
  - she is speaking to $\mathcal{RO}$ AND
  - $\mathcal{RO}$ returns colliding outputs

Provable security of counter mode as such (cont'd)



- After queries, $\mathcal{A}_P$ returns 1 if there was a collision and 0 otherwise

  $\mathrm{Adv}_{\mathcal{A}_P}(\mathcal{P}, \mathcal{RO}) = |\Pr(\mathcal{A}_P = 1 \mid \mathcal{RO}) - \Pr(\mathcal{A}_P = 1 \mid \mathcal{P})| = \Pr(\text{coll.} \mid \mathcal{RO})$

- We have

$$\Pr(\text{coll.} \mid \mathcal{RO}) \leq \binom{M}{2} 2^{-128} \leq M^2 2^{-129}$$

- Advantage gets close to 1 when $M \approx 2^{64}$: the *birthday bound*

Authentication with block ciphers

Message authentication code (MAC) functions

- MAC: cryptographic checksum
  - input: key K and arbitrary-length message m
  - output: tag (aka MAC) T with some length ℓ
- Applications:
  - message authentication: append tag to message
  - entity authentication: compute tag over challenge

We can formally write: $T \leftarrow \mathrm{MAC}_K(m)$

Two types of MAC function (online) queries:

- Generation: give m and get $T \leftarrow \mathrm{MAC}_K(m)$
- Verification: give (m, T) and get 1 if $T = \mathrm{MAC}_K(m)$ and else 0

Security goal

MAC forgery

Generating a couple (m, T) such that tag verification returns 1 without knowing K and without querying tag generation with m

- Security goal of a MAC function: forgery should be hard. How hard?
- Ideal MAC function:
  - tags fully unpredictable when keyed with unknown K
  - ...except that same message returns same tag
  - like a random oracle with ℓ-bit output!
- Success probability of forgery after M attempts for $\mathcal{RO}$: $M2^{-\ell}$
- Try (m, T) with same m and different T until we hit the right tag

So we want our keyed MAC function to be like a random oracle

Pseudorandom function (PRF) security of a MAC function

MAC() is PRF-secure if $\mathrm{MAC}_K(m)$ is hard to distinguish from $\mathcal{RO}$

Note: same security concept as for a stream cipher

Implications of PRF-security bound of a MAC function

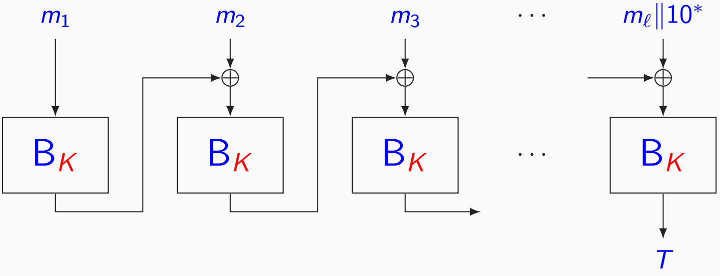
PRF-advantage of a MAC function

$\mathrm{Adv}_{\mathcal{A}}(\mathsf{MAC}_K, \mathcal{RO}) = |\Pr(\mathcal{A} = 1 \mid \mathsf{MAC}_K) - \Pr(\mathcal{A} = 1 \mid \mathcal{RO})|$

A (claimed) advantage $\mathrm{Adv}_{\mathcal{A}}(\mathsf{MAC}_K, \mathcal{RO}) \leq \epsilon(M, N)$ says something about the success probability of forgery

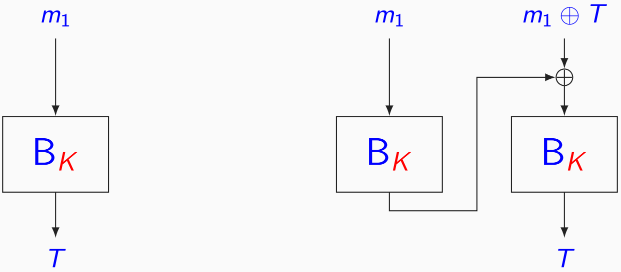
- Recipe for distinguishing adversary $\mathcal{A}$ based on forging ability:
  (1) Spend resources N and M on trying to generate forgery
  (2) If it works, return 1, else return 0
- $\Pr(\mathcal{A} = 1 \mid \mathcal{RO}) = \Pr(\text{forgery success for } \mathcal{RO}) \leq M2^{-\ell}$
- $\Pr(\mathcal{A} = 1 \mid \mathsf{MAC}_K) = \Pr(\text{forgery success for } \mathsf{MAC}_K)$
- Due to the claim: $\Pr(\text{forgery success for } \mathsf{MAC}_K) \leq M2^{-\ell} + \epsilon(M, N)$

Cipher Block Chaining MAC mode (CBC-MAC)



- Observation: in CBC ciphertext block C_i depends on m_0 to m_i
- Idea:
  - apply CBC encryption with zero IV to (padded) message
  - take tag equal to last ciphertext block
  - throw away other blocks (essential for security)
- This is the basis for most block-cipher based MAC functions

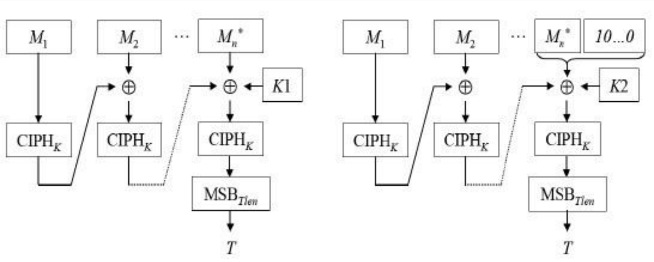
CBC-MAC weakness: length extension



- Distinguishing from random oracle $\mathcal{RO}$ in two queries:
  - query m_1 returns $T = B_K(m_1)$
  - query $m_1 \| m_2$ with $m_2 = m_1 \oplus T$ returns

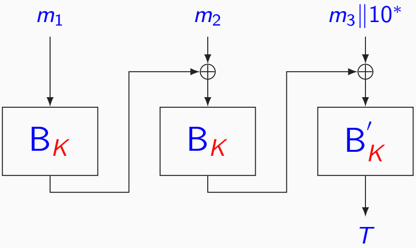
$$B_K(m_2 \oplus B_K(m_1)) = B_K(m_1 \oplus T \oplus B_K(m_1)) = B_K(m_1) = T$$

- A random oracle will give two completely unrelated tags
- Note: attack ignores padding, but this can be dealt with
- Truncating the tag T helps (somewhat) against this attack



- Trick: avoid length-extension problem by *doing something different at the end*: finalization
- Here: addition of a *subkey* before last application of B_K
- Advantage in distinguishing this from $\mathcal{RO}$ assuming random $\mathcal{P}$
  - birthday bound $M^2 2^{-(b+1)}$ due to *inner collisions*
  - see next slide

Security of CBC-MAC based modes: inner collisions



- Consider CBC-MAC with *finalization* B'_K, e.g., C-MAC
- Distinguishing this from a $\mathcal{RO}$:
  - query for many 3-block inputs $m^{(i)}$ of the form $m_1 m_2 m_3$
  - m_1 and m_2 different in each query, m_3 always the same
- Collision for $i \neq j$ at input of B'_K gives colliding tags
  - probability $\approx M^2 2^{-(b+1)}$ with M number of queries
  - detect *internal collision* by tag collision plus some check queries
  - then $\forall m'$: $m^{(i)}\|m'$ gives same tag as $m^{(j)}\|m'$
- $\mathcal{RO}$ has no internal collisions

Summary

Summary

- Block ciphers are versatile:
  - block encryption modes: e.g., ECB and CBC
  - stream encryption modes: e.g., OFB and counter
  - MAC computation modes: e.g., CBC-MAC and C-MAC
- Inverse permutation only used in block encryption modes
- Security analysis of cryptographic schemes splits in two parts
  - primitives must be cryptanalyzed, no security proofs
  - modes can be proven secure with probability theory
- Most modes only secure up to *birthday bound*
  - processing $2^{b/2}$ blocks with same key will show non-ideal behaviour