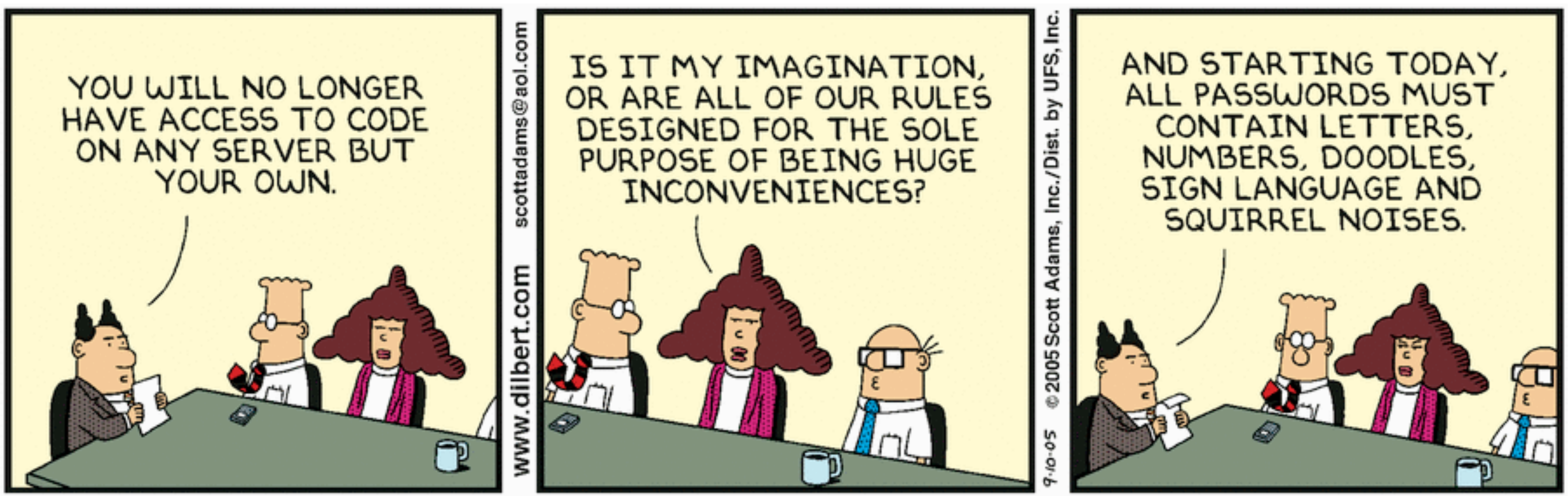
Message Authentication and Digital Signatures

February 10, 2022



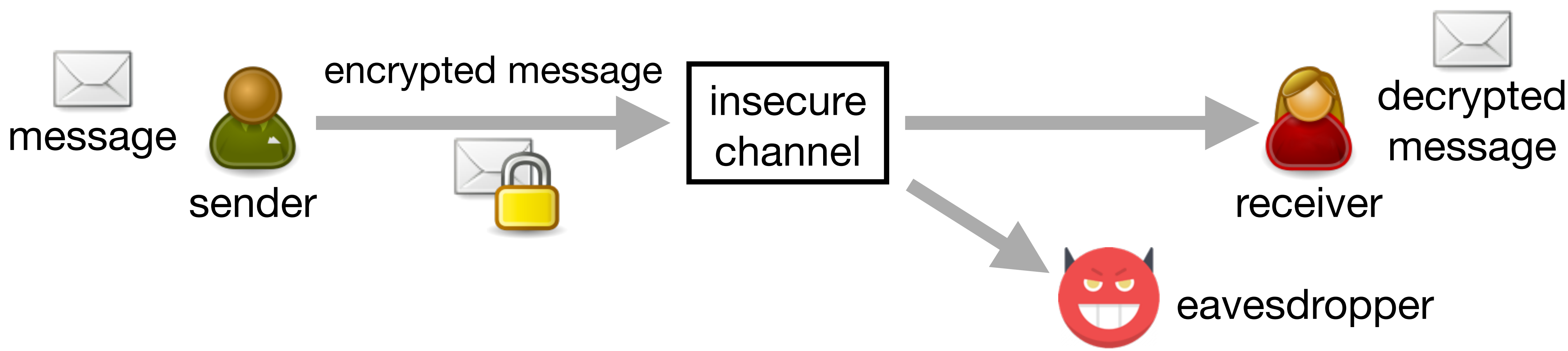
Homework Assignment & Today

- Homework 1
  - available on **Blackboard**
  - based on cryptography lectures, **requires Python or Java programming**
  - due **February 20th** (Sunday) at 11:59pm
- Today: **integrity**
  - message authentication
  - digital signatures (quick overview)

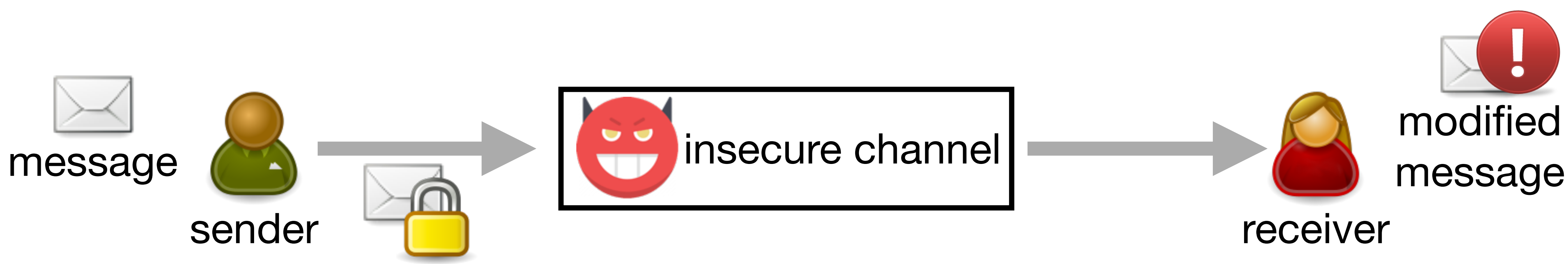
Feedback: `https://forms.gle/JGbNCmCsU69iWaTv8`

Attacks Against Communication

- *Previously*: protecting **confidentiality** against **passive attacks**

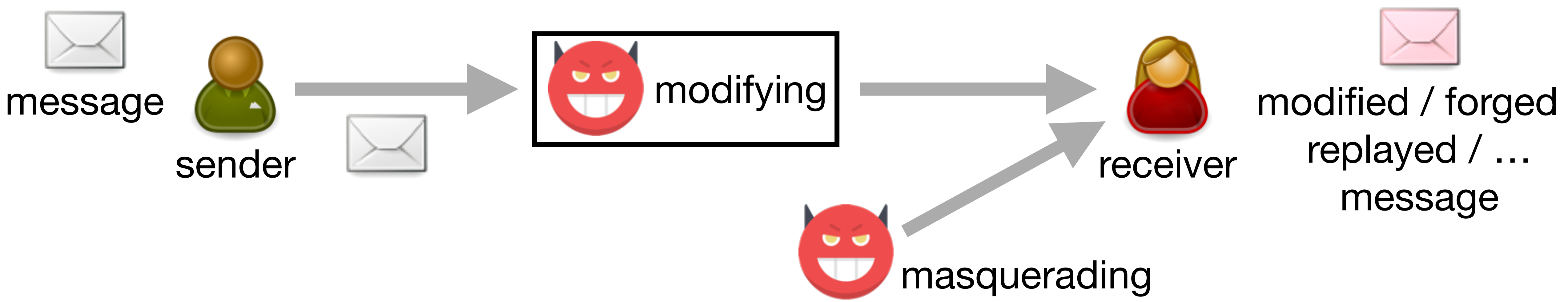


- Active attacks



- Data integrity: information cannot be modified in an **unauthorized** and **undetected** manner

Reminder:
Active Attacks in a Communication Channel

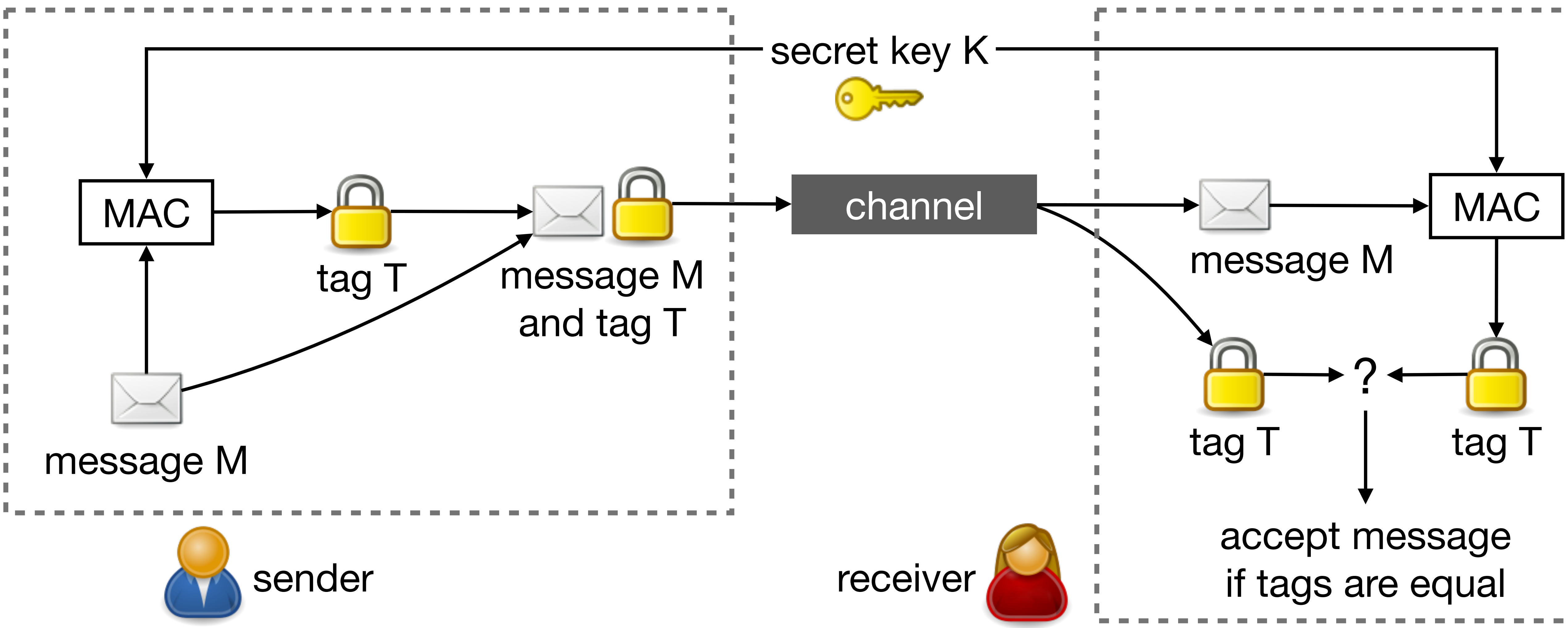


- **Content modification:** changing the contents of a message
- **Sequence modification:** changing the sequence of messages, including deleting some of them
- **Timing modification:** delay or replay messages
- **Masquerade** (i.e., forgery): inserting messages of fraudulent source

Message Authentication Codes

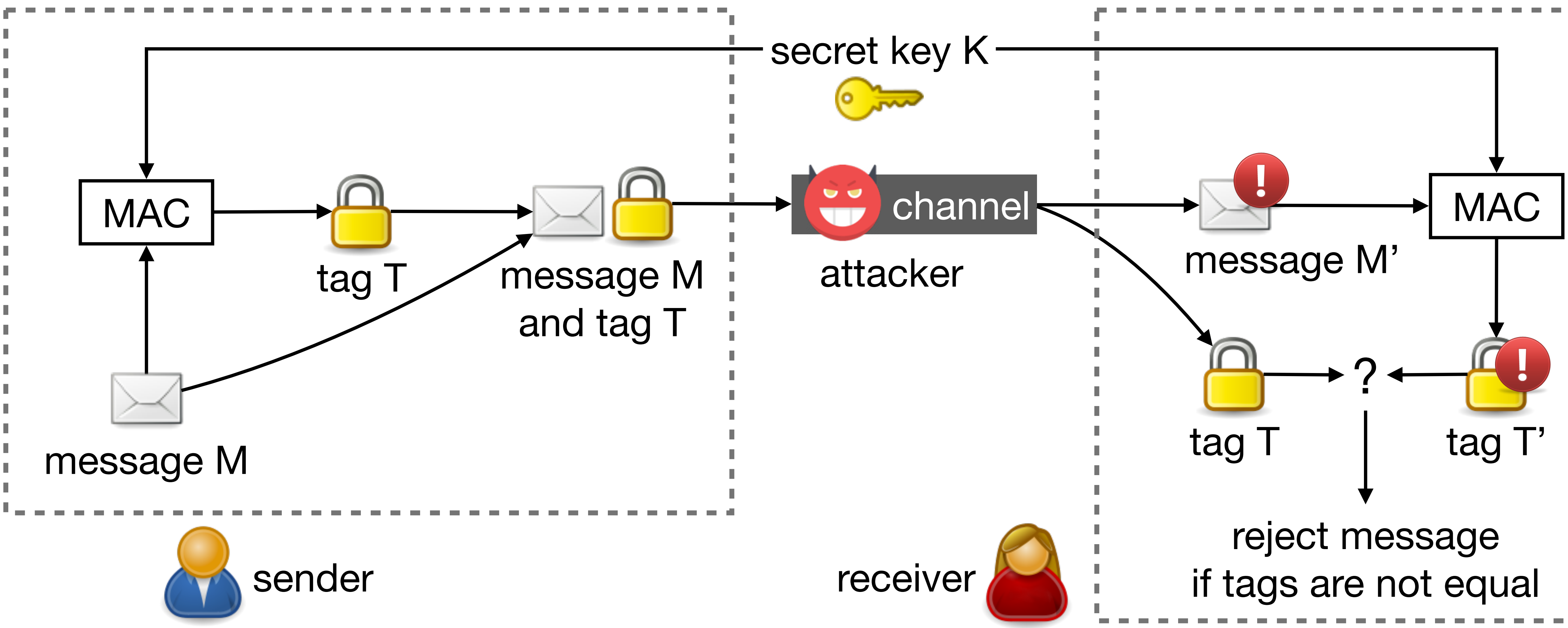
Message Authentication Code

- <u>Message authentication code</u> MAC(K, M):
  takes a **secret key K** and an **arbitrary-length input M**, and produces a **tag T**
  - can be efficiently and deterministically computed given key **K** and message **M**
  - cannot be computed efficiently given only a message **M** (without key **K**)



Message Authentication Code

- <u>Message authentication code</u> MAC(K, M):
  takes a **secret key K** and an **arbitrary-length input M**, and produces a **tag T**
  - can be efficiently and deterministically computed given key **K** and message **M**
  - cannot be computed efficiently given only a message **M** (without key **K**)



Message Authentication Code Properties

- MAC(K, M)
  - looks like a pseudorandom function to the attacker
  - does not need to be invertible ← both sender and receiver use it the same way
- Correct tag proves
  - **authenticity**: message is from an entity that knows the secret key
  - **integrity**: message has not been altered by an entity that does not know the key
- Detecting timing or sequence attacks:
  include **timestamp** or **sequence number** in the message
- MAC can be **independent of encryption**:
  we can provide only integrity, only confidentiality, or both

Brute-Force Attacks

- Tag forging

  What is the probability that a random tag matches a message?

  success depends on the length of the tag
  (independent of the message length)

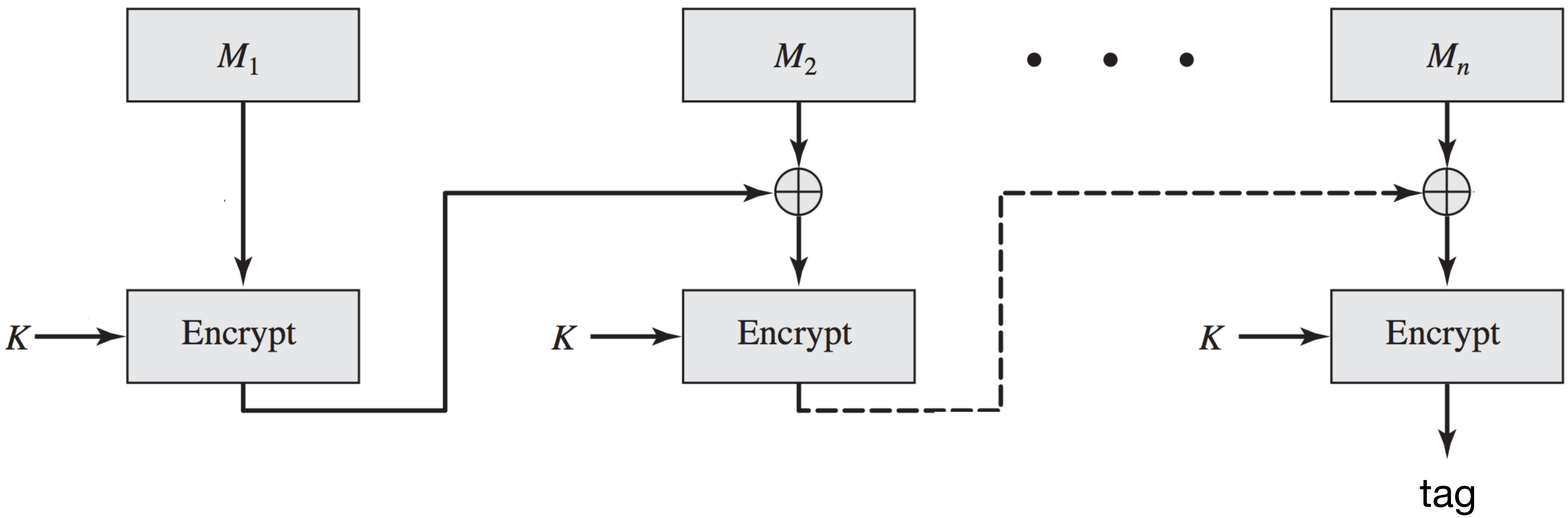
  - too short → high probability of an arbitrary tag matching a modified message (with an **n**-bit tag, probability is $\mathbf{2^{-n}}$)
  - too long → consumes bandwidth

- Key search
  - suppose that key length is **k bits**
  - finding the right key takes on average $\mathbf{2^{k-1}}$ steps

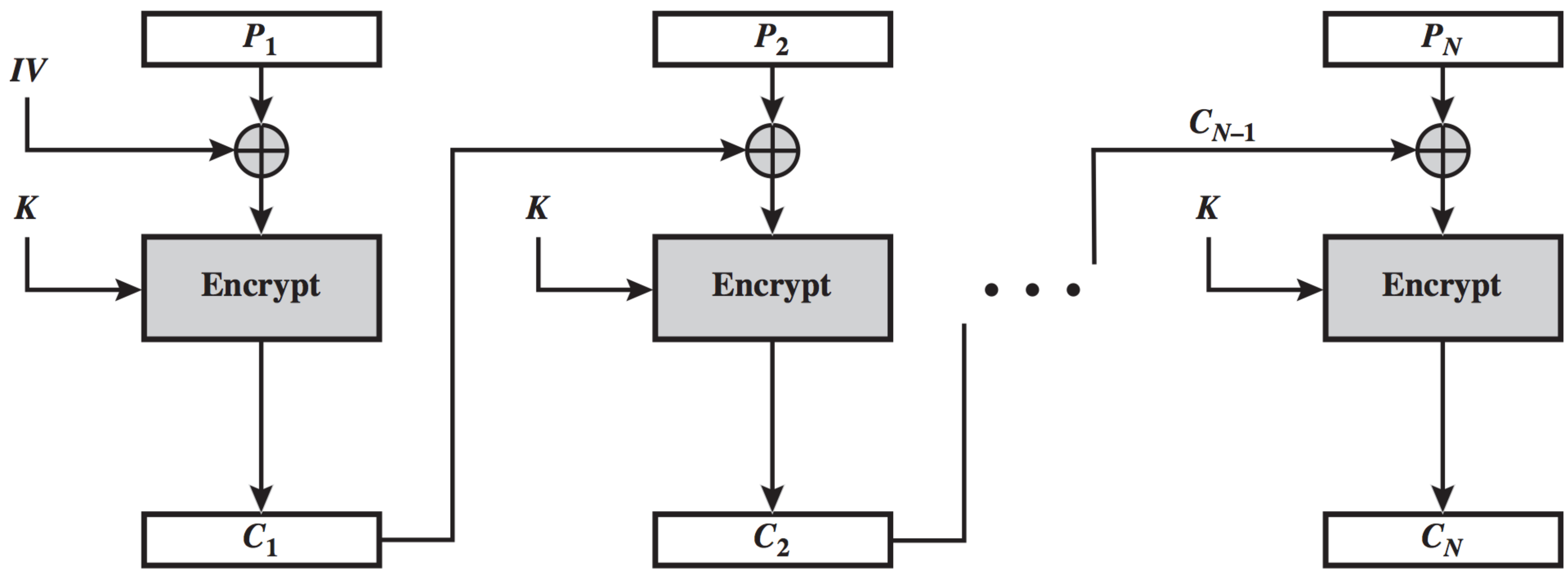
MAC Based on Block Ciphers

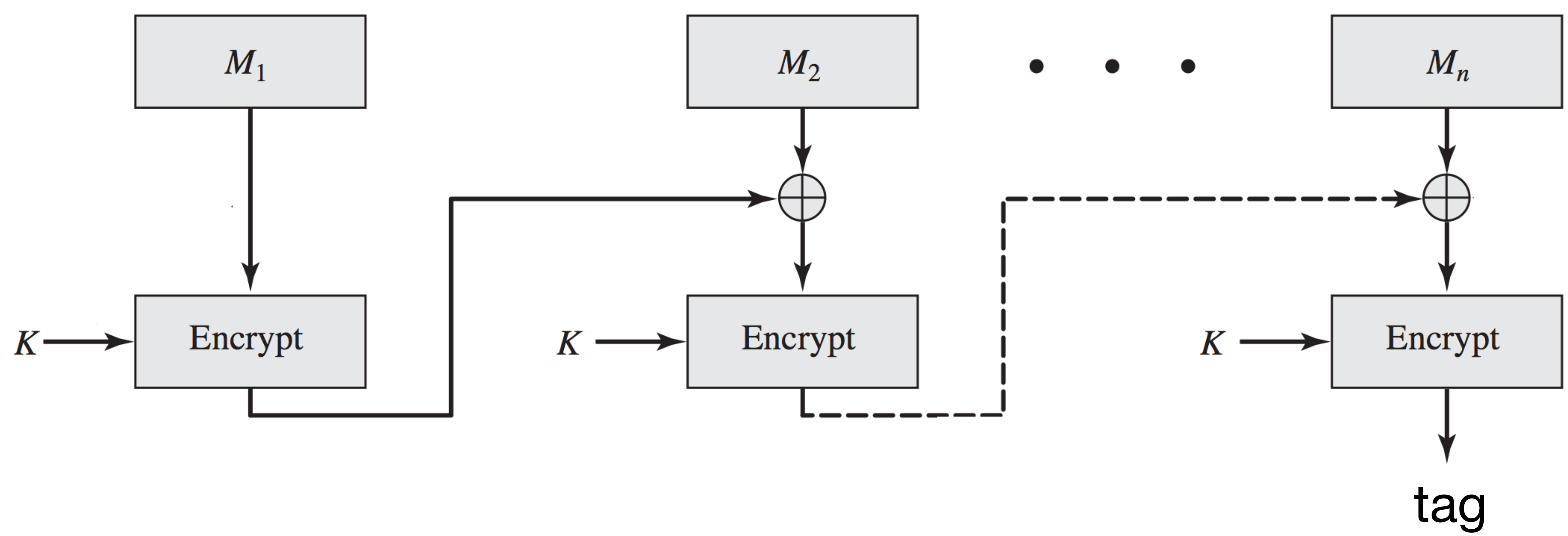
CBC-MAC

- Based on the Cipher Block Chaining (CBC) mode of operation



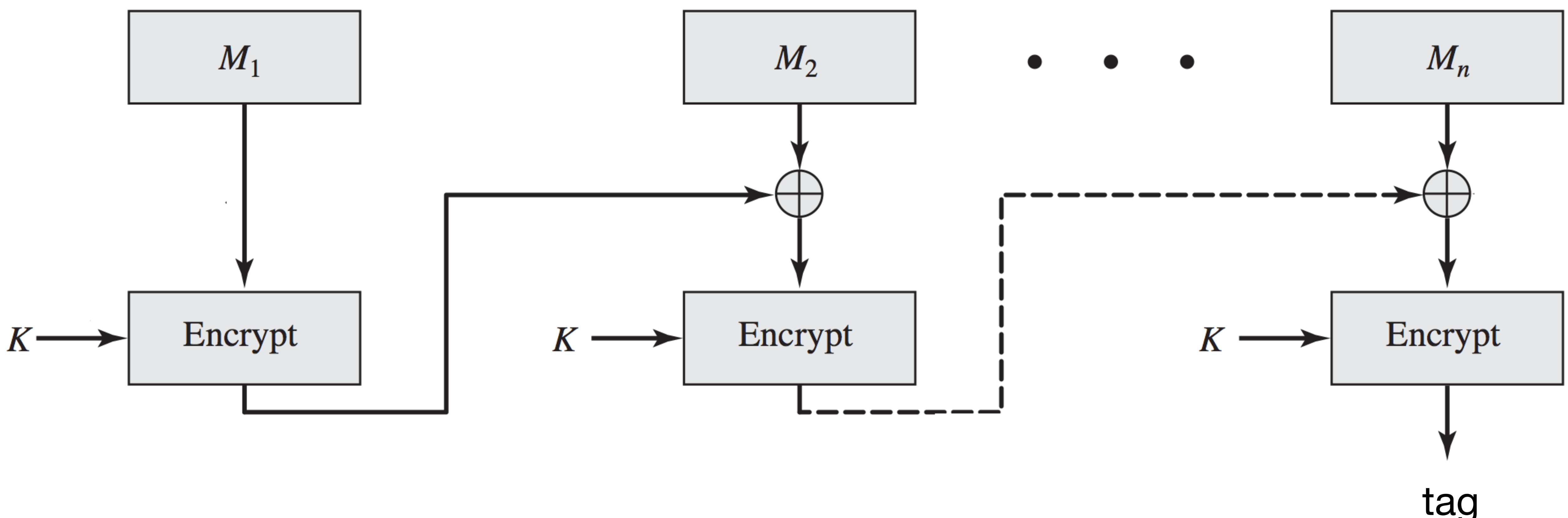
Reminder: Cipher-Block Chaining (CBC)





CBC-MAC

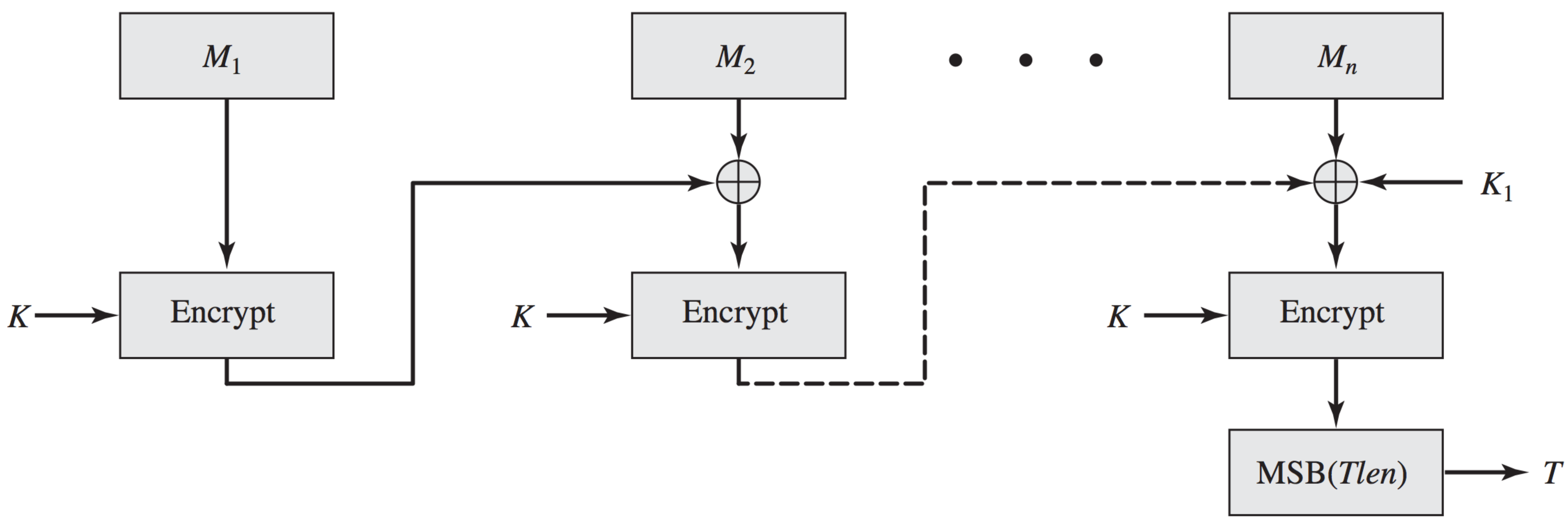
- Based on the Cipher Block Chaining (CBC) mode of operation
- Must use different keys for CBC encryption and CBC-MAC auth.



- **Not secure** for variable-length messages
  - given a one-block message **X** and its tag **T = MAC(K, X)**, attacker can create message **X | (X $\oplus$ T)**, whose tag is also **T**

Cipher-based MAC (CMAC)

- Standardized in 2005 by NIST
- Thwarts forgery for variable-length messages
- Second key K_1 is derived from E(K, 0)



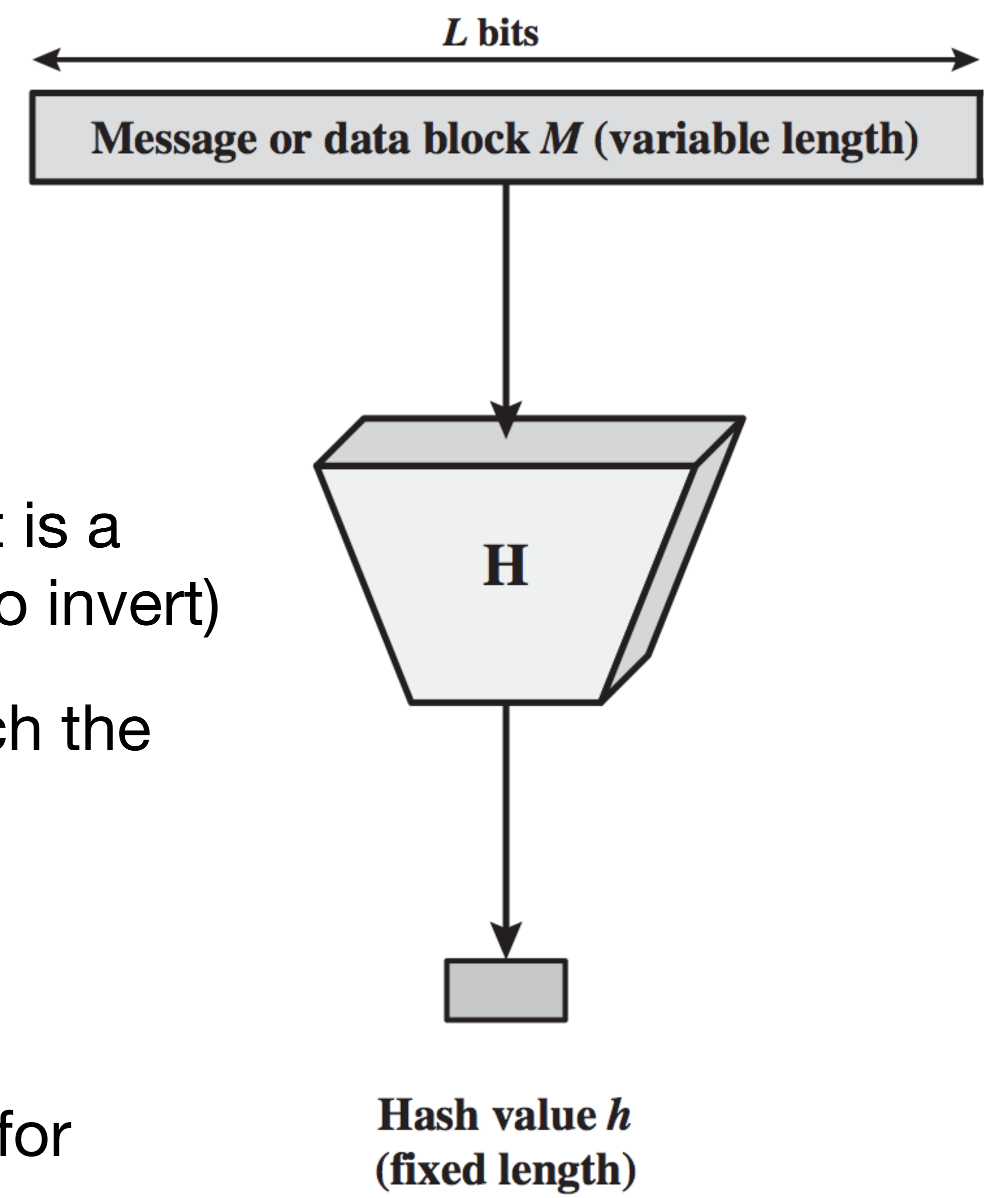
use as many bits $Tlen$ for the tag as you want

MAC Based on Hash Functions

Reminder: Cryptographic Hash Functions

- Hash function H:
  deterministically maps an input M
  to a fixed-length hash value H(M)

- Requirements
  - **one-way**: finding an input for which the output is a given hash value is hard (i.e., function is hard to invert)
  - **collision-resistant**: finding two inputs for which the hash values are the same is hard
  - **efficient**: computing the hash value of a given input is easy
  - **pseudorandom**: output meets standard tests for pseudo-randomness

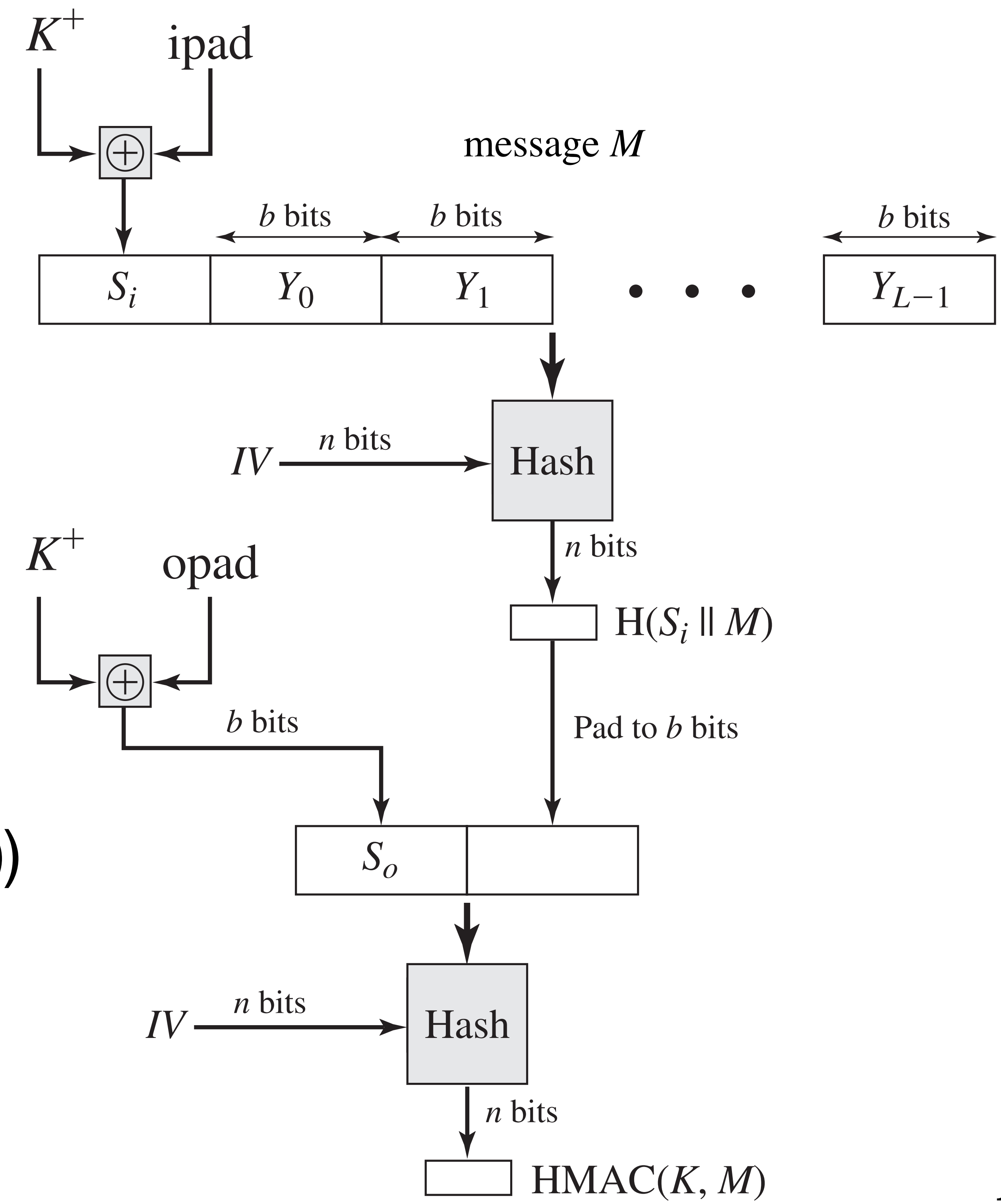


HMAC

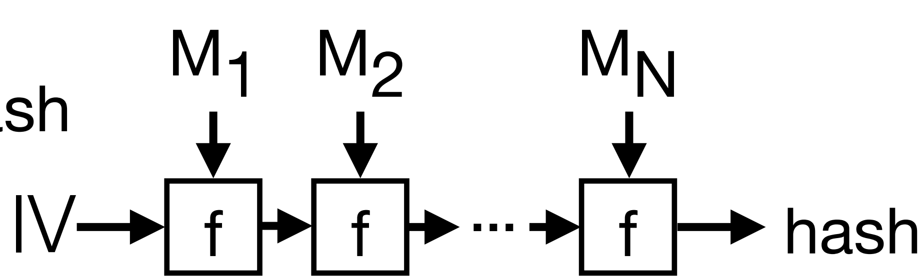
- Published in 1996 by Mihir Bellare, Ran Canetti, and Hugo Krawczyk
- **Provably secure** if the hash function is pseudorandom (collision-resistance is not necessary)
- Works with any hash function
  - but it is more efficient with iterative hash functions
- Widely used
  - part of the IPSec and SSL/TLS protocols
  - standardized in NIST FIPS PUB 198 and RFC 2104
- Formula:
  $HMAC(K, M) = H\big(K_{outer} \mid H(K_{inner} \mid M)\big)$

HMAC Structure

- **b**: block size of the hash function
- **IV**: initial value of the hash function
- Inputs
  - **Y_0, …, Y_{L-1}**: message blocks
  - **K^+**: key (padded with zeros)
  - **ipad** = `00110110` repeated
  - **opad** = `01011100` repeated
- Output HMAC(K, M) =
  $H(K^+ \oplus \text{opad} \mid H(K^+ \oplus \text{ipad} \mid M))$

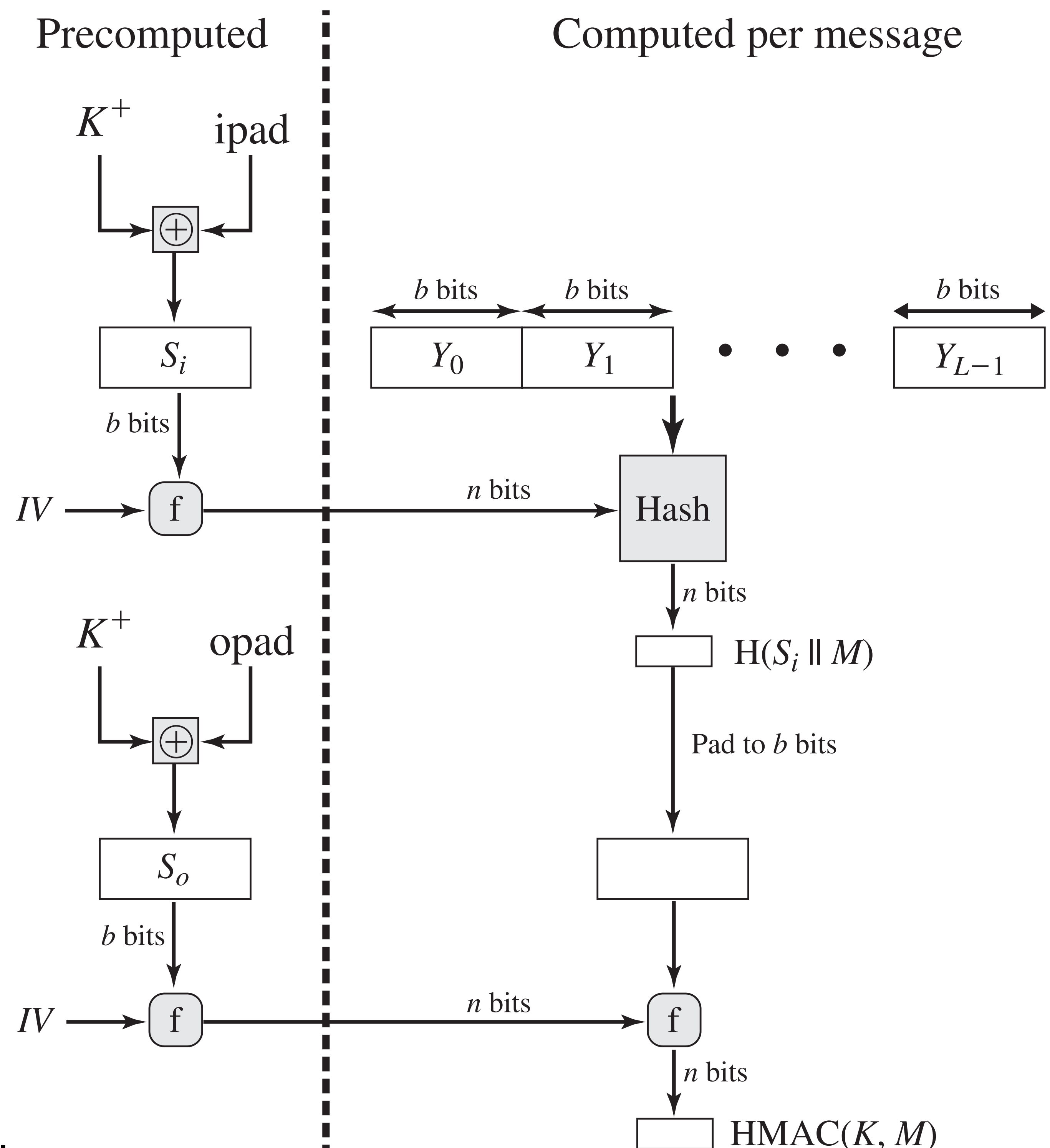


Iterative hash function:

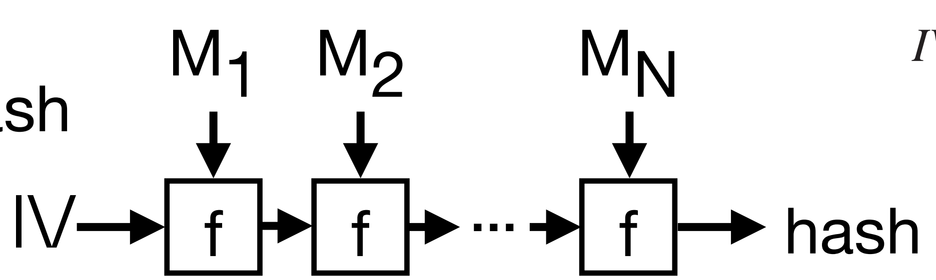


HMAC Precomputation

- Precompute
  - f(IV, $K^+ \oplus$ ipad)
  - f(IV, $K^+ \oplus$ opad)
- For L input blocks, we need only L + 1 compressions



Iterative hash function:



Authenticated Encryption

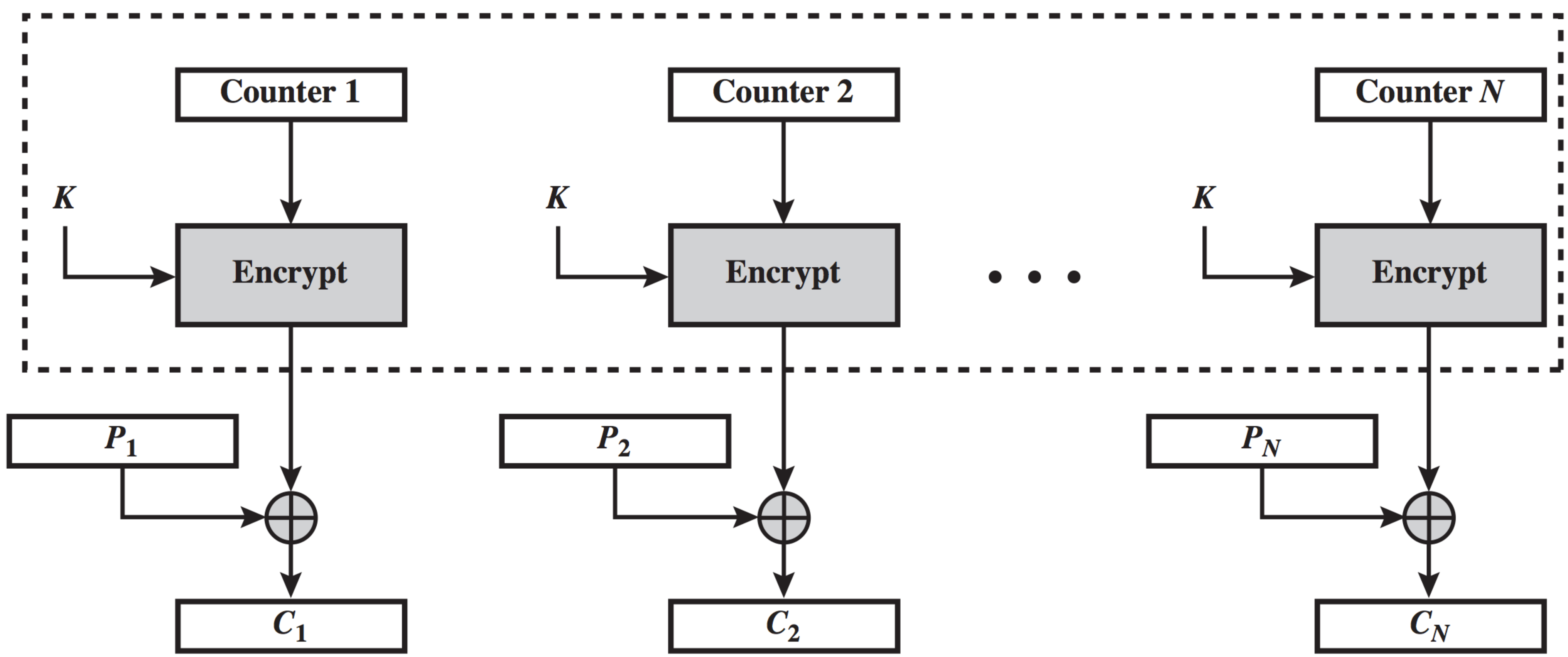
Authenticated Encryption

- Motivation
  - widely-used cryptographic primitives are (almost always) secure
  - secure encryption + secure authentication ↮ secure combination
  - some security protocols have used cryptographic primitives in an insecure way (*e.g.*, WEP)
- Authenticated encryption:
  encryption systems that provide both confidentiality and integrity
  ~ block cipher modes that provide confidentiality and integrity
- Approaches
  - **authentication followed by encryption** (*e.g.*, SSL/TLS)
  - **encryption followed by authentication** (*e.g.*, IPSec)
  - **independently encrypt and authenticate** (*e.g.*, SSH)

Counter with CBC-MAC (CCM)

- Standardized by NIST in NIST SP 800-38C
  - standard was developed in 2004 for WPA2
  - defined for the AES block cipher
- Encryption: based on the Counter (CTR) block-cipher mode

Reminder: Counter (CTR) Mode of Operation



Counter with CBC-MAC (CCM)

- Standardized by NIST in NIST SP 800-38C
  - standard was developed in 2004 for WPA2
  - defined for the AES block cipher
- Encryption: based on the Counter (CTR) block-cipher mode
  - converts a block cipher into a stream cipher
  - very efficient, blocks can be encrypted / decrypted in parallel
- Authentication: based on CBC-MAC message authentication
- Combination: first authenticate, then encrypt
  - compute CBC-MAC of the message, a nonce, and associated data (*e.g.*, protocol headers) that may need integrity protection
  - encrypt message and authentication tag in CTR mode

Galois/Counter Mode (GCM)

- Standardized in 2007 by NIST
- Encryption: based on the Counter (CTR) block-cipher mode
- Authentication: $\mathrm{GHASH}_H(X)$
  - inputs: hash key H, message blocks $X_1, \ldots, X_m$ (in 128-bit blocks)
  - output: $(X_1 \cdot H^m) \oplus (X_2 \cdot H^{m-1}) \oplus \cdots \oplus (X_{m-1} \cdot H^2) \oplus (X_m \cdot H)$
    where · is a special multiplication for 128 bit numbers
    - $H^m, H^{m-1}, \ldots, H^2$ can be precomputed, · can be performed in parallel
- Combination: first encrypt, then authenticate
  - authentication includes message length and associated data
- Overall, GCM mode is very efficient and parallelizable
  - widely used (*e.g.*, in HTTPS)

Message Authentication Conclusion

- Message authentication code (MAC)
  - ensures authenticity and integrity
  - requires a secret key
- Based on block-cipher: CMAC
- Based on hash-function: HMAC
- Authenticated encryption: CCM and GCM

Digital Signatures

Quick Overview

Motivation for Digital Signatures

- Message authentication does not protect the sender and receiver from each other
  - receiver can forge a message and claim that it is from the sender
  - sender can deny sending a message and claim that it was forged by the receiver
- <u>Non-repudiation:</u>
  sender cannot deny that it has sent a message
- Digital signature
  ≈ message authentication + non-repudiation
  - provide integrity and authenticity protection as well as non-repudiation
  - similar to traditional signatures: signee cannot deny signing a document
  - in many countries, digital signatures have legal significance

Next lecture:

Digital Signatures,
Key Exchange and Agreement