Course: Modern Cryptography

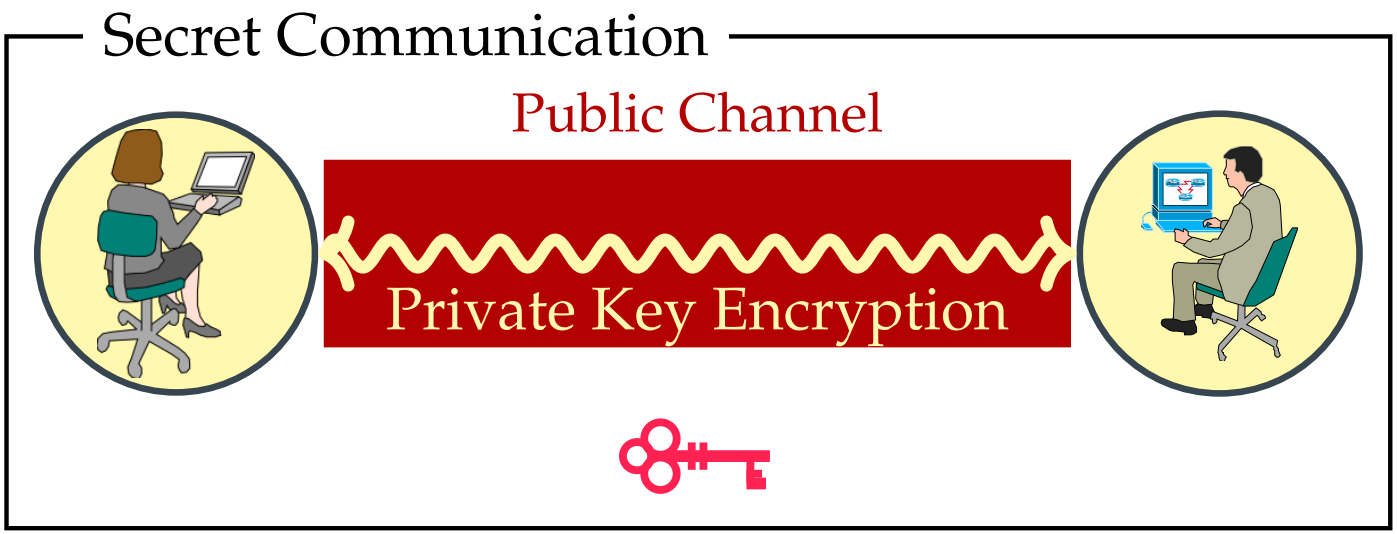
Message Authentication Codes and Authenticated Encryptions

Shashank Singh

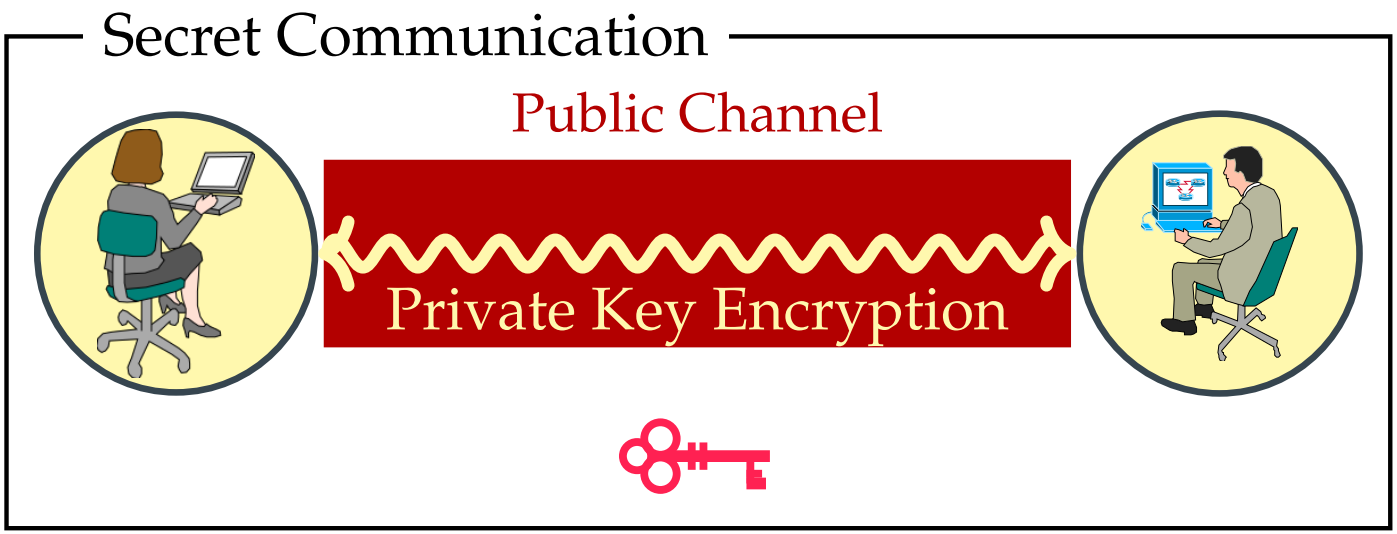
IISER Bhopal

October 16, 2025

Message Authentication Codes



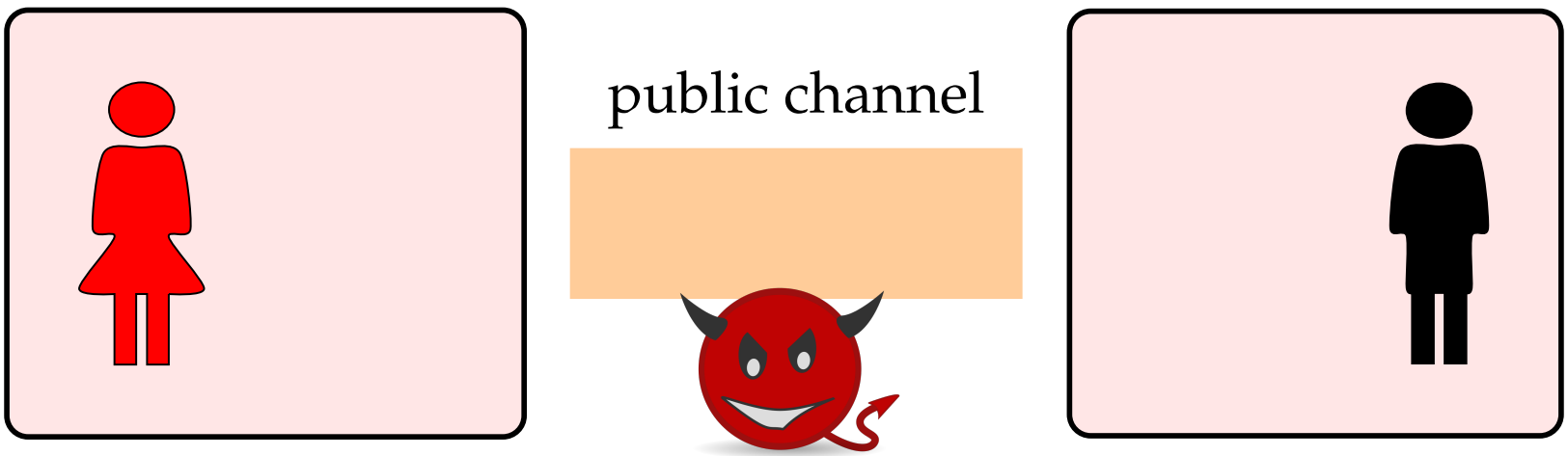
Message Authentication Codes



- While secrecy is important, ensuring message integrity and authentication can be of equal or greater importance!
- It's fascinating to note that encryption alone doesn't fully tackle this issue, highlighting the need for a broader approach to security.

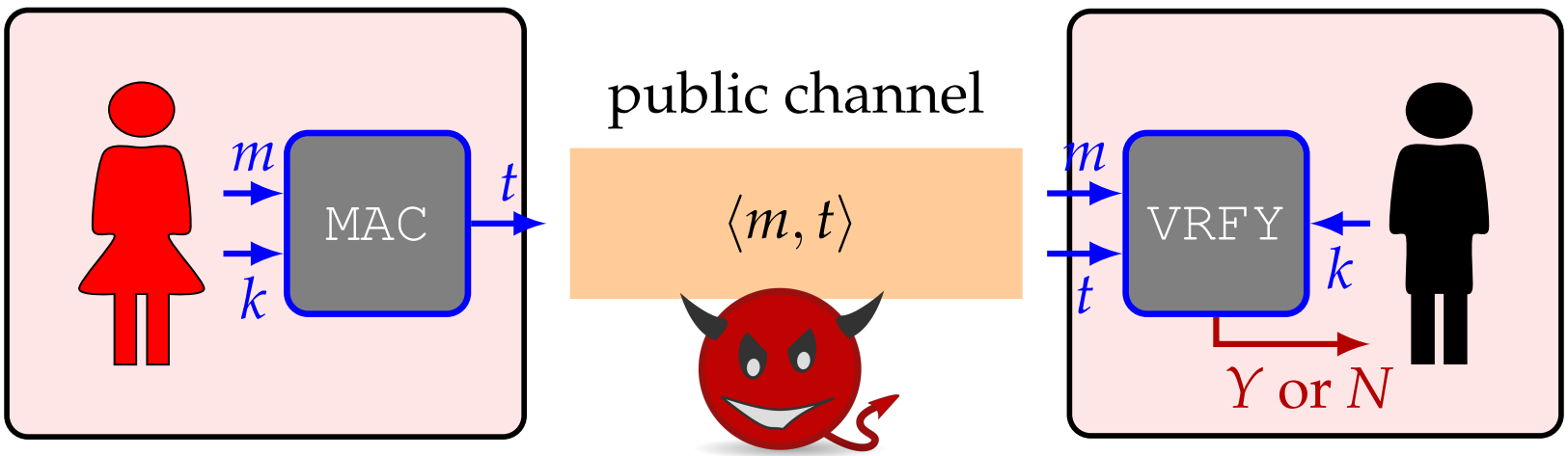
Setting of Message Authentication

For this section, assume that we are only concerned about message integrity and authentication.



Setting of Message Authentication

For this section, assume that we are only concerned about message integrity and authentication.



Message Authentication Code

A message authentication code (or MAC) consists of three probabilistic polynomial-time algorithms $(\texttt{GEN}, \texttt{MAC}, \texttt{VRFY})$ and a security parameter n, such that:

- $k \leftarrow \texttt{GEN}(n)$, with $|k| \geq n$.
- The tag-generation algorithm $\texttt{MAC}$ generates a tag $t \leftarrow \texttt{MAC}(k, m)$ for the given message m and key k.
- The deterministic tag verification algorithm $\texttt{VRFY}$, on input a key k, a message m, and a tag t returns a bit b.

It is required that for every message m, every key $k \leftarrow \texttt{GEN}(n)$, and every $m \in \{0,1\}^*$, it holds that $\texttt{VRFY}(k, m, \texttt{MAC}(k, m)) = 1$.

Message Authentication Code..

- As the keygen $\texttt{GEN}(n)$ almost always return k from uniform distribution we can omit this in the definition of MAC.
- If the MAC scheme is only defined for messages $m \in \{0,1\}^{\ell(n)}$, it is called a fixed length MAC.
- Canonical verification: Often in deterministic MACs, the verification is simply achieved using $\texttt{MAC}$ algorithm as follow:

$$\texttt{MAC}_k(m) \stackrel{?}{=} t$$

Thus the MAC schemes can be defined by the one algorithm i.e. $\texttt{MAC}()$.

MAC Security

Informally, we say that a MAC is secure if it is practically infeasible to come up with a message (may be gibberish) and its corresponding valid tag without knowing the key, even after seeing many valid message-tag pairs generated using the same key. Such a MAC is termed as:

existentially unforgeable under an adaptive chosen-message attack.

MAC Security

Consider the following experiment for a MAC $\Pi = (\mathtt{GEN}, \mathtt{MAC}, \mathtt{VRFY})$, an adversary $\mathcal{A}$ and a security parameter n.

The message authentication experiment $\mathtt{MacForge}_{\mathcal{A},\Pi}(n)$:

- $k \leftarrow \mathtt{GEN}(n)$
- The adversary $\mathcal{A}$ is given input n and oracle access to $\mathtt{MAC}_k(\cdot)$.
- The adversary eventually outputs (m^*, t^*). Let Q denote the set of all queries that $\mathcal{A}$ made to $\mathtt{MAC}_k(\cdot)$.
- $\mathcal{A}$ succeeds if and only if $m^* \notin Q$ and $\mathtt{VRFY}_k(m^*, t^*) = 1$. In that case the output of $\mathtt{MacForge}_{\mathcal{A},\Pi}(n)$ is defined to 1 and 0 otherwise.

Secure MAC

Definition

A MAC $\Pi = (\texttt{GEN}, \texttt{MAC}, \texttt{VRFY})$ is existentially unforgeable under an adaptive chosen-message attack, or just secure, if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function ε such that:

$$\Pr\left[\texttt{MacForge}_{\mathcal{A},\Pi}(n) = 1\right] \leq \varepsilon(n)$$

- What about MAC security against replay attacks?

Strongly Secure MAC

The experiment $\texttt{MacsForge}_{\mathcal{A},\Pi}(n)$:

- $k \leftarrow \texttt{GEN}(n)$
- The adversary $\mathcal{A}$ is given input n and oracle access to $\texttt{MAC}_k(\cdot)$.
- The adversary eventually outputs (m^*, t^*). Let Q denote the set of pairs (m, t), where t is the outcome of oracle query $\texttt{MAC}_k(m)$, by the adversary.
- $\mathcal{A}$ succeeds if and only if $\texttt{VRFY}_k(m^*, t^*) = 1$ and $(m^*, t^*) \notin Q$. In that case the output of $\texttt{MacForge}_{\mathcal{A},\Pi}(n)$ is defined to 1 and 0 otherwise.

Strong MAC

Definition

A MAC $\Pi = (\texttt{GEN}, \texttt{MAC}, \texttt{VRFY})$ is strongly secure, or a strong MAC, if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function ε such that:

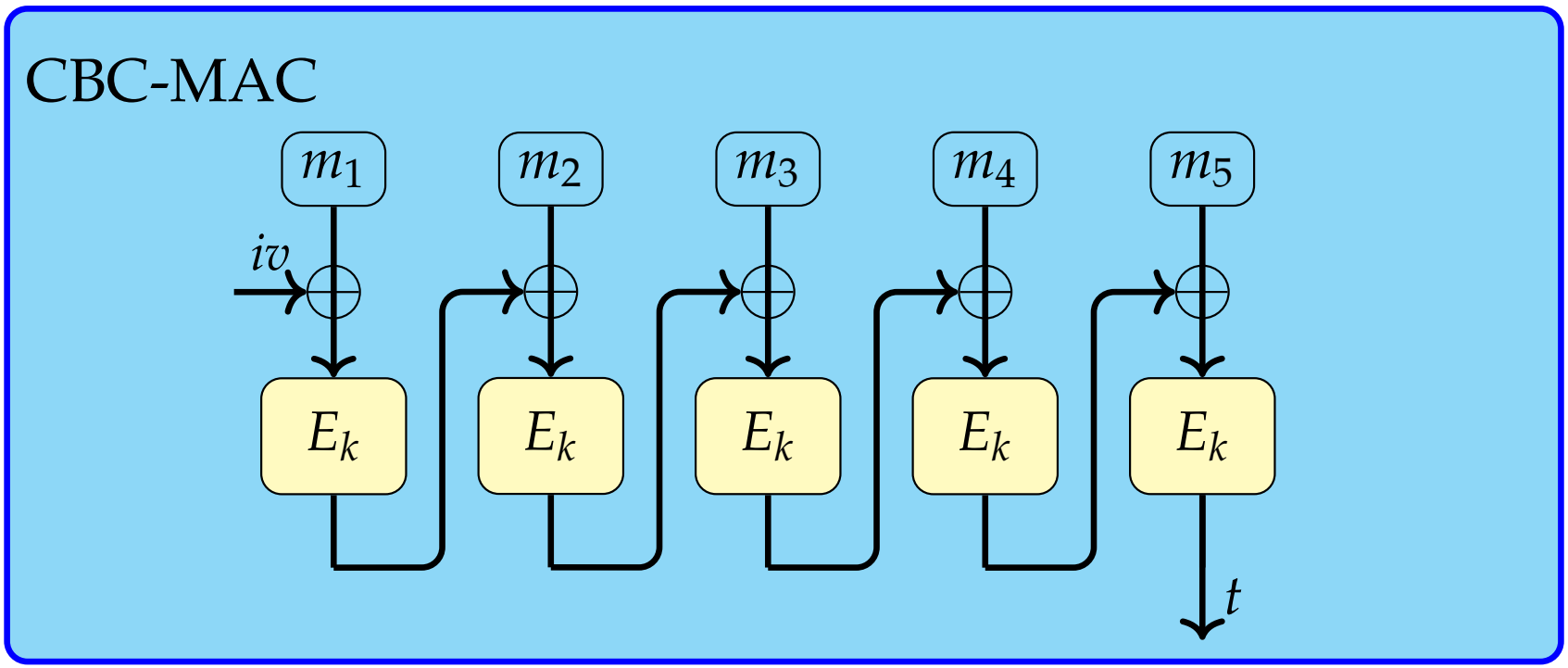
$$\Pr\left[\texttt{MacsForge}_{\mathcal{A},\Pi}(n) = 1\right] \leq \varepsilon(n)$$

Proposition

Let $\Pi = (\texttt{GEN}, \texttt{MAC}, \texttt{VRFY})$ be a secure MAC that uses canonical verification. Then Π is a strong MAC.

Construction of Secure MACs

- We will again use Pseudorandom Functions as a building blocks and also use some kind of modes of operations.



CBC-MAC..

- iv is usually set to all zero i.e., CBC-MAC is deterministic.
- What if, we chose random iv? What will happen to the security?
- How is it different to CBC mode of encryption?
- For any fixed length ℓ, CBC-MAC is secure MAC for messages of length $\ell \cdot n$.
- How to construct a MAC for variable length messages?

Exercises:

1. Show that the CBC-MAC described above, when applied to the variable length messages, is not a secure MAC.
2. Suggest a quick fix to the above problem.
3. In the construction of above MAC, initialisation vector is set to 0. Comment on the security of the above MAC if a random initialisation vector is used.

Cryptographic Hash Function–an informal overview

- It is a function $H : \{0,1\}^* \to \{0,1\}^n$ where n is a fixed value.
- Given m, it should be easy to compute $H(m)$. (Efficient)
- The function H should have the following additional properties:

1. Preimage Resistant: Given $H(m)$, it is (computationally) hard to find message m.
2. Second-Preimage Resistant: Given m, it is (computationally) difficult to find m' such that $H(m) = H(m')$.
3. Collision Resistant: It is (computationally) hard to find m and m', such that $H(m) = H(m')$.

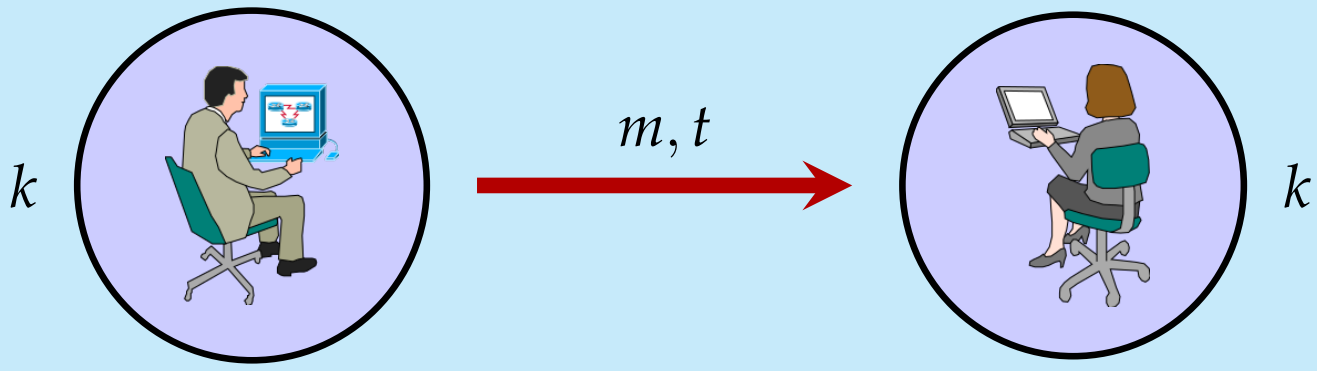
Cryptographic Hash Function–an informal overview

- It is a function $H : \{0,1\}^* \to \{0,1\}^n$ where n is a fixed value.
- Given m, it should be easy to compute $H(m)$. (Efficient)
- The function H should have t[illegible] properties:

  1. Preimage Resistant: Given [illegible] hard to find message m.
  2. Second-Preimage Resistant: [illegible] (computationally) difficult t[illegible] $H(m) = H(m')$.
  3. Collision Resistant: It is (co[illegible] and m', such that $H(m) = H$[illegible]

MAC based on Hash Function

Hash and MAC



$$t = \mathbf{Mac}_k(H(m))$$

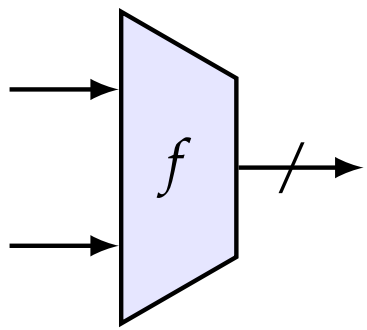
- If $\mathbf{Mac}_k$ is secure and H is collision-resistant, then it is a secure MAC.

HMAC

- This is a message authentication code, different form "Hash and MAC", which is also based on Cryptographic Hash Functions.
- HMAC is designed by Bellare, Canetti, and Krawczyk in 1996.
- Used for authentication in SSL, IPSEC etc.

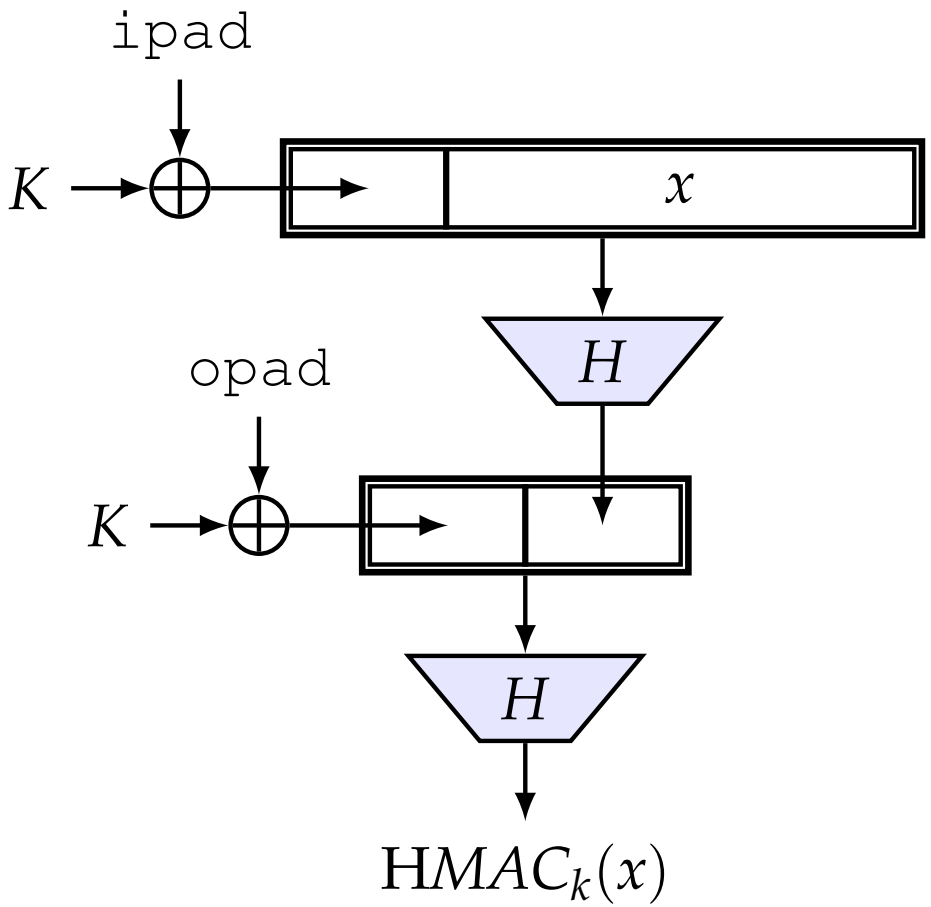
HMAC..

- Hash functions were not originally designed for message authentication. In particular, they are not keyed primitives.
- **Question:** How to use a hash function to construct a secure MAC?
  - Let H be an iterated b-bit hash function.
  - Let $f : \{0,1\}^{n+r} \mapsto \{0,1\}^n$ be its compression function. (For SHA-256, $n = 256$, $r = 512$.)
  - Let $k \xleftarrow{\$} \{0,1\}^n$ and K denotes k padded with $(r - n)$ 0's, so the bit-length of K is r.



HMAC..

$$\text{HMAC}(k, x) \;=\; H\left(K \oplus \texttt{opad}, H\left(K \oplus \texttt{ipad}, x\right)\right)$$



HMAC..

HMAC is commonly used as a key derivation function (KDF).

- Suppose that Alice has a secret key k, and wishes to derive several session keys sk_i, e.g. to encrypt data in different communication sessions.
- Alice computes

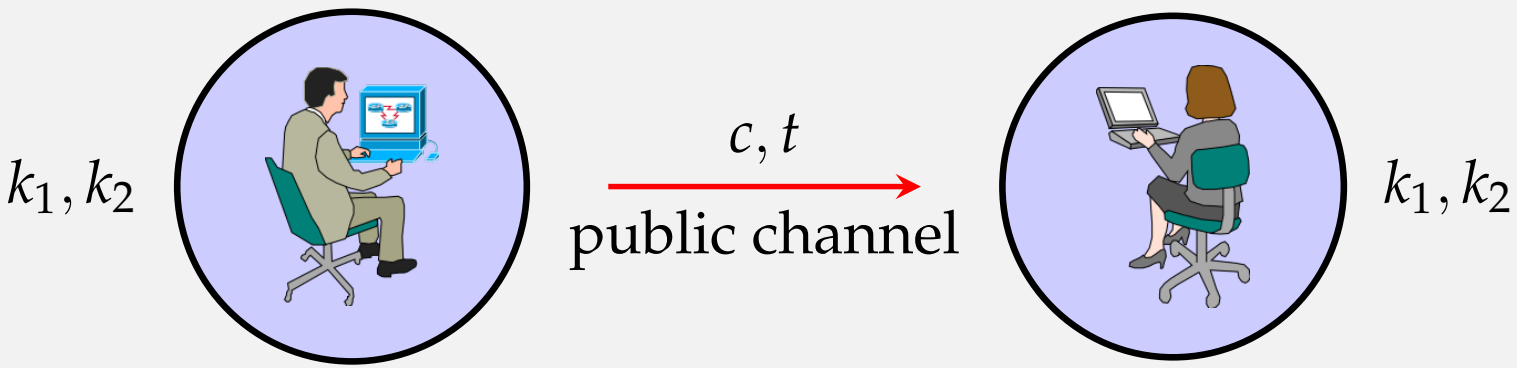
$$sk_1 = \text{HMAC}_k(1), sk_2 = \text{HMAC}_k(2), sk_3 = \text{HMAC}_k(3), \ldots$$

- **Rationale:** Without knowledge of an adversary is unable to learn anything about any particular session key even though she might have learnt some other session keys.

Authenticated Encryption

- Ideally, we require secrecy and message authentication both in an encryption scheme.

Encrypt and Authenticate



Encrypt and Authenticate

- Is there any problem with this scheme?
- Tag may leak some information about the message.(Think of Deterministic MAC?)
- There are dedicated (authenticated) encryption schemes which provide secrecy and authentication both.

Authenticated Encryption Security Notions

- An AE scheme is basically an encryption scheme, which provides message integrity in addition to the encryption.
- For encryption security of an AE $\Pi := (\texttt{GEN}, \texttt{ENC}, \texttt{DEC})$, we already have standard notions, e.g., the CPA, and the CCA security.
- As Π does not follow the syntax of the MAC, we can't directly use the notion of existential unforgeability under the adaptive chosen message attack for the integrity in this case. However, we have similar notion namely the unforgeable encryption in this case.

Authenticated Encryption Security Notions..

Let $\Pi = (\mathtt{GEN}, \mathtt{ENC}, \mathtt{DEC})$ be a private key encryption scheme. We design the following experiment for Π, an adversary $\mathcal{A}$ and value n for security parameter:

The unforgeable encryption EXP. [$\mathtt{EncForge}_{\mathcal{A},\Pi}(n)$]:

- $k \leftarrow \mathtt{GEN}(n)$
- The adversary $\mathcal{A}$ is given input n and oracle access to $\mathtt{ENC}_k(\cdot)$. The adversary outputs a ciphertext c.
- Let $m := \mathtt{DEC}_k(c)$ and Q be the set of all queries that $\mathcal{A}$ made to $\mathtt{ENC}_k(\cdot)$.The output of the experiment is 1 if and only if (1) $m \neq \perp$ and (2) $m \neq Q$.

AE Security..

Definition

A private-key encryption scheme $\Pi = (\texttt{GEN}, \texttt{ENC}, \texttt{VRFY})$ is unforgeable if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function ε such that:

$$\Pr\left[\texttt{EncForge}_{\mathcal{A},\Pi}(n) = 1\right] \leq \varepsilon(n)$$

Definition

A private-key encryption scheme is a (secure) authenticated encryption scheme if it is CCA-secure and unforgeable.

Authenticated Encryption schemes

Encrypt then Authenticate

k_1, k_2

c, t

public channel

k_1, k_2

$c = \mathbf{Enc}_{k_1}(m)$

$t = \mathbf{Mac}_{k_2}(c)$

$1 \stackrel{?}{=} \mathbf{Vrfy}_{k_2}(c, t)$

$m = \mathbf{Dec}_{k_1}(c)$

Construction 4.18: Encrypt-then-authenticate

Let $\Pi_E = (\mathtt{ENC}, \mathtt{DEC})$ be a private-key encryption scheme and let $\Pi_M = (\mathtt{MAC}, \mathtt{VRFY})$ be a message authentication code, where in each case key generation is done by simply choosing a uniform key $k \in \{0,1\}^n$. Define a private key encryption scheme $(\mathtt{GEN}', \mathtt{ENC}', \mathtt{DEC}')$ as follows:

- $\mathtt{GEN}'$: on input 1^n, choose independent, uniform $k_M, k_E \in \{0,1\}^n$ and return (k_E, k_M).
- $\mathtt{ENC}'$: on input a key (k_E, k_M) and a message m, compute $c \leftarrow \mathtt{ENC}_{k_E}(m)$ and $t \leftarrow \mathtt{MAC}_{k_M}(m)$. Output the cipher (c, t).
- $\mathtt{DEC}'$: on input a key (k_E, k_M) and a ciphertext (c, t), first compute $\mathtt{VRFY}_{k_M}(c, t) \stackrel{?}{=} 1$, if yes, then output $\mathtt{DEC}_{k_E}(m)$; if no, then output $\perp$.

Theorem

Let Π_E be a CPA-secure private-key encryption scheme, and let Π_M be a strongly secure message authentication code. Then Construction 4.18 is an authenticated encryption scheme.

Proof.

...

□

Theorem

Let Π_E be a CPA-secure private-key encryption scheme, and let Π_M be a strongly secure message authentication code. Then Construction 4.18 is an authenticated encryption scheme.

Proof.

...

□

Exercise

Comment on the security of Encrypt-then-Mac AE scheme when same key is used for both encryption and MAC.