# Message Authentication

We've talked about message secrecy until now.

# Message Authentication

We've talked about message secrecy until now.
How about message integrity?
Alice wants to know that Bob sent this message, and nobody else.

# Message Authentication

We've talked about message secrecy until now.
How about message integrity?
Alice wants to know that Bob sent this message, and nobody else.

This is has nothing to do with privacy or encryption!
Even if the message is sent in the clear, Alice might still like a proof that Bob sent it.

# Message Authentication

We've talked about message secrecy until now.
How about message integrity?
Alice wants to know that Bob sent this message, and nobody else.

This is has nothing to do with privacy or encryption!
Even if the message is sent in the clear, Alice might still like a proof that Bob sent it.

We still assume Alice and Bob share a key.
Intuitively, we want a way to "mark" or "tag" a message, using the secret key.
The recipient of the message should be able to verify that the tag was generated using the shared key: it could only have come from Bob.

# Message Authentication

## Message Authentication Codes

A MAC consists of three p.p.t. algorithms, $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$.

$\mathsf{Gen}(1^n)$: outputs a key $k$, $|k| > n$.

$\mathsf{Mac}(k, m)$: outputs a tag, $t$. This might be randomized, so we write it as $t \leftarrow \mathsf{Mac}(k, m)$. Also, $m$ might be variable length, or we might only support some fixed length for each $n$: $m \in \{0,1\}^{\ell(n)}$

$\mathsf{Vrfy}(k, m, t)$: is a deterministic algorithm that outputs a bit $b$. $b = 1$ means $t$ is a valid tag for $m$ using $k$, and $b = 0$ means $t$ is invalid and $m$ should be rejected.

# Message Authentication

## Message Authentication Codes

A MAC consists of three p.p.t. algorithms, $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$.

$\mathsf{Gen}(1^n)$: outputs a key $k$, $|k| > n$.

$\mathsf{Mac}(k, m)$: outputs a tag, $t$. This might be randomized, so we write it as $t \leftarrow \mathsf{Mac}(k, m)$. Also, $m$ might be variable length, or we might only support some fixed length for each $n$: $m \in \{0,1\}^{\ell(n)}$

$\mathsf{Vrfy}(k, m, t)$: is a deterministic algorithm that outputs a bit $b$. $b = 1$ means $t$ is a valid tag for $m$ using $k$, and $b = 0$ means $t$ is invalid and $m$ should be rejected.

Just as with encryption, we have a correctness requirement:
For every $n$, every $k$ output by Gen and every $m$ (of appropriate size),
$\mathsf{Vrfy}(k, m, \mathsf{Mac}(k, m)) = 1$.

# Message Authentication

## Message Authentication Codes

A MAC consists of three p.p.t. algorithms, $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$.

$\mathsf{Gen}(1^n)$: outputs a key $k$, $|k| > n$.

$\mathsf{Mac}(k, m)$: outputs a tag, $t$. This might be randomized, so we write it as $t \leftarrow \mathsf{Mac}(k, m)$. Also, $m$ might be variable length, or we might only support some fixed length for each $n$: $m \in \{0,1\}^{\ell(n)}$

$\mathsf{Vrfy}(k, m, t)$: is a deterministic algorithm that outputs a bit $b$. $b = 1$ means $t$ is a valid tag for $m$ using $k$, and $b = 0$ means $t$ is invalid and $m$ should be rejected.

Just as with encryption, we have a correctness requirement:
For every $n$, every $k$ output by Gen and every $m$ (of appropriate size),
$\mathsf{Vrfy}(k, m, \mathsf{Mac}(k, m)) = 1$.

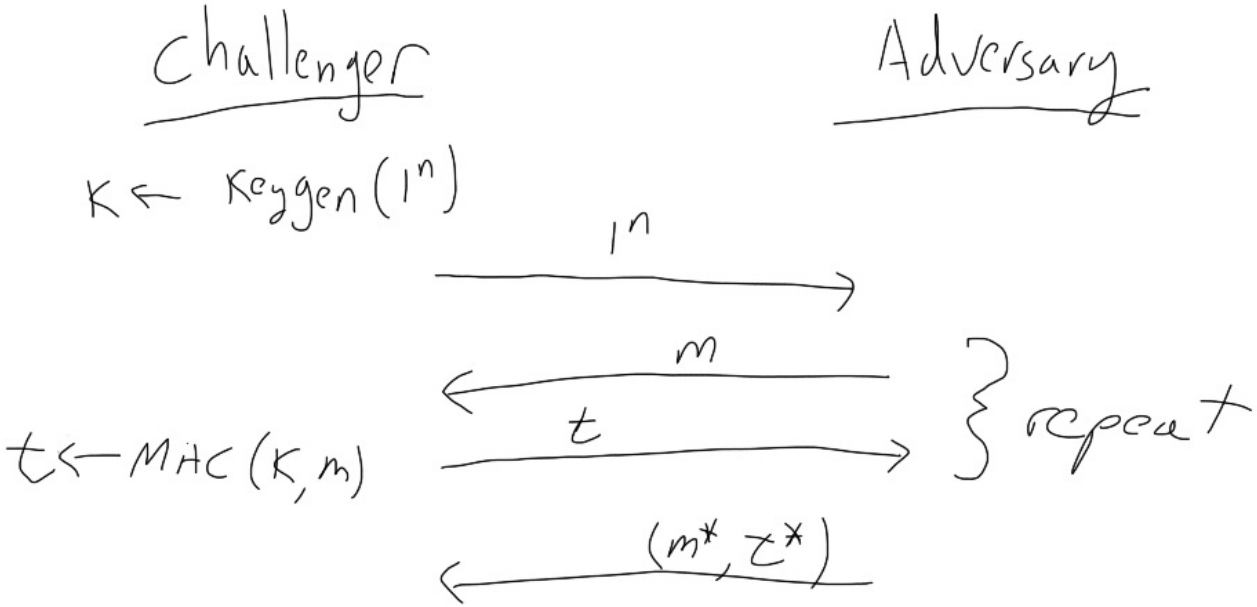
Canonical Verification: Unlike with encryption schemes, deterministic MAC algorithms *can* be secure. For such algorithms, we have a simple way of verifying.
$\mathsf{Vrfy}(k, m, t)$: Compute $t' = \mathsf{Mac}(k, m)$. If $t' = t$ output 1. Else, output 0.

# Unforgeability

We define a new security game, Mac-forge$_{\mathcal{A},\Pi}$, for capturing that a MAC scheme, $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$ is *unforgeable*. That is, without the key $k$, an adversary $\mathcal{A}$ cannot create a valid tag.

# Unforgeability

We define a new security game, Mac-forge$_{\mathcal{A},\Pi}$, for capturing that a MAC scheme, $\Pi = (\text{Gen}, \text{Mac}, \text{Vrfy})$ is *unforgeable*. That is, without the key $k$, an adversary $\mathcal{A}$ cannot create a valid tag.




Let $Q$ be the set of messages queried

$\mathcal{A}$ wins if 1) $\text{Vrfy}(k, m^*, t^*) = 1$

2) $m^* \notin Q$

# Unforgeability

## Definition

A MAC scheme $\Pi = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ is existentially unforgeable under an adaptive chosen-message attack, (or, *secure*) if for all p.p.t. adversaries $\mathcal{A}$, there is some negligible function negl such that:

$$\Pr[\text{Mac-forge}_{\mathcal{A},\Pi}(n) = 1] \leq \mathrm{negl}(n)$$

# A Construction

## A fixed length MAC, $\Pi$

Let $F$ be a PRF. We define a fixed length MAC for messages of length $n$:

$\text{Gen}(1^n) : k \leftarrow \{0,1\}^n$

$\text{Mac}(k, m)$: If $|m| \neq n$, abort. Otherwise, output $t = F_k(m)$.

$\text{Vrfy}(k, m, t)$: If $|m| \neq n$, output 0. Otherwise, compute $t' = F_k(m)$. If $t' = t$ output 1, otherwise, output 0.

# A Construction

## A fixed length MAC, $\Pi$

Let $F$ be a PRF. We define a fixed length MAC for messages of length $n$:

$\text{Gen}(1^n) : k \leftarrow \{0,1\}^n$

$\text{Mac}(k, m)$: If $|m| \neq n$, abort. Otherwise, output $t = F_k(m)$.

$\text{Vrfy}(k, m, t)$: If $|m| \neq n$, output 0. Otherwise, compute $t' = F_k(m)$. If $t' = t$ output 1, otherwise, output 0.

Claim: If $F$ is a PRF, then the construction above is a secure fixed-length MAC.

# A Construction

## A fixed length MAC, $\Pi$

Let $F$ be a PRF. We define a fixed length MAC for messages of length $n$:

$\text{Gen}(1^n) : k \leftarrow \{0,1\}^n$

$\text{Mac}(k, m)$: If $|m| \neq n$, abort. Otherwise, output $t = F_k(m)$.

$\text{Vrfy}(k, m, t)$: If $|m| \neq n$, output 0. Otherwise, compute $t' = F_k(m)$. If $t' = t$ output 1, otherwise, output 0.

Claim: If $F$ is a PRF, then the construction above is a secure fixed-length MAC.

Claim: If there exists $\mathcal{A}$ with non-negligible advantage in the Mac-forge$_{\mathcal{A},\Pi}$ game, there exists $\mathcal{A}_r$ that wins in the $\text{PrivK}^{\text{prf}}_{\mathcal{A},\text{F}}$ game with probability $\frac{1}{2} + \frac{1}{p(n)}$.