

Message Authentication Codes

Lecture 19: A secret tag for a public message

Lecture overview

Alice and Bob share a secret key. A short tag lets Bob check a message even when an attacker controls the channel. A random function gives a particularly clean construction and proof.

Correctness, unforgeability, and the cost of a random table.

Based on the instructor's Lecture 19 [Maj26].

Abstract and lecture roadmap

We have studied how to hide messages. Today the message may be public. Bob instead wants to know whether someone changed it or invented a new one.

1. Specify what a message authentication code does.
2. Describe exactly what counts as a successful forgery.
3. Authenticate using a secret random function.
4. Prove security, then examine the size of the secret key.

Proof roadmap

The attacker sees some rows of a random table. The proposed forgery must use a new row. That row is still uniform, so guessing its ℓ -bit value succeeds with probability $2^{-\ell}$.

[BS23, §§6.1, 6.3, pp. 217–225].

Main results

Let $\mathcal{M} = \{0,1\}^n$ and $\mathcal{T} = \{0,1\}^\ell$, where $n, \ell \geq 1$ are integers.

A random function gives an information-theoretic MAC

Choose a uniform function $f: \mathcal{M} \rightarrow \mathcal{T}$ as the secret key. Tag m with $f(m)$ and verify by recomputing it.

After any number of adaptive tagging queries, one attempt to forge a tag on an unqueried message succeeds with probability at most $2^{-\ell}$. The attacker may have unlimited computation.

The cost, and the connection to Lecture 18

The full random table contains $\ell 2^n$ secret bits. Replacing it with a secure PRF gives a short-key MAC against efficient attackers, with forgery probability at most

$$2^{-\ell} + \text{PRF distinguishing advantage.}$$

[BS23, §6.3, Thm. 6.2, pp. 224–225].

A public message can still need protection

Alice's instruction

Pay Bob 10 dollars

An attacker changes the instruction to Pay Bob 1000 dollars.

What Bob needs

A way to check that the received instruction was authorized by someone who knows the shared secret key.

- The attacker may read the message and its tag.
- The attacker may replace either one.
- Bob checks the received message and received tag together.

Message-integrity motivation: [BS23, §6.1, pp. 217–218].

Three algorithms

A MAC has a key space $\mathcal{K}$, a message space $\mathcal{M}$, and a tag space $\mathcal{T}$.

Key generation

$\text{Gen}()$ samples a secret key $K \in \mathcal{K}$, shared by Alice and Bob.

Tagging

$\text{Sign}_K(m)$ returns a tag $\tau \in \mathcal{T}$. Alice sends (m, τ) .

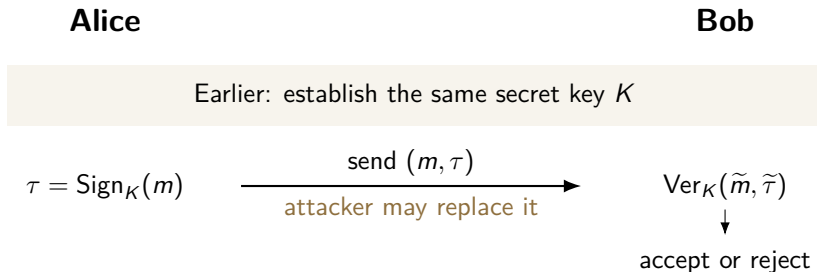
Verification

$\text{Ver}_K(\tilde{m}, \tilde{\tau}) \in \{0, 1\}$ checks the received pair. Output 1 means accept, and output 0 means reject.

We retain the source slides' name Sign for the tagging algorithm. This is a shared-key primitive.

[BS23, Def. 6.1, p. 217]; source slides, pp. 3–4 [Maj26].

Alice and Bob share the verification key



The channel carries the message and tag. The secret key stays with the two parties.

Source diagram: p. 4 [Maj26]; [BS23, Fig. 6.1, p. 216].

Correctness: honest tags are accepted

Perfect correctness

For every $m \in \mathcal{M}$,

$$\Pr[\text{Ver}_K(m, \text{Sign}_K(m)) = 1] = 1.$$

The probability includes key generation and any randomness used by the algorithms.

Correctness applies to every message, including messages chosen after the key is fixed, when the condition holds for every generated key.

Our deterministic constructions

For every possible key K , verification accepts exactly the tag $\text{Sign}_K(m)$. Thus correctness holds separately for every key and every message.

[BS23, Def. 6.1 and deterministic MACs, p. 217].

What a MAC guarantees

Question	What the MAC provides
Can the attacker read m ?	Yes. The message is sent in the clear.
Can Bob generate tags?	Yes. Bob also knows K .
Is a valid tag public proof that Alice sent m ?	No. Either key holder can generate it. Digital signatures use a different key arrangement.
Can an old valid pair be replayed?	The tag still verifies. Replay protection needs protocol state, such as a counter included in the authenticated message.

The security goal concerns an outsider who does not know the key. Revealing the key lets that outsider generate tags too.

[BS23, §6.1, pp. 217–219]; signatures: [HPS14, §4.1, pp. 193–195].

Two ways to modify a message–tag pair

Suppose the attacker sees (m, τ) .

Same-tag substitution

The attacker sends (m', τ) with $m' \neq m$.
The message changes, but the original tag is kept.

General forgery

The attacker sends (m', τ') with $m' \neq m$.
The tag τ' is arbitrary. It may equal τ .

Our security definition allows the general attack, so it also covers same-tag substitution.

In the source slides, the first attack motivates “message integrity.” Standard MAC unforgeability controls both attacks.

Source slides, pp. 7–8 [Maj26]; [BS23, §6.1, pp. 218–219].

One observed tag: the security experiment

1. The challenger samples $K \xleftarrow{\$} \text{Gen}()$ and keeps it secret.
2. The attacker chooses $m \in \mathcal{M}$ and receives $\tau = \text{Sign}_K(m)$.
3. Using this information, the attacker outputs (m^*, τ^*) .

The attacker wins exactly when

$$m^* \neq m \quad \text{and} \quad \text{Ver}_K(m^*, \tau^*) = 1.$$

A valid tag on a new message is a forgery. Repeating the observed pair does not win this experiment.

Security bounds the probability of this event over the key and the attacker's randomness.

Chosen-message viewpoint: [BS23, Attack Game 6.1 and Def. 6.2, pp. 218–219].

The conditional probability we would like

For a deterministic, unique-tag scheme, a particularly strong guarantee is

Ideal one-observation guarantee

For all $m' \neq m$ and all $\tau, \tau' \in \mathcal{T}$, whenever the conditioning event has positive probability,

$$\Pr_K[\text{Sign}_K(m') = \tau' \mid \text{Sign}_K(m) = \tau] = \frac{1}{|\mathcal{T}|}.$$

Even after seeing one tag, every possible tag on a different message is equally likely.

This is an exact property of our random-function construction. A general MAC definition asks for a small forgery probability, rather than this equality.

Source slides, p. 8 [Maj26]; ideal-world analysis: [BS23, §6.3, pp. 224–225].

Why guessing a tag can never be ruled out

Fix any fresh message m^* . Suppose verification accepts exactly one tag in $\mathcal{T} = \{0, 1\}^\ell$ for each key.

A universal guessing attack

Choose $\tau^* \xleftarrow{\$} \mathcal{T}$, independently of the key and transcript.

$$\Pr[\text{Ver}_K(m^*, \tau^*) = 1 \mid K] = \frac{1}{|\mathcal{T}|} = 2^{-\ell}.$$

Averaging over K gives the same probability.

Thus $2^{-\ell}$ is the best possible bound for one fresh-message attempt in this setting.

This argument lets the attacker choose a *new random tag*. It does not claim the same bound for every attack that reuses an old tag.

Unique-tag convention: [BS23, Def. 6.1, p. 217].

Uniform tags alone do not imply security

Take $\mathcal{M} = \mathcal{T} = \{0, 1\}^n$ and choose a uniform secret $K \in \{0, 1\}^n$. Consider

$$\text{Sign}_K(m) = K \oplus m.$$

For each fixed message, its tag is uniformly distributed.

One observation is enough to forge

From $\tau = K \oplus m$, the attacker learns $K = \tau \oplus m$. For any $m^* \neq m$, it computes

$$\tau^* = \tau \oplus m \oplus m^* = K \oplus m^*.$$

Verification accepts with probability 1.

We need unpredictability after observations. A uniform distribution before any observation is not enough.

Security framework: [BS23, §6.1, pp. 218–219].

Many observations, including adaptive choices

The challenger samples one key K . The attacker makes up to t queries:

$$m_1 \mapsto \tau_1, \quad m_2 \mapsto \tau_2, \quad \dots, \quad m_q \mapsto \tau_q, \quad q \leq t.$$

Each m_i may depend on earlier replies, and $\tau_i = \text{Sign}_K(m_i)$.

The final forgery

Let $\mathcal{Q} = \{m_1, \dots, m_q\}$ be the set of queried messages. The attacker outputs (m^*, τ^*) and wins if

$$m^* \notin \mathcal{Q} \quad \text{and} \quad \text{Ver}_K(m^*, \tau^*) = 1.$$

We count *one final forgery attempt*. Repeated tagging queries are allowed. If every message has been queried, no fresh-message forgery is possible.

[BS23, Attack Game 6.1, pp. 218–219]; the unique-tag convention makes the games equivalent.

A random function is a table of independent values

For a small example, take $\mathcal{M} = \{0, 1\}^2$ and $\mathcal{T} = \{0, 1\}^3$.

Message	Secret table entry
00	$f(00) \xleftarrow{\$} \{0, 1\}^3$
01	$f(01) \xleftarrow{\$} \{0, 1\}^3$
10	$f(10) \xleftarrow{\$} \{0, 1\}^3$
11	$f(11) \xleftarrow{\$} \{0, 1\}^3$

What “uniform random function” means

Choose each row independently and uniformly.
Keep the chosen value fixed whenever that input appears again.

Different rows may contain the same tag. A random function need not be injective.

Seeing $f(00)$ and $f(10)$ leaves $f(01)$ and $f(11)$ independent and uniform.

[BS23, §4.4.2, pp. 133–134].

The random-function construction

Fix $\mathcal{M} = \{0, 1\}^n$ and $\mathcal{T} = \{0, 1\}^\ell$.

Key generation

Choose a uniform random function $f: \mathcal{M} \rightarrow \mathcal{T}$. Give its entire table to Alice and Bob as their shared secret key.

Tag a message

$$\text{Sign}_f(m) = f(m).$$

Send $(m, f(m))$.

Verify a received pair

$$\text{Ver}_f(\tilde{m}, \tilde{\tau}) = \begin{cases} 1 & \tilde{\tau} = f(\tilde{m}), \\ 0 & \text{otherwise.} \end{cases}$$

The key is sampled once. All messages under this key use the same function.

Source slides, pp. 11–12 [Maj26]; ideal construction in [BS23, §6.3, p. 224].

Correctness and the first security calculation

Correctness for every table

For every f and m , Alice sends $\tau = f(m)$. Bob checks the same equality, so

$$\text{Ver}_f(m, \text{Sign}_f(m)) = \text{Ver}_f(m, f(m)) = 1.$$

Now suppose the attacker learns that $f(m) = \tau$. Fix $m' \neq m$ and any proposed tag τ' .

One observed row leaves the next row uniform

$$\Pr_f[f(m') = \tau' \mid f(m) = \tau] = 2^{-\ell}.$$

The two entries were sampled independently. This also holds when $\tau' = \tau$.

[BS23, §4.4.2, pp. 133–134; §6.3, pp. 224–225].

The same probability, by counting tables

Write $N = |\mathcal{M}| = 2^n$ and $R = |\mathcal{T}| = 2^\ell$.

Restrictions on the table	Number of tables
None	R^N
One specified entry: $f(m) = \tau$	R^{N-1}
Two specified entries: $f(m) = \tau, f(m') = \tau'$	R^{N-2}
Every table was equally likely. For distinct m, m' ,	

Conditional probability

$$\Pr[f(m') = \tau' \mid f(m) = \tau] = \frac{R^{N-2}}{R^{N-1}} = \frac{1}{R} = 2^{-\ell}.$$

Counting/probability background: [HPS14, §5.1, pp. 208–213; §5.3.1, pp. 228–232].

Several observed rows

Suppose the attacker has seen $f(m_i) = \tau_i$ for s distinct messages $m_1, \dots, m_s$.

A fresh row still has all its possibilities

For any $m^* \notin \{m_1, \dots, m_s\}$ and any $\tau^* \in \mathcal{T}$,

$$\Pr[f(m^*) = \tau^* \mid f(m_1) = \tau_1, \dots, f(m_s) = \tau_s] = \frac{R^{N-s-1}}{R^{N-s}} = 2^{-\ell}.$$

The observed entries fix s rows of the table. Each other row is still independent and uniform.

What if the attacker chooses its queries and proposed forgery based on the tags it sees?

[BS23, §6.3, proof of Thm. 6.2, pp. 224–225].

Why adaptive queries do not change the argument

Expose the table only when a row is requested

On a new message, sample an independent uniform tag and remember it. On a repeated message, return the remembered tag.

This gives exactly the same replies as choosing the whole random table at the start.

1. Fix the attacker's random coins and a possible complete transcript.
2. The transcript fixes the queried rows. It also fixes the attacker's final choice (m^*, τ^*) .
3. If m^* is fresh, its table entry has not been exposed. It remains uniform, so the conditional success probability is $2^{-\ell}$.

Average over all transcripts and coins. Invalid or nonfresh outputs contribute zero, giving total success probability at most $2^{-\ell}$.

Lazy sampling: [BS23, §4.4.2, pp. 133–134]; application: [BS23, Thm. 6.2, pp. 224–225].

The exact security guarantee

Random-function MAC theorem

For every attacker, even one with unbounded computation, and every budget t of tagging queries,

$$\Pr[\text{one successful fresh-message forgery}] \leq 2^{-\ell}.$$

Equality holds whenever the attacker always outputs a fresh message and a valid-length tag.

- Adaptive choices and repeated tagging queries are allowed.
- The final tag may equal an earlier tag.
- If no fresh message remains, the success probability is zero.

There is no factor of t : observing t tags still leaves only *one final guess* in this experiment.

[BS23, §6.3, proof of Thm. 6.2, pp. 224–225].

Tag length determines the guessing probability

For a single forgery attempt, the random-function scheme has error $2^{-\ell}$.

Tag length ℓ	One-attempt probability	Interpretation
3	$1/8$	Useful for a classroom table
5	$1/32$	Too large for meaningful protection
128	2^{-128}	Smaller still

The source slides' choice

If $100 \mid n$, taking $\ell = n/100$ gives error $2^{-n/100}$ and table size $(n/100)2^n$ bits. For arbitrary $n \geq 100$, use $\ell = \lfloor n/100 \rfloor$.

Tag length must make guessing sufficiently unlikely for the intended use.

Source slides, pp. 11, 15 [Maj26]; random-function bound: [BS23, §6.3, pp. 224–225].

Tag collisions and forgery are different events

When $|\mathcal{M}| > |\mathcal{T}|$, every table has collisions. Two different messages can have the same tag.

Seeing a collision

If the attacker queries m_1, m_2 and finds $f(m_1) = f(m_2)$, both messages are already in its query set. Reusing either observed pair is not a fresh-message forgery.

Trying an old tag on a fresh message

For any chosen old tag τ_i and unqueried m^* , the attacker still has

$$\Pr[f(m^*) = \tau_i \mid \text{its transcript}] = 2^{-\ell}.$$

The key point is the unobserved entry at the forged message, even when some observed entries collide.

Freshness in [BS23, Attack Game 6.1, pp. 218–219]; proof in [BS23, §6.3, pp. 224–225].

The random table is an enormous secret key

There are 2^n messages and 2^ℓ possible tags for each message.

Counting all possible keys

$$\#\{f: \{0,1\}^n \rightarrow \{0,1\}^\ell\} = (2^\ell)^{2^n} = 2^{\ell 2^n}.$$

A full table stores one ℓ -bit tag for each of 2^n messages, so it contains $\ell 2^n$ bits.

For example

With $n = 512$ and $\ell = 128$, the key table has $128 \cdot 2^{512} = 2^{519}$ bits. Transmitting one tag is easy. Storing the entire key is infeasible.

The table cost is exponential in the message length, even if the tag length is modest.

Source slides, p. 22 [Maj26]; counting: [HPS14, §5.1, pp. 208–213].

What the random-function construction teaches us

Feature	Consequence
Independent secret table rows	A fresh message has an unpredictable tag.
No computational assumption	Even unlimited computation does not improve one fresh guess.
A fixed table for the key	Repeated messages receive the same tag.
$\ell 2^n$ secret table bits	The construction is an ideal benchmark, with a prohibitive storage cost.

The exponential key is a cost of *this full random table*. Short-key information-theoretic MACs exist for bounded use, for example one-time authentication.

[BS23, §6.3, pp. 224–225; §7.6, Thm. 7.12, pp. 275–276].

Preview: replace the table with a PRF

Lecture 18 gave us a way to compute apparently random table entries using a short secret key.

A MAC from a pseudorandom function

For a PRF family $F_K: \{0, 1\}^n \rightarrow \{0, 1\}^\ell$, set

$$\text{Sign}_K(m) = F_K(m), \quad \text{Ver}_K(m, \tau) = \mathbf{1}[\tau = F_K(m)].$$

The stored secret is the PRF key K .

Quantitative guarantee

For an efficient attacker A using at most t tagging queries, there is a PRF distinguisher B using at most $t + 1$ function queries such that

$$\Pr[A \text{ forges}] \leq \text{Adv}_F^{\text{prf}}(B) + 2^{-\ell}.$$

[BS23, §6.3, Thm. 6.2, pp. 224–225].

Why the PRF replacement works

The distinguisher's experiment

Given oracle access to a function O , run the forger. Answer each tagging query m with $O(m)$. When it outputs (m^*, τ^*) , check freshness and output 1 exactly when

$$m^* \notin \mathcal{Q} \quad \text{and} \quad \tau^* = O(m^*).$$

The last check uses at most one extra oracle query.

- If $O = F_K$, output 1 has probability p_{real} , the forger's actual success probability.
- If $O = f$ is uniform random, output 1 has probability $p_{\text{ideal}} \leq 2^{-\ell}$.

By the definition of PRF advantage,

$$p_{\text{real}} \leq p_{\text{ideal}} + |p_{\text{real}} - p_{\text{ideal}}| \leq 2^{-\ell} + \text{Adv}_F^{\text{prf}}(B).$$

[BS23, Def. 4.2, p. 133; proof of Thm. 6.2, pp. 224–225].

Questions to check the definitions

1. Alice sends (m, τ) twice. Has anyone forged?

No. Replay is outside the fresh-message forgery event.

2. Must a successful forged tag differ from every observed tag?

No. The message must be fresh. The tag may repeat.

3. Does seeing a million tags multiply the one-attempt random-table bound by a million?

No. An unqueried row is still uniform.

4. What changes if the attacker may submit many verification attempts?

The number of attempts must enter the security bound.

Experiments and verification queries: [BS23, §§6.1–6.2, pp. 218–223].

The ideas to carry forward

Authentication

A shared secret lets Bob verify a public message. Correctness handles honest messages. Unforgeability handles an outsider's attempts to create a valid pair for a fresh message.

The random-table proof

An unqueried input selects a secret uniform ℓ -bit value. One guess succeeds with probability $2^{-\ell}$, regardless of the attacker's computation or adaptive tagging queries.

The full table costs $\ell 2^n$ bits. A PRF replaces that table with a short key and computational security.

Next: efficient authentication constructions and how to handle longer messages.

[BS23, §§6.1–6.4, pp. 217–226].

Boneh–Shoup, version 0.6

- §6.1, pp. 217–219: the MAC interface and forgery experiment.
- §4.4.2, pp. 133–134: exposing random functions as queries arrive.
- §6.3, Thm. 6.2, pp. 224–225: PRFs give MACs.
- Optional: §6.2, pp. 220–223, for verification queries, and §7.6, pp. 275–276, for one-time unconditional MACs.

Hoffstein–Pipher–Silverman, second edition

§4.1, pp. 193–195: the contrasting public-key signature interface.

§5.1, pp. 208–213: counting.

§5.3.1–5.3.2, pp. 228–235: probability and conditioning.

[BS23, HPS14]. Page references are to printed book pages.

References

- [BS23] Dan Boneh and Victor Shoup.
A Graduate Course in Applied Cryptography.
Online textbook, 2023.
Version 0.6, January 2023.
URL: <https://toc.cryptobook.us/book.pdf>.
- [HPS14] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman.
An Introduction to Mathematical Cryptography.
Undergraduate Texts in Mathematics. Springer, New York, second edition, 2014.
[doi:10.1007/978-1-4939-1711-2](https://doi.org/10.1007/978-1-4939-1711-2).
- [Maj26] Hemanta K. Maji.
Lecture 19: Message authentication codes.
CS355, Purdue University, Fall 2026, 2026.
Instructor's supplied lecture slides, 23 pages.

Technical inputs: the random-function proof

Independent table entries

A uniform random $f: \mathcal{M} \rightarrow \mathcal{T}$ has independent uniform entries. Fixing some entries leaves the others independent and uniform.

Precise reference: Boneh–Shoup, v0.6, §4.4.2, pp. 133–134. The table-counting proof above establishes this directly.

Conditioning and averaging

For any finite transcript H and event E ,

$$\Pr(E) = \sum_{h: \Pr(H=h) > 0} \Pr(H = h) \Pr(E \mid H = h).$$

Background: Hoffstein–Pipher–Silverman, 2nd ed., §5.3.1–5.3.2, pp. 228–235. No hardness assumption is needed for the random-function theorem.

[BS23, HPS14].

Technical inputs: the computational preview

The PRF assumption

For every efficient oracle algorithm B , the quantity

$$\text{Adv}_F^{\text{prf}}(B) = \left| \Pr_K[B^{F_K} = 1] - \Pr_f[B^f = 1] \right|$$

is negligible in k . Here f is uniform random, and evaluation time and lengths are polynomially bounded in k .

Reference: Boneh–Shoup, Game 4.2 and Def. 4.2, pp. 132–133. MAC reduction: Thm. 6.2, pp. 224–225.

Tag length and security convention

$2^{-\ell(k)}$ must also be negligible, for example with $\ell = k$. Unique accepted tags make fresh-pair and fresh-message forgery equivalent. See Boneh–Shoup, Def. 6.1 and Game 6.1, pp. 217–219.