Introduction to Cryptology

5.3 - Message Authentication Codes (MACs)

Federico Pintore

Mathematical Institute, University of Oxford (UK)

Michaelmas term 2020

Message Integrity

Secrecy of messages is only one part of security.

Cryptographic schemes allow parties to
securely communicate over unsecured channels.

- What if messages were modified in transit (integrity)?
- What about authenticity?

Secrecy does not guarantee integrity

Consider the One Time Pad encryption scheme.

- Given a ciphertext, a new valid ciphertext can be produced by just flipping a single bit!
- Perfect secrecy is not contradicted, but it does not imply message integrity.

Secrecy does not guarantee integrity

Consider the One Time Pad encryption scheme.

- Given a ciphertext, a new valid ciphertext can be produced by just flipping a single bit!
- Perfect secrecy is not contradicted, but it does not imply message integrity.

Different cryptographic tools must be used
to achieve secrecy and integrity.

Message Authentication Codes (MACs)

Message Authentication Code is a cryptographic tool to ensure message integrity and authenticity.

Parties need to share a secret key beforehand.

Symmetric-key setting!

Message Authentication Codes (MACs)

Definition

A MAC *is a tuple of three PPT algorithms*

$$S = (\mathrm{KeyGen}, \mathrm{Mac}, \mathrm{Verify}) :$$

- $k \leftarrow \mathrm{KeyGen}(n)$*: takes the security parameter* n *and outputs a key* $k \in \{0,1\}^*$ *s.t.* $|k| \geq n$*.*
- $t \leftarrow \mathrm{Mac}(k, m)$*: the tagging algorithm takes a key* k *and a message* $m \in \{0,1\}^*$*, and outputs a tag* t*.*
- $0/1 \leftarrow \mathrm{Verify}(k, m, t)$*: a deterministic algorithm that outputs 0 if the tag is invalid, and 1 if it is valid.*

It is required that $\mathrm{Verify}(k, m, \mathrm{Mac}(k, m)) = 1$ *for every security parameter* n*, key* $k \leftarrow \mathrm{KeyGen}(n)$ *and message* $m \in \{0,1\}^*$*.*

Message Authentication Codes (MACs)

Notation: $\mathrm{Mac}_k(m) = \mathrm{Mac}(k, m)$, $\mathrm{Verify}_k(m, t) = \mathrm{Verify}(k, m, t)$.

If, for every n and k, $\mathrm{Mac}(k, \cdot)$ is only defined for $m \in \{0,1\}^{\ell(n)}$, the scheme is a fixed-length MAC for messages of length $\ell(n)$.

Mac may be randomised or deterministic.

Canonical Verification (when Mac is deterministic): it recomputes the tag and checks for equality.

MACs - Definition of security

Intuition: an adversary should not be able to efficiently produce a valid tag on a new message that was not authenticated before.

The threat model considers an adversary $\mathcal{A}$ that can see message-tag pairs – $\mathcal{A}$ is given access to a tagging oracle.

MACs - Definition of security

Message Authentication Experiment $\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n)$

Challenger Ch	Adversary $\mathcal{A}$
$k \leftarrow \mathrm{KeyGen(n)}$	
$Q = \{\text{queried } m\}$	Queries to Mac_k
	Outputs a forgery (m, t)

MACs - Definition of security

Message Authentication Experiment $\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n)$

Challenger Ch	Adversary $\mathcal{A}$
$k \leftarrow \mathrm{KeyGen(n)}$	
$Q = \{\text{queried } m\}$	Queries to Mac_k
	Outputs a forgery (m, t)

$\mathcal{A}$ wins the game, i.e. $\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n) = 1$, if:

- $\mathrm{Verify}_k(m, t) = 1$;
- $m \notin Q$.

MACs - Definition of security

Definition

A MAC $S = (\mathrm{KeyGen}, \mathrm{Mac}, \mathrm{Verify})$ *is secure if, for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(n)$ *s.t.*

$$\Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n) = 1) \leq \mathrm{negl}(n)\,.$$

MACs and Replay Attacks

When S is secure, an adversary cannot change the message (the tag is no longer valid) and cannot produce a new tag.

MACs and Replay Attacks

When S is secure, an adversary cannot change the message (the tag is no longer valid) and cannot produce a new tag.

However, the adversary can forward a valid pair message-tag.

MACs and Replay Attacks

When S is secure, an adversary cannot change the message (the tag is no longer valid) and cannot produce a new tag.

However, the adversary can forward a valid pair message-tag.

- The receiver cannot detect this replay attack.

MACs and Replay Attacks

When S is secure, an adversary cannot change the message (the tag is no longer valid) and cannot produce a new tag.

However, the adversary can forward a valid pair message-tag.

- The receiver cannot detect this replay attack.
- Common techniques to prevent replay attacks:
  - Time-stamps: append the current time to the message before authenticating it.
  - Counters: users maintain synchronised state.

MACs - Definition of Security (2)

Strong Message Authentication Experiment $\mathrm{Mac}_{\mathcal{A},S}^{\mathrm{s-unforg}}(n)$

Challenger Ch	Adversary $\mathcal{A}$
$k \leftarrow \mathrm{KeyGen(n)}$	
$Q = \{(m_i, \mathrm{Mac}_k(m_i))\}$	Queries to Mac_k
	Outputs a forge (m, t)

MACs - Definition of Security (2)

Strong Message Authentication Experiment $\mathrm{Mac}_{\mathcal{A},S}^{\mathrm{s-unforg}}(n)$

Challenger Ch	Adversary $\mathcal{A}$
$k \leftarrow \mathrm{KeyGen(n)}$	
$Q = \{(m_i, \mathrm{Mac}_k(m_i))\}$	Queries to Mac_k
	Outputs a forge (m, t)

$\mathcal{A}$ wins the game, i.e. $\mathrm{Mac}_{\mathcal{A},S}^{\mathrm{s-unforg}}(n) = 1$, if:

- $\mathrm{Verify}_k(m, t) = 1$;
- $(m, t) \notin Q$.

MACs - Definition of security (2)

Definition

A MAC $S = (\mathrm{KeyGen}, \mathrm{Mac}, \mathrm{Verify})$ *is strongly secure if, for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(n)$ *s.t.*

$$\Pr(\mathrm{Mac}_{\mathcal{A},S}^{\mathrm{s-unforg}}(n) = 1) \leq \mathrm{negl}(n)\,.$$

MACs - Definition of security (2)

Definition

A MAC $S = (\mathrm{KeyGen}, \mathrm{Mac}, \mathrm{Verify})$ *is strongly secure if, for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(n)$ *s.t.*

$$\Pr(\mathrm{Mac}_{\mathcal{A},S}^{\mathrm{s-unforg}}(n) = 1) \leq \mathrm{negl}(n)\,.$$

If the Mac in S is deterministic - and therefore the verification is canonical - then S is secure if and only if it is strongly secure.

MACs - Side Channel Attacks

Alternative threat model: $\mathcal{A}$ is also given access to $\text{Verify}_k(\cdot)$.

- If the verification is canonical, it makes not difference;
- a strongly secure MAC is secure also in this case.

MACs - Side Channel Attacks

Alternative threat model: $\mathcal{A}$ is also given access to $\text{Verify}_k(\cdot)$.

- If the verification is canonical, it makes not difference;
- a strongly secure MAC is secure also in this case.

In a real system, $\mathcal{A}$ may be able to obtain the time necessary to reject a pair message-tag.

MACs - Side Channel Attacks

Alternative threat model: $\mathcal{A}$ is also given access to $\text{Verify}_k(\cdot)$.

- If the verification is canonical, it makes not difference;
- a strongly secure MAC is secure also in this case.

In a real system, $\mathcal{A}$ may be able to obtain the time necessary to reject a pair message-tag.

- if Mac is deterministic and the verification does not use time-independent string comparison, then $\mathcal{A}$ can exploit the time differences to deduce new bytes of the tag!

MACs - Side Channel Attacks

This is a realistic attack!

MACs - Side Channel Attacks

This is a realistic attack!

Xbox 360 had a difference of 2.2 milliseconds in comparing j or $j+1$ bytes.

Attackers exploited it.

MACs - Side Channel Attacks

This is a realistic attack!

Xbox 360 had a difference of 2.2 milliseconds in comparing j or $j+1$ bytes.

Attackers exploited it.

Conclusion: canonical verification should compare all the bytes!

Further Reading I


N.J. Al Fardan and K.G. Paterson.
Lucky thirteen: Breaking the TLS and DTLS record protocols.
In Security and Privacy (SP), 2013 IEEE Symposium on, pages 526–540, May 2013.


J Lawrence Carter and Mark N Wegman.
Universal classes of hash functions.
In Proceedings of the ninth annual ACM symposium on Theory of computing, pages 106–112. ACM, 1977.


Jean Paul Degabriele and Kenneth G Paterson.
On the (in) security of IPsec in MAC-then-Encrypt configurations.
In Proceedings of the 17th ACM conference on Computer and communications security, pages 493–504. ACM, 2010.

Further Reading II


Ted Krovetz and Phillip Rogaway.
The software performance of authenticated-encryption modes.
In Fast Software Encryption, pages 306–327. Springer, 2011.


Douglas R. Stinson.
Universal hashing and authentication codes.
Designs, Codes and Cryptography, 4(3):369–380, 1994.