Lecture 11: Hash Functions and MACs

COSC362 Data and Network Security

Book 1: Chapters 11 and 12 – Book 2: Chapters 2 and 21

Spring Semester, 2021

Motivation

- Examples of message authentication codes (MACs) built from block ciphers (previously seen).
- However, these MACs are not commonly used in TLS.
- Another MAC, called HMAC, is widely used in TLS.
- Authenticated encryption mode GCM is also widely used in TLS (previously mentioned).
- Hash functions are typical building blocks in cryptography for MACs and digital signatures.

Outline

Outline

Hash Functions
- Security Properties
- Iterated Hash Functions
- Standardized Hash Functions
- Using Hash Functions

Message Authentication Code (MAC)
- MAC from Hash Function (HMAC)

Authenticated Encryption
- Combining Encryption and MAC
- Galois Counter Mode (GCM)

Hash Functions

A *hash function* H is a PUBLIC function s.t.:

- H is simple and fast to compute
- H takes as input a message m of ARBITRARY length and outputs a message *digest* $H(m)$ of FIXED length

Security Properties

- *Collision resistant:*
  - It should be infeasible to find any 2 different values x_1, x_2 s.t. $H(x_1) = H(x_2)$.
- *Second-preimage resistant:*
  - Given a value x_1, it should be infeasible to find a different value x_2 s.t. $H(x_1) = H(x_2)$.
- *Preimage resistant (one-way):*
  - Given a value y, it should be infeasible to find any input x such that $H(x) = y$.

An attacker who can break second-preimage resistance can break collision resistance.

Birthday Paradox

- Let a group of 23 randomly chosen people:
  - The probability that at least 2 have the same birthday is over 0.5.
- If choosing around $\sqrt{|S|}$ values from a set S, then the probability of getting 2 values the same is around 0.5.
- Let H be a hash function with output size of k bits:
  - Let H be seen as a random function.
  - Then $\sqrt{2^k} = 2^{k/2}$ trials are enough to find a collision with probability around 0.5.
- Today, 2^{128} trials would be considered infeasible:
  - Hash functions should have output of at least 256 bit to satisfy collision resistance.

Example with $|S| = 100$

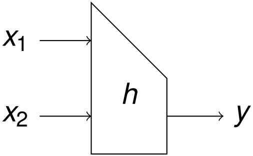
# trials	Collision prob.	# trials	Collision prob.
1	0	13	.55727
2	.01000	14	.61483
3	.02980	15	.66876
4	.05891	16	.71845
5	.09656	17	.76350
6	.14174	18	.80371
7	.19324	19	.83905
8	.24972	20	.86964
9	.30975	21	.89572
10	.37188	22	.91762
11	.43470	23	.93575
12	.49689	24	.95053

Iterated Hash Functions

- Cryptographic hash functions need to:
  - take arbitrary-sized inputs
  - produce a fixed-sized output
- From block ciphers, arbitrary-sized data can be processed by:
  - having a function processing fixed-sized data
  - using it repeatedly
- An *iterated hash function* splits the input blocks of fixed size and operates on each block sequentially using the same function with fixed-sized inputs.
- *Merkle-Damgård:* using a *compression function h* taking fixed-sized inputs and applied to multiple blocks of the message.

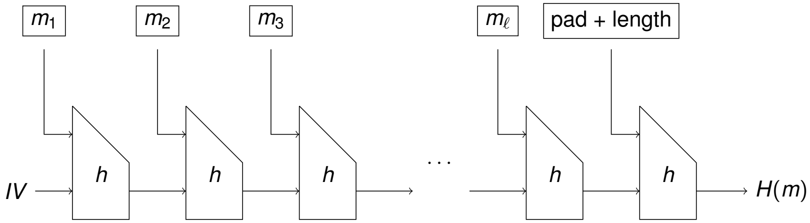
Compression Function h

h takes 2 n-bit input strings x_1 and x_2 and produces an n-bit output string y:



Merkle-Damgård Construction

1. Break message m into n-bit blocks $m_1 \| m_2 \| \ldots \| m_\ell$.
2. Add padding and an encoding of the length of m:
   - This process may, or may not, add one block.
3. Input each block into compression function h along with chained output:
   - Use IV to get started.



Using Merkle-Damgård Construction

- *Security:* if compression function h is collision-resistant then hash function H is collision-resistant.
- But security weaknesses:
  - *Length extension attacks:* once there is one collision, easy to find more.
  - *Second-preimage attacks:* not as hard as they should be.
  - *Collisions for multiple messages:* found without much more difficulty than collisions for 2 messages.
- *Examples:* MD5, SHA-1, SHA-2 family.

MDx Family

- Proposed by Ron Rivest and widely used in the 90s.
- Deployed family members: MD2, MD4 and MD5.
- 128-bit output.
- All are broken:
  - Real collisions have been found.
  - MD5 collisions can be found in 1 minute on a PC (2006).

Secure Hash Algorithm (SHA)

- Based on MDx family design:
  - More complex design.
  - Larger output of 160 bits.
- Published by US standard agency NIST (previously called NBS) in 1993:
  - Called SHA-0.
  - Replaced by SHA-1 with very small changes to algorithm (1995).
- Both have been broken:
  - SHA-0: collisions found in 2004.
  - SHA-1: collisions found in 2017 s.t. attack 100,000 faster than brute force search.

SHA-2 Family

- Developed in response to (real and theoretical) attacks on MD5 and SHA-1.
- Standard: FIPS PUB 180-4 (Aug. 2015).
- Known as SHA-2.

Name	Hash size	Block size	Security match
SHA-224	224 bits	512 bits	2 key 3DES
SHA-512/224	224 bits	1024 bits	2 key 3DES
SHA-256	256 bits	512 bits	AES-128
SHA-512/256	256 bits	1024 bits	AES-128
SHA-384	384 bits	1024 bits	AES-192
SHA-512	512 bits	1024 bits	AES-256

Padding in SHA-2 Family

- *Message length field:*
  - 64 bits when block length is 512 bits.
  - 128 bits when block length is 1024 bits.
- Always at least one bit of padding.
- There is an exact number of complete blocks:
  - After the 1st bit “1”, enough bits “0” are added.
  - Length field is then added.
- Adding the padding and length field sometimes add an extra block, and sometimes does not.

SHA-3

- Crisis in hash function design late 2000s:
  - MDx and SHA families all based on same basic design.
  - Unexpected attacks against them in recent years.
- NIST announced a competition for new hash standard SHA-3 (Nov. 2007):
  - Entries closed in Oct. 2008 with 64 original submissions.
  - 14 went through Round 2.
  - 5 finalists announced in Dec. 2010.
  - Keccak selected as winner in Oct. 2012.
- Keccak does not use compression function:
  - Instead, a *sponge function*.
- Standard: FIPS PUB 202 (Aug. 2015).

Using Hash Functions

- Applying a hash function is NOT encryption:
  - Hash computation does NOT depend on a key.
  - Not possible to go backwards to find the input in general.
- Helping to provide data authentication:
  - But not providing it alone!
  - Authenticating the hash of a message to authenticate the message.
  - Building block for MACs.
  - Building block for signatures.

Storing Passwords for Login

- Storing user passwords on servers using hash functions.
- Storing salted hashes of passwords:
  1. Pick at random *salt*
  2. Compute $h = H(pw, salt)$
  3. Store $(salt, h)$
- Easy to check entered password pw': $h = H(pw', salt)$?
- Hard to recover *pw* from *h* assuming that *H* is preimage resistant.
- The attacker needs to store a different dictionary for EACH *salt*.
- Using a *slower* hash function slows down password guessing.

Outline

Message Authentication Code (MAC)

- Message authentication code (MAC) is a cryptographic mechanism to ensure message integrity:
  - Inputs: message M of arbitrary length, secret key K
  - Ouput: (short) fixed-sized tag $T = \mathrm{MAC}(M, K)$
- Alice, the sender, appends the tag T to the message M (in the clear).
- Bob, the recipient, computes $T' = \mathrm{MAC}(M', K)$ with the received message M', and checks whether $T = T'$.

Properties

- *Unforgeability:* it is not feasible to produce a valid pair (M, T) s.t. $T = \mathrm{MAC}(M, K)$ without knowledge of K.
- *Unforgeability under chosen message attack:*
  - The attacker has access to a *forging oracle* s.t. on input any message M of the attacker's choice, the oracle outputs the tag $T = \mathrm{MAC}(M, K)$.
  - The attacker should not be able to produce a valid forgery that was not asked to the oracle.

MAC from Hash Function (HMAC)

- Proposed by Bellare, Canetti and Krawczyk in 1996.
- Built from ANY iterated hash function H:
  - *Examples:* MD5, SHA-1, SHA-256, ...
- Standard: FIPS PUB 198-1 (July 2008).
- Used in many applications such as TLS and IPSec.

Construction

- H: iterated cryptographic hash function
- M: message to be authenticated
- K: key padded with zeros to be of block size of H
- $\texttt{opad}$: fixed string $\texttt{0x5c5c5c...5c}$
- $\texttt{ipad}$: fixed string $\texttt{0x363636...36}$
- $||$: concatenation of bit strings

$$\mathrm{HMAC}(M, K) = H\left((K \oplus \texttt{opad}) \parallel H((K \oplus \texttt{ipad}) \parallel M) \right)$$

Security

- HMAC is secure if:
  - either H is collision resistant
  - or H is a pseudorandom function
- Designed to resist length extension attacks:
  - Even if H is a Merkle-Damgård hash function.
- Often used as a *pseudorandom function* for deriving keys in cryptographic protocols.

Outline

Authenticated Encryption

- Let Alice and Bob share a key K.
- Alice wants to send to Bob a message M with *confidentiality* and *authenticity/integrity*.
- 2 options:
  - Split K into 2 parts K_1, K_2, encrypt with K_1 (confidentiality) and use K_2 with a MAC (authenticity/integrity).
  - Use the *authenticated encryption* algorithm providing both confidentiality and authenticity/integrity.

Combining Encryption and MAC

3 options:

- Encrypt and MAC: encrypt M, apply MAC to M, and send the ciphertext C and the tag T.
- MAC then encrypt: apply MAC to M, then encrypt $M||T$, and send the ciphertext C.
- Encrypt then MAC: encrypt M, apply MAC to the ciphertext C, and send C and the tag T.

Encrypt-then-MAC option is the safest approach:

1. $C = \text{Enc}(M, K_1)$
2. $T = \text{MAC}(C, K_2)$
3. Send $C||T$

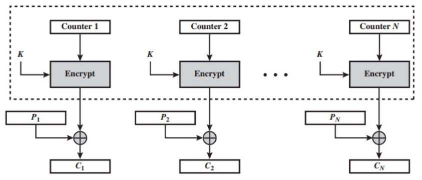
Authenticated Encryption Mode

- MAC-then-encrypt construction used in older versions of TLS (SSL 3, TLS 1.0 and 1.1):
  - Several security vulnerabilities.
- Combined authentication encryption modes in recent versions (TLS 1.2 and 1.3):
  - Supporting CCM and GCM modes of operation for block ciphers.
  - Allowing data to be only authenticated (not encrypted) with *authenticated encryption with associated data* (AEAD).

CTR Mode for Block Ciphers

CTR is a synchronous stream cipher:

- A counter is initialised using a randomly chosen nonce N.
- Keystream generated by encrypting successive values of the counter:
  - $O_t = E(T_t, K)$ where $T_t = N||t$ is the concatenation of N and block number t.



(a) Encryption

Encryption: $C_t = O_t \oplus P_t$

Decryption: $P_t = O_t \oplus C_t$

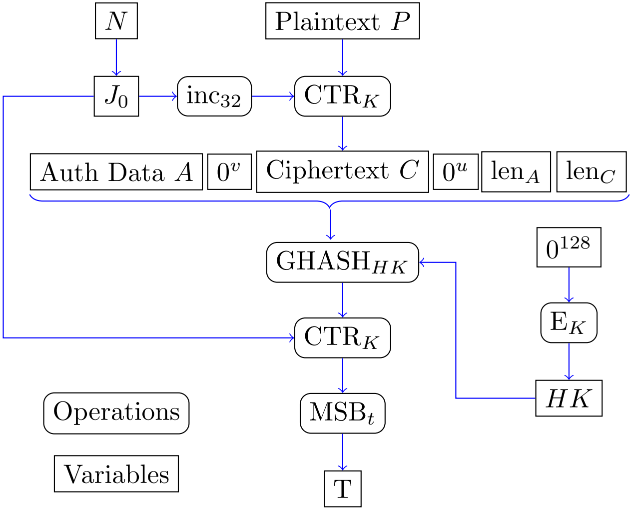
Galois Counter Mode (GCM)

- CCM mode (Lecture 8) is NOT suitable for processing of streaming data:
  - Formatting function for N, A, P requires knowledge of length of A and P.
- Galois counter mode (GCM) overcomes such limitation.
- Standard: NIST SP-800 38D.
- AES with GCM faster than AES with HMAC:
  - Hardware support of AES and carry-less addition in modern Intel chips.

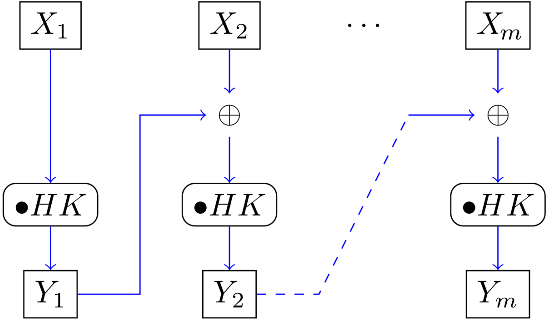
Algorithm

- Combining CTR mode on block cipher E (e.g. AES) with a special keyed hash function GHASH:
  - GHASH uses multiplication in finite field $GF(2^{128})$.
- Inputs: plaintext P, authenticated data A, nonce N.
- Outputs: ciphertext C and tag T.
- Length len_A of A and length len_C of C are 64-bit values:
  - u and v are minimum numbers of zeros required to expand A and C to complete blocks, respectively.
- Length t of T is 128 bits and length of N is 96 bits.
- Initial block input is $J_0 = N||0^{31}||1$.
- Function inc_{32} increments the 32 MSB of input string by 1 modulo 2^{32}.

Algorithm


N
Plaintext P
J_0
inc_{32}
CTR_K
Auth Data A
0^v
Ciphertext C
0^u
len_A
len_C
0^{128}
GHASH_{HK}
E_K
CTR_K
HK
Operations
MSB_t
Variables
T

GHASH



- Output is $Y_m = \mathrm{GHASH}_{HK}(X_1, \ldots, X_m)$
- Operation $\bullet$ is multiplication in the finite field $GF(2^{128})$
- $HK = E(0^{128}, K)$ is the hash subkey.

Decryption

- Elements transmitted to Bob, the recipient:
  - ciphertext C, nonce N, tag T, authenticated data A
- Bob computes the tag T' using the shared key K and received C, N, A.
- Bob compares T' with received T:
  - If $T' \neq T$ then output “invalid”.
  - If $T' = T$ then the plaintext P is computed by generating the same keystream from CTR mode as for encryption.