Class Notes Monday

September 27th 2021

**Hash Function:**

A keyless algorithm that takes a variable length input X and reutnrs an output Y of fixed length N (message digest length)

**Cryptographic Hash:**

**1.)** Preimage Resistance

**2.)** Second Preimage Resistance

**3.)** Collision Resistance

What we really want is that a hash function to behave like a random function.

**Preiamge Resistance:**

Given an output y, it should be hard to find any input x suc hthat h(x) - y.

A gneric attack would require around 2^n operations.

**Preimage Resistance Example:**

In a password file one does not store(username and passwr0d) but stores (username( hash(pasword))

This is sufficient to verify a password

An attacker that acptures the password file has to find a preimage

**Second Preiamge Resistance:**

A given output y and input x1 suc hthat h(x1) = y, it should be hard to find any other input x2 such that h(x2) = y.

Meaning you should never be able to find out x2 given an input and output.

**SPR Example:**

Store the hashes of your files on your comptuer or write them down on paper.

If the attacker modifies the files (e.g. put a malicious code inside them) and can still obtain the same hash value, then the attacker can fool you.

**Collision Resistance:**

It should be hard to find two inputs x1 and x2 such that h(x1) = h(x2).

Meaning you should never be able to get the same output given two different inputs.

A generic attack would requre around 2^n/2 operations

**CR Example:**

Hacker prepares two verison of a softare driver for an OS wehre x is rcorrect and x’ contains a backdoor that allows access to the machine

Hacker sends x to the OS compnay for inspection

OS company signs the h(x) with their private key

Hacker distrubes x’ to the users to verify the signagure on x wit hteh OS companys public key

Sicne h(x) = h(x’) the signature works.

**Birthday Paradox:**

How many people need to be in a room before it is more likely than not that at least two people share a birthday?

**Swiss Army Knife Of Cryptography (Hash Functions):**

1.) Digital Signatures

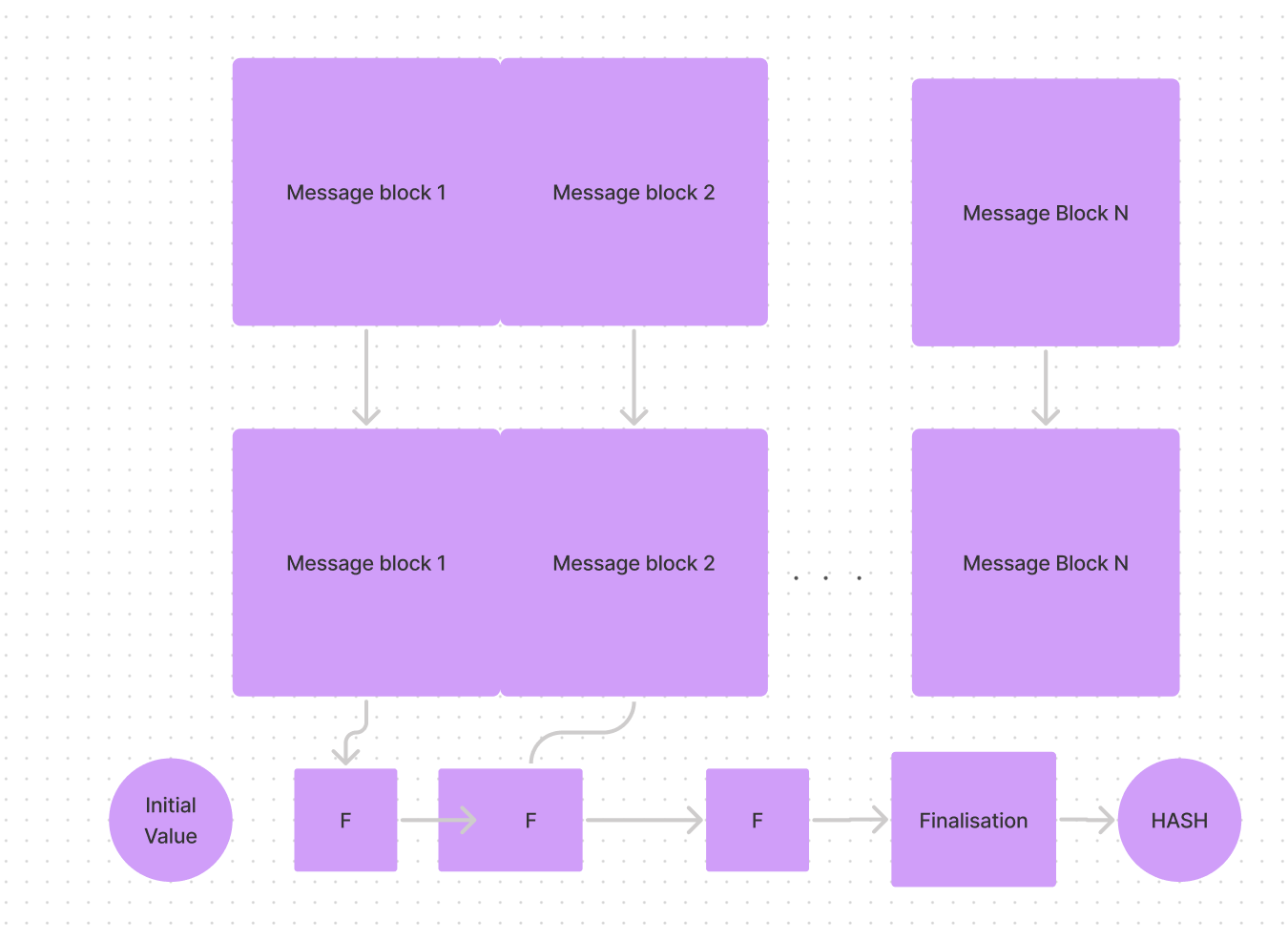
2.) Message authenticaiton codes

3.) PRNG

4.) Key derivation functions

**Iterated Construction of Hash Function:**

* Many hash functions adopt an iterative design to accommodate a variable length input
* Merkle\_Damgard construction is the most famous one
* Message M is divded into fixed length blocks and a suitable padding is applied
* Message blocks are compressed one after the other to produce h sub I using a compression function f
* H(I) is called a chaining variable and used in the compression of the next message block
* Compression function uses M(I) and H(I-1) to produce next H(I)
* H(0) is fixed and specified as a part of the hash function specifications
* A finalization operatin is applied to overcome some attacks



**Hash Functions Types:**

* Meyer-Davis
* Matyas-Meyer-Oseas
* Miyaguchi-Prenell

**MD4 Hash Function Block Cipher:**

-Developed by Ronald Rivest in 1990

-Blcok size is 512 bits

-Message digest lengh is n = 128 bits

-Number of rounds is 48

-A,B,C,D are state words of 32 bits. Mi is the ith message block.

-Collision resistance: Generation a collision takes a few microseconds

-Preimage resistance: 2^95 operations

-Retired in 2011 because of a found attack

**MD5:**

Very similar to MD4 except it added an extra XOR after the bitshift.

Collision resistance: Generating a colliion only requres 2^18

Preimage resistance : 2^123

Many attacks over the years showed that collision resistance could break MD5 as well

**SHA-1:**

Developed by the NSA, first publisehd in 1993 as SHA-0 but it was found unsecure

A small modifcation is made in 1995 and ranamed as SHA-1

* Block size 512 bits

**SHA-2:**

Still consdiered a M&D Type which wasn’t good because it was based on previous flaws

**SHA-3:**

* NIST acconued a Hash Competition
* Received 64 submissions
* SHA-3 Zoo is dedicated to this competition
* Keccak is the winner of the Design. Designed by Guido Bertoni, Joan Daemen, Peters and Giles
* Uses a a new design called “Sponge hash function”

**Sponge Construction for Hash Functions:**

Pi are input, Zi are hashed output. The unused capacity “c” should be twice the desired resistance to collision or preimage attacks.

**Passwords:**

We don’t store the passwords but we store their hasshes. Leakage of the database should not leak the apsswords. Hence the ahsh funciton should be preimate resistant.

* Attacks: Exhaustive Search & Dictionary Attack
* Counter Measure: Add SALT to each password before the hash function
* Counter Measure: Slowdown the hash computation speed

**SALT:**

Instead of the hash of a password, store the hash of password||salt where salt is randomly generated public value. Thus, the attacker need to genreate tables for every salt which is not practical.

**Hash Computation Speed:**

Instead of using the hash value of a password, repeatedly use the hash output as the hash input. For instance, 1024 times. This way the attackers job bcomes 1024 tiems harder.

**Password-Based Key Derivation function 2:**

A key derivation function that is part of RSA Laboratories Public-Key Cryptograph Standards

DK = PBKDF2(PRF, Password, Salt, c, dkLen)

* PRF is pseudorandom function of two aprameters
* C is the number of iterations desired
* DkLen is the desriged length of the dervied key
* DK is the generated dervied key

**AUTHENTICATION**

**Message athentication Codes:**

Why authentication is needed – Eve may try to alter or fake message

Tag Generatiion Algorithms & Verificaiton Alrogithms needed for this

Main difference between MAC algorithm and Hash funcion is MAC uses a key

Only keyholder can generate/verify MACs

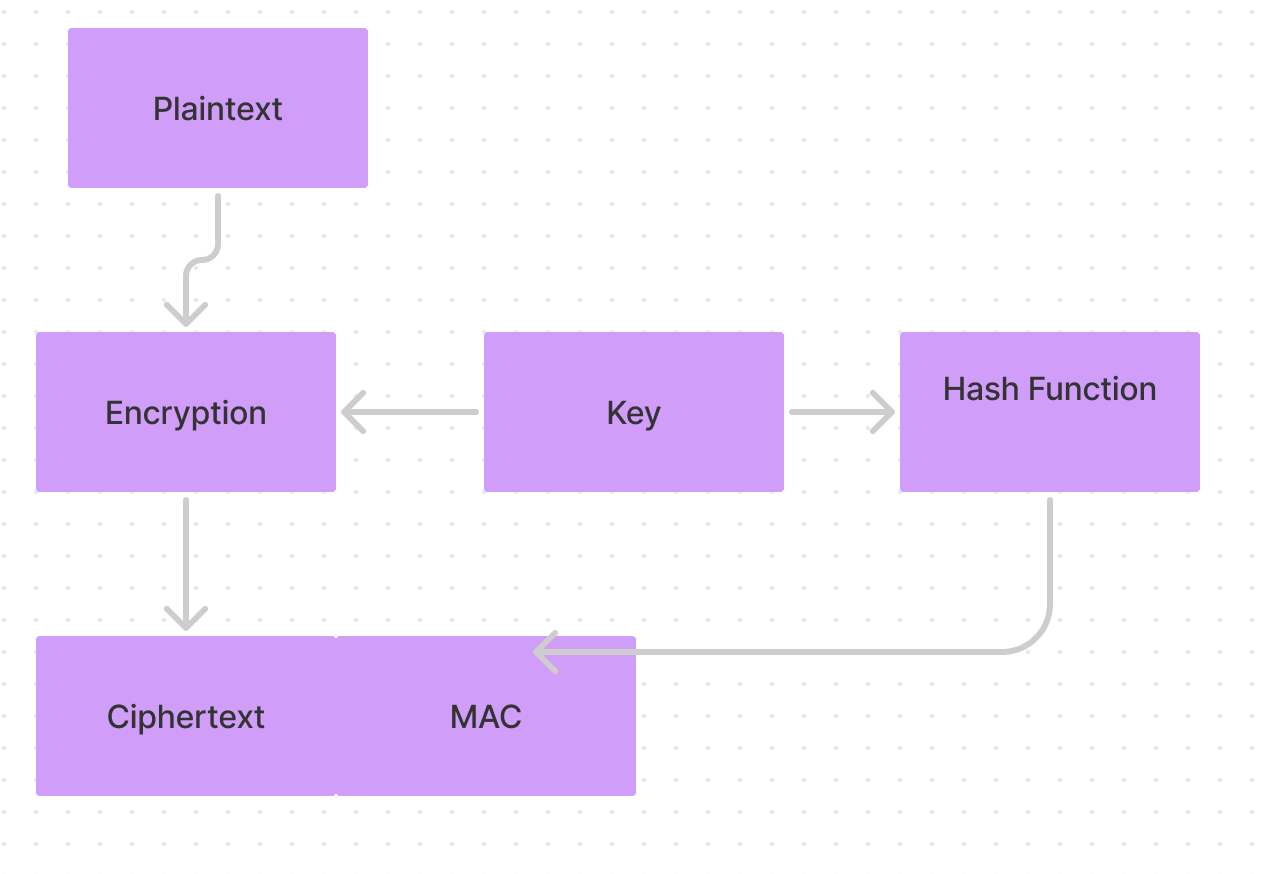
Message Authentication Code (MAC)

* Services:
  + Integrity
  + Authenticity
* Does not prevent:
  + Replay attacks
  + Message deletion or reordering
* Is Symmetric: Either Alice or Bob can create MACs

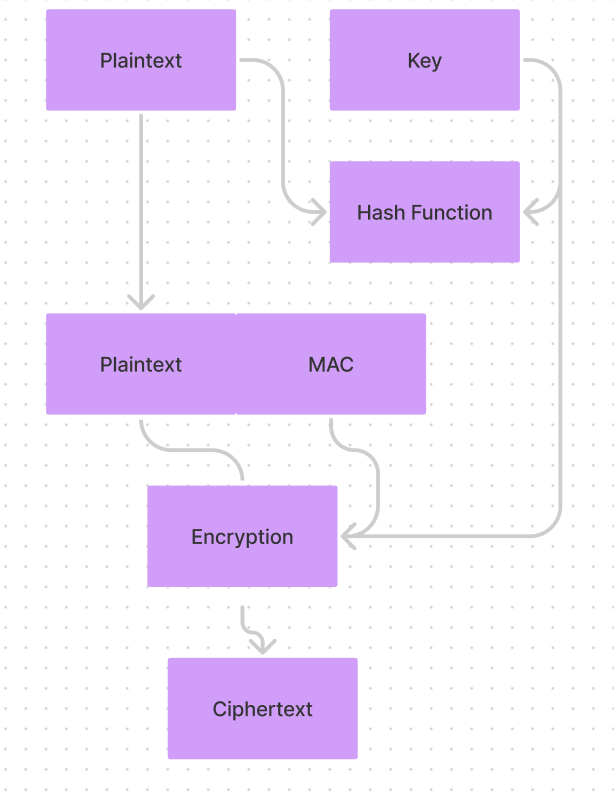
When the MAC is obtained via a hash function, the follwing methods are widley sued

* MAC and Encrypt
* MAC then Ecnrypt
* Encrypt then MAC

**MAC & Encrypt**



**MAC then Encrypt**



**Encrypt then MAC**

