

Cryptography basics – Integrity: Hashes and MACs

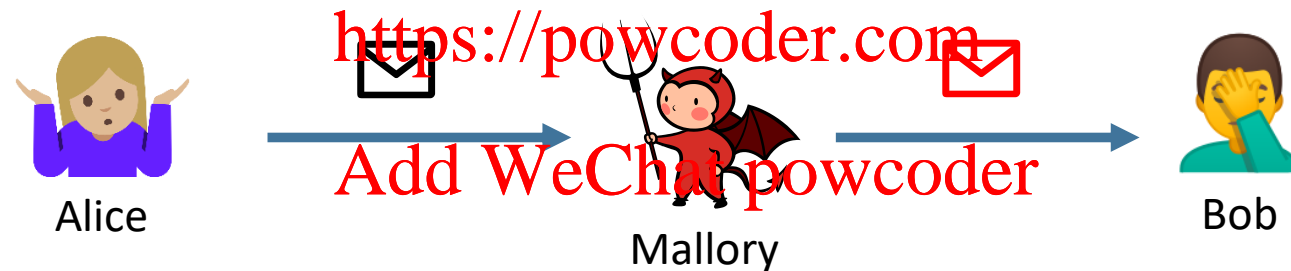
Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

ECEN 4133
Jan 19, 2021

Alice and Bob

Alice wants to send message m to Bob

- Can't fully trust the messenger or network carrying the message
- Want to be sure what Bob receives is actually what Alice sent



Threat model:

- Mallory can see, modify, forge messages
- Mallory wants to trick Bob into accepting a message Alice didn't send

Solution:

Message Authentication Code (MAC)

One approach:

- Alice computes $\mathbf{v} := f(\mathbf{m})$
- Bob verifies that $\mathbf{v}' = f(\mathbf{m}')$



Function f ?

Easily computable by Alice and Bob;
not computable by Mallory

(Idea: Secret only Alice & Bob know)

We're sunk if Mallory can learn
 $f(\mathbf{x})$ for any $\mathbf{x} \neq \mathbf{m}$!

Candidate f : Random Function

Input: Any size

Output: Fixed size (e.g. 256 bits)

Defined by a giant lookup table that's filled in by flipping coins

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

in	→	out
0	→	0011111001010001...
1	→	1110011010010100...
2	→	0101010001010000...
⋮		⋮

Completely impractical [why?]

Provably secure [why?]

(Mallory can't do better than randomly guessing)

Hash Functions

Random Functions are impractical

Hash functions approximate a random function:

- Any size input
- Fixed size output (e.g. 256 bits)
- Hard (but not impossible!) to invert (given output, find input)

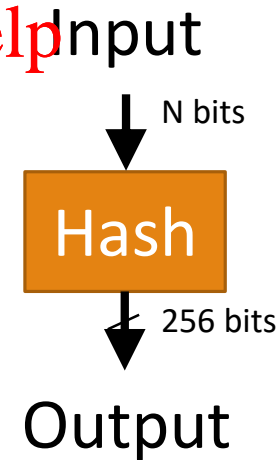
Properties of a secure cryptographic hash:

- First pre-image resistant – Given $H(m)$, hard to find m
- Second pre-image resistant – Given m_1 , hard to find m_2 s.t $H(m_1) == H(m_2)$
- Collision resistant – Hard to find $m_1 \neq m_2$ s.t $H(m_1) == H(m_2)$

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Example Hash Function: SHA256

What is **SHA256**?

“Cryptographic hash function”

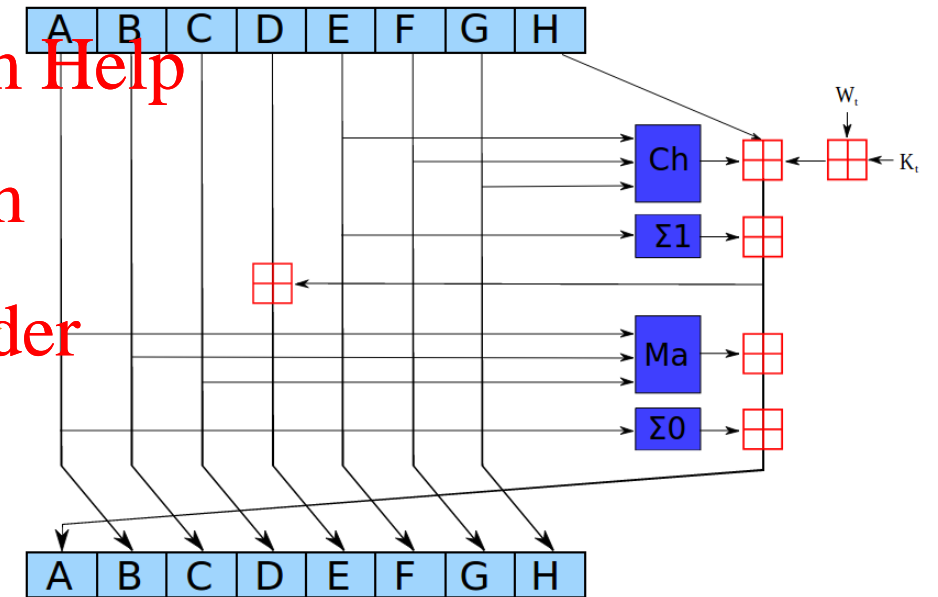
Input: arbitrary length data (No key)

Output: 256 bits

Built with “compression function” h

(256 bits, 512 bits) in \rightarrow 256 bits out

Designed to be really hairy (64 rounds of this:)



Compression functions

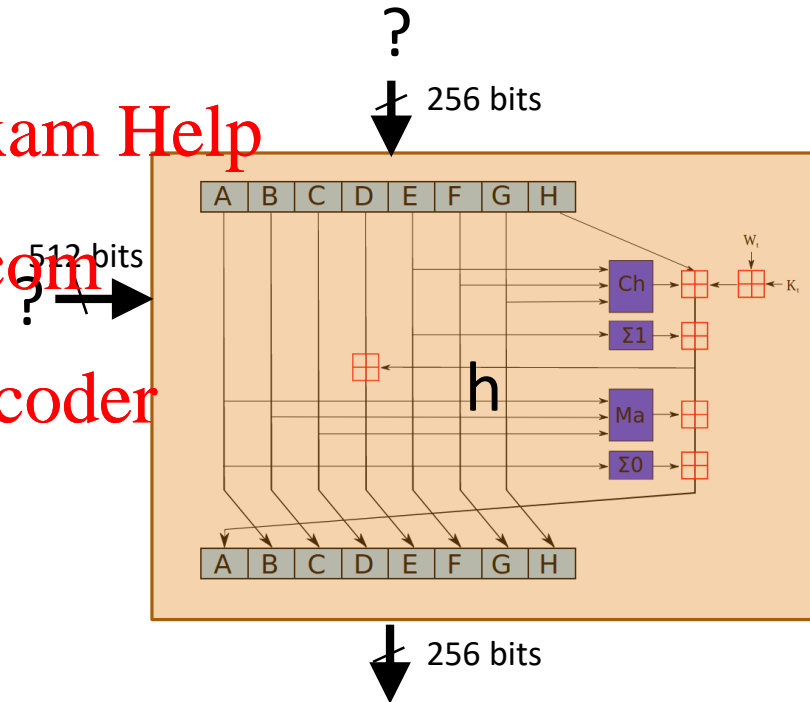
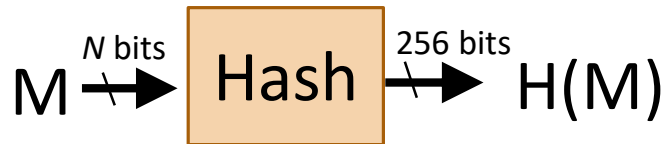
Compression function h take (two) fixed-length inputs, produce fixed-length output

Assignment Project Exam Help

How do we build a hash function from h that takes an arbitrary length input?

<https://powcoder.com>

Add WeChat powcoder



Solution: Merkle–Damgård Construction

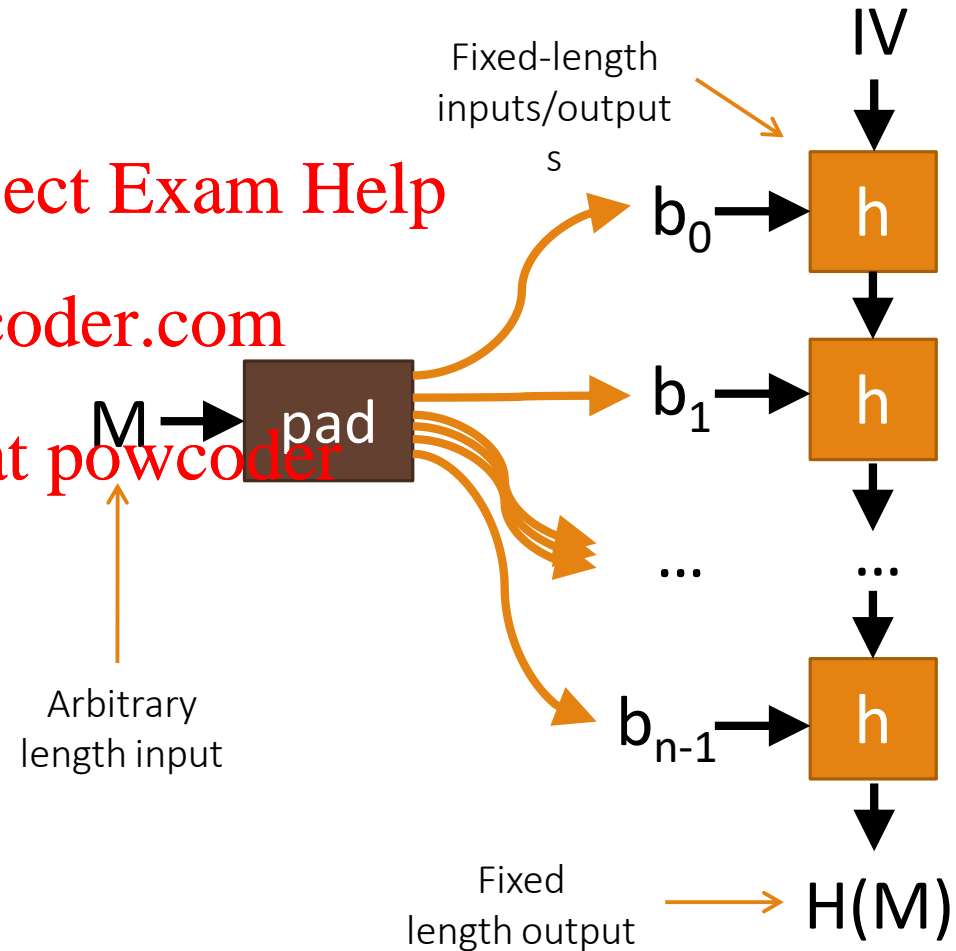
Entire algorithm:

1. Pad input M to a multiple of 512 bits
2. Break into 512-bit blocks b_0, b_1, \dots, b_{n-1}
3. $y_0 = \text{const (IV)}$,
 $y_1 = h(y_0, b_0)$,
...,
 $y_i = h(y_{i-1}, b_{i-1})$
4. Return y_n

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



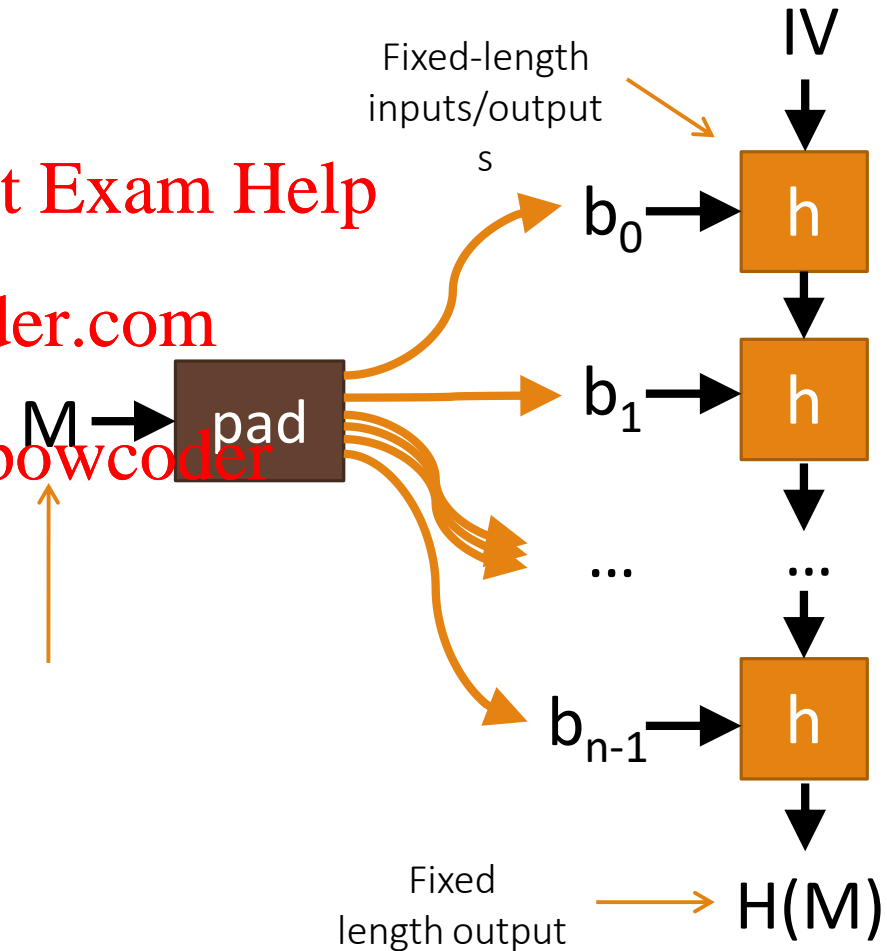
Merkle–Damgård Problem: Length Extension Attacks

Given $H(m)$, attacker can compute $H(m \parallel x)$
for arbitrary x , **without knowing m !** [How?]

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Length Extension Attack

Given $H(m)$, attacker can compute $H(m || x)$
for arbitrary x , **without knowing m !**

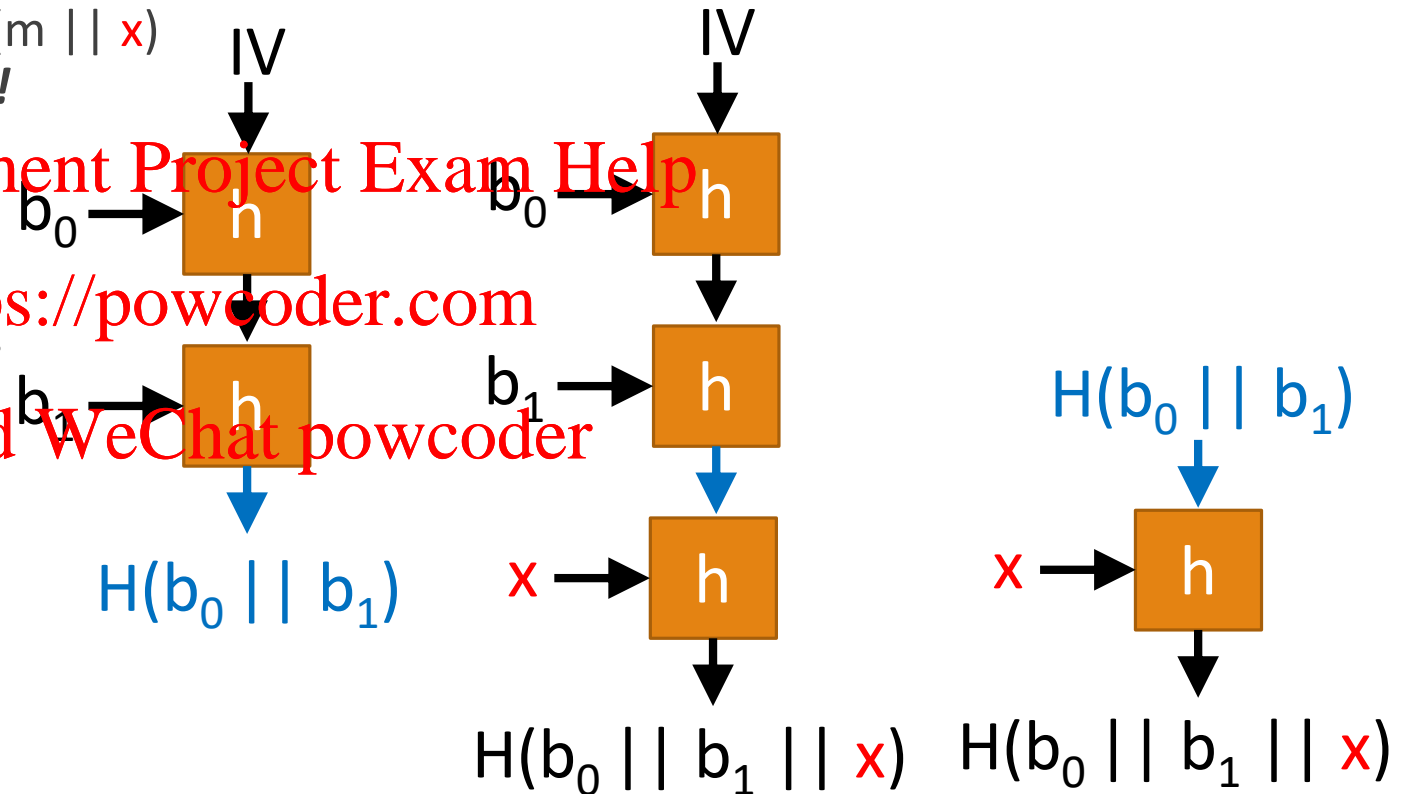
$H(m)$ is essentially the *internal state*
of the hash function after hashing m .

Can just feed $H(m)$ as IV to compute
hash of $H(m || x)$

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Other hash functions

MD5

Once ubiquitous, *broken in 2004*

Turns out to be easy to find *collisions*
(pairs of messages with same MD5 hash)

You'll investigate this in Project 1

Assignment Project Exam Help

<https://powcoder.com>

SHA1

Deprecated in 2011, but still widely used

Collisions found in 2017:

Took 9,223,372,036,854,775,808 SHA1 computations to find (6,500+ CPU-years)

Don't use!

Add WeChat powcoder

SHA3

Different “sponge” construction

Not susceptible to length-extension

Try hash functions yourself!

```
$ echo -n "Hello, World" | sha256sum
```

```
03675ac53ff9cd1535ccc7dfcdfa2c458c5218371f418dc136f2d19ac1fbe8a5 -
```

```
$ echo -n "Hello, World" | openssl dgst -sha3-256
```

```
(stdin)= 844af7bf6a5d414359cd8445cb52d515997410e1668e00c8469ea8728c4ffe8
```

```
$ echo -n "Hello, World" | openssl dgst -sha3-256
```

```
(stdin)= 7cccf7d7c35d3b321c85bce74564b8da936bcd9eed4877ad775f262f106b71f3d
```

Hash functions -> Integrity?

Can we use hash functions to provide integrity?



Add WeChat powcoder

Hash functions -> Integrity?

Can we use hash functions to provide integrity?



Not directly: Mallory could still change m to m' , and compute $H(m')$

[Alternative?]

Keyed hash function: Message Authentication Code (MAC)

Assume Alice and Bob have a
shared secret k

Alice computes MAC
over the message m
with her key k :

$$v = \text{MAC}_k(m)$$

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Alice



Mallory

$m', v'?$



Bob

Mallory doesn't know k , so cannot produce $v' = \text{MAC}_k(m')$

Building a MAC from a hash function: HMAC

Assignment Project Exam Help

$$\text{HMAC-SHA256}(k, m) = \text{SHA256}(k \oplus c_1 \parallel \text{SHA256}(k \oplus c_2 \parallel m))$$

<https://powcoder.com>

Add WeChat powcoder

SHA256 function

takes arbitrary length input,
returns 256-bit output

Not vulnerable to length extension!

Using HMAC

```
$ echo "Hello, World" | openssl dgst -sha256 -hmac "NotVerySecret"  
(stdin)= a502137ae2ad88313fcb267747a0474f0286ae671a1e639d5448a82bc5efb44a
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Tricky question: are hashes secure?

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Tricky question: are hashes secure?

Answer: **we don't know!**

Hashes have been broken in the past:

- MD5 introduced in 1992, first collision in 2004
- SHA1 introduced in 1995, first collision in 2017
- SHA2 introduced in 2001, no known collision ...yet!
- SHA3 introduced in 2015, no known collision ...yet!

We know collisions exist, but hope they are difficult to find [Why?]

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

MAC crypto game

Game against Mallory

1. Give Mallory $\text{MAC}(K, m_i) \forall m_i \in M$ and M (but not K !)
 2. Mallory tries to discover $\text{MAC}(K, m')$ for a new $m' \notin M$
 3. If Mallory succeeds, MAC is **insecure**
-

Assignment Project Exam Help

Other uses for hashes/HMACs? <https://powcoder.com>

Add WeChat powcoder