

Integrity (Crypto Lecture #1)



Administrivia

Office hours

SSD forms

Project one: released next week, watch Piazza for announcement
Find your partner!

Homework 1 due on Friday, September 14, 6 PM
Don't be late!
Submit w/ Canvas

Submission is through Gradescope, **must be a PDF**

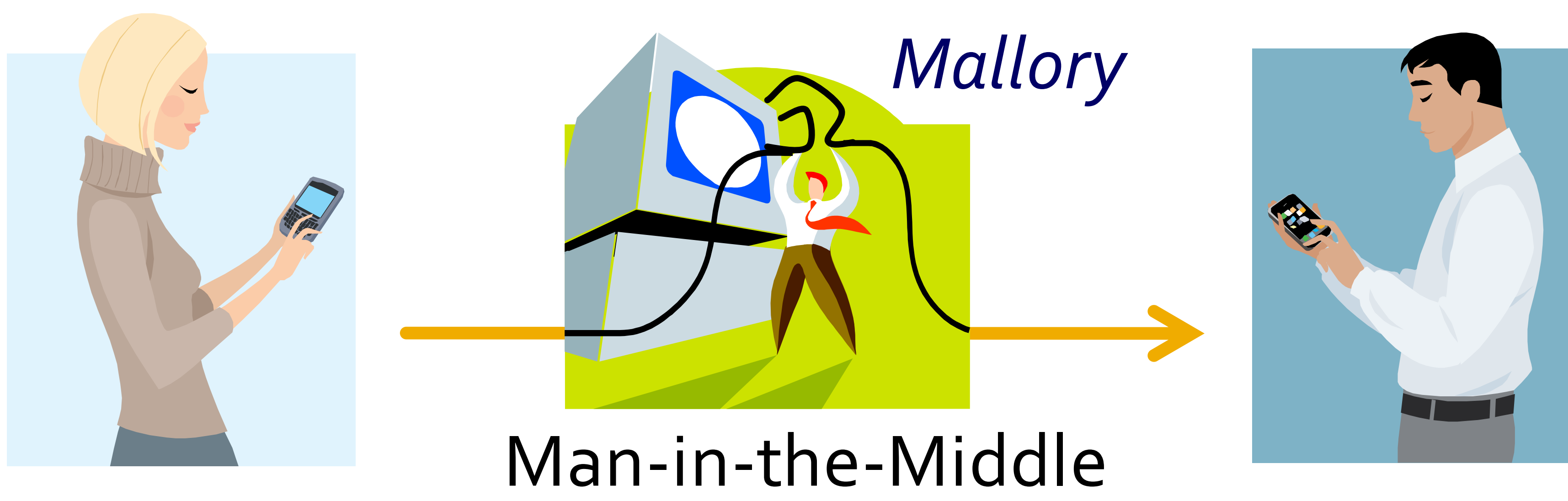
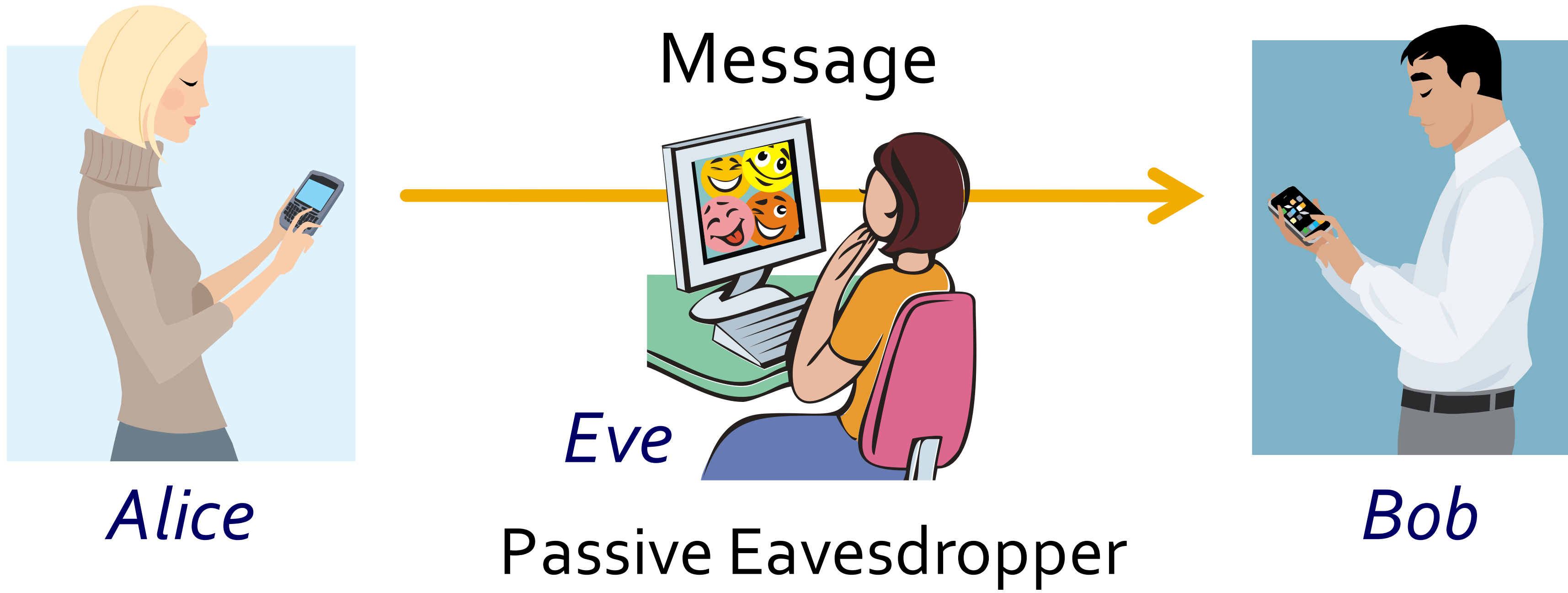
Cryptography?

What is cryptography?

Why study cryptography?

What do we want to use cryptography to defend against?

What properties does cryptography provide / what properties do we want from the cryptography that we use?



Properties of a Secure Channel

Confidentiality

Integrity 

Authentication

Message Integrity

An attacker cannot modify messages without being detected.

Goal: Message Integrity

Alice wants to send message ***m*** to Bob, *but...*

- We don't trust the messenger
- I.e., we don't trust the network



Want to be sure **what Bob receives** is actually **what Alice sent**.

Goal: Message Integrity

Threat Model

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

man-in-the-middle



One Approach...

1. Alice computes $\mathbf{v} := f(\mathbf{m})$

2. E.g. \mathbf{m} = "Attack at dawn", $f(\mathbf{m})$ = 628369867...



3. Bob verifies that $\mathbf{v}' = f(\mathbf{m}')$, and accepts the message if and only if this is true.

Function f ?

Bob accepts the message iff $\mathbf{v}' = \mathbf{f}(\mathbf{m}')$.

We want \mathbf{f} to be easily computable by Alice and Bob, but **not** computable by Mallory.

(As we shall see) we lose if Mallory can learn $\mathbf{f}(\mathbf{x})$ for any $\mathbf{x} \neq \mathbf{m}$!



Candidate: Random Function

Input: Any size up to huge maximum

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

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

0	→	0011111001010001...
1	→	1110011010010100...
2	→	0101010001010000...

Random Function

Looks pretty **secure**

Mallory can't do better than random guessing

Looks pretty **impractical**

0	→	0011111001010001...
1	→	1110011010010100...
2	→	0101010001010000...

Candidate: Pseudorandom Function (PRF)

Want a function that's practical but “looks random”...

Building a PRF

Start with a big family of functions $f_0(), f_1(), f_2(), \dots$ all known to Mallory
 $f_i: \{0,1\}^n \rightarrow \{0,1\}^n$

Let $g()$ be $f_k()$, where k is a secret value (or “key”) known only to Alice and Bob

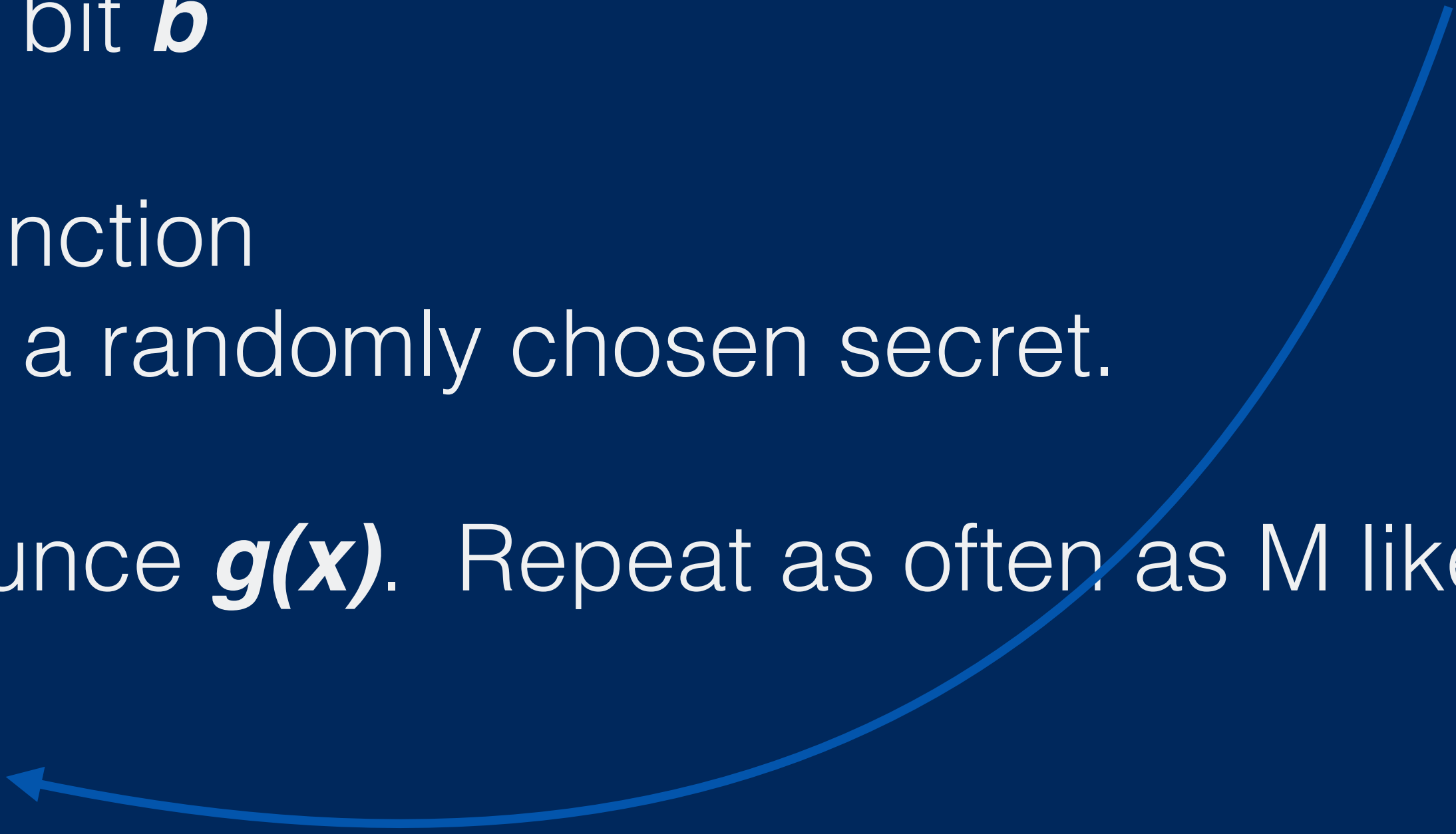
k is n bits, chosen randomly

Kerchoff’s Principle: *A cryptosystem should be secure even if everything about the system, except the key, is public knowledge*

Formal Definition of a Secure PRF

1. We flip a coin secretly to get bit b
2. If $b = 0$, let g be a random function
If $b = 1$, let $g = f_k$, where k is a randomly chosen secret.
3. Mallory chooses x ; we announce $g(x)$. Repeat as often as M likes.
4. Mallory guesses b quickly

Mallory can always
win slowly!



Security Definition: We say $g()$ is a secure PRF if Mallory can't select b any better than random guessing

PRF for Alice and Bob

1. Let $g() = f_k()$ be a secure PRF, where k is a random key known only to Alice and Bob.
2. Alice computes $v := f_k(m)$.
3. Bob verifies $v' = f_k(m')$, accepts if and only if this is true.



Annoying Question!

Q: Do PRF's actually exist?

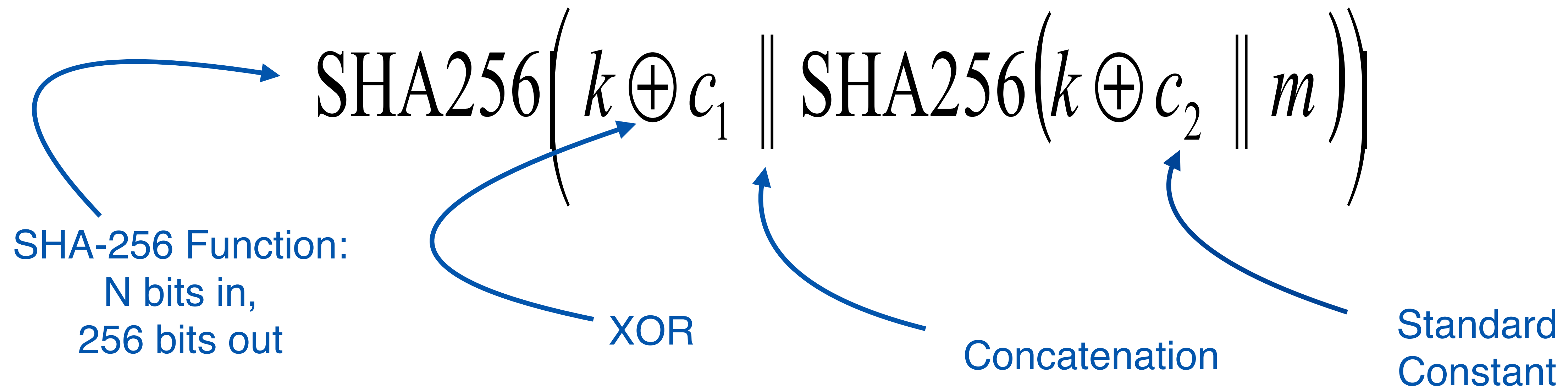
A: We don't know.

Best We Can Do: *Well studied functions where we haven't spotted a problem yet (e.g. HMAC-SHA256)*

Jargon

Message Authentication Code (MAC): *Effectively the same thing as a PRF*

Currently popular “PRF” (we hope!): *HMAC-SHA256*



Hash Functions, e.g., SHA-256

Input: Arbitrary Length Data

Output: Fixed-size digest (n bits)

No key, fixed function

Examples: MD5, ~~SHA-1~~, SHA-256, SHA-512, SHA-3

Good Hash Functions

Collision Resistance: Hard to find pair of input \mathbf{x} , \mathbf{x}' such that $H(\mathbf{x}) = H(\mathbf{x}')$

Preimage Resistance: Given y , hard to find \mathbf{x}' such that $H(\mathbf{x}') = y$

Second Preimage Resistance: Given \mathbf{x} , hard to find an \mathbf{x}' such that $H(\mathbf{x}) = H(\mathbf{x}')$

Other Hash Functions

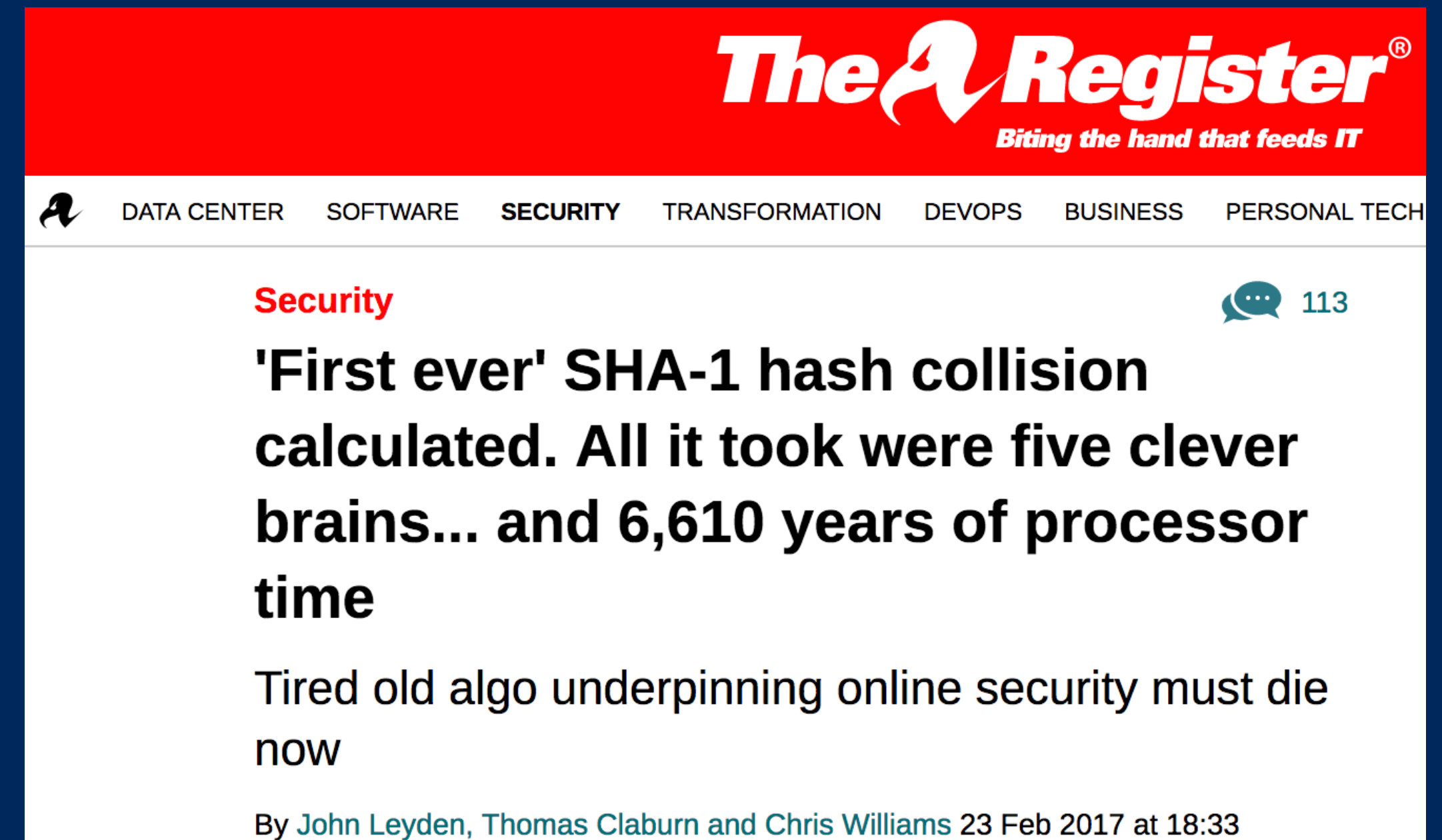
MD5

- Once ubiquitous
- Broken in 2004
- Now it's easy to find collisions (pairs of messages with the same MD5 hash)
- Exploited to attack real systems
- Project 1

(More) Other Hash Functions

SHA-1

- Fairly widely used
- Started to be unsupported in HTTPS in Jan 2016
- Turns out getting the Internet to change is hard
- Don't use in new applications!
- Might be project 1 in the future!



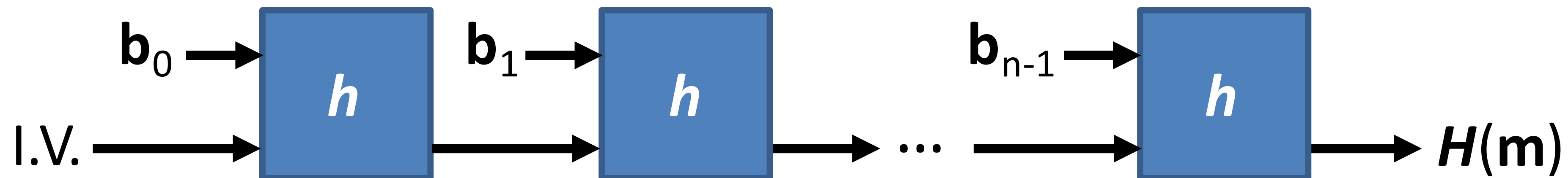
Constructing SHA-256

Input: Arbitrary Length Data

Output: 256-bit digest

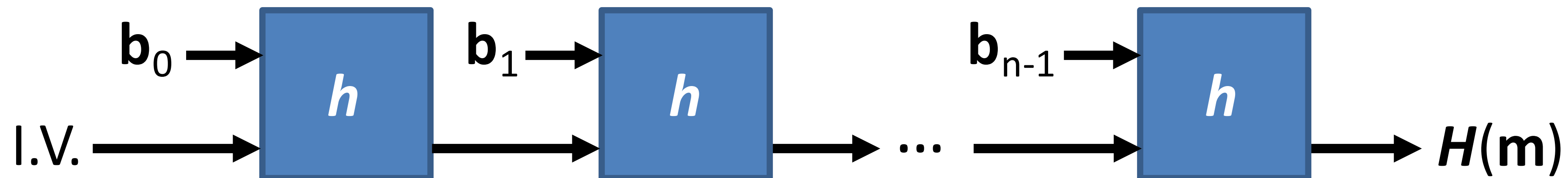
Built with “compression function” ***h*** using Merkle–Damgård construction

- (256 bits, 512 bits) -> 256 bits out
Designed to be “really hairy”, not going into details



SHA-256 Algorithm

1. Pad input ***m*** to multiple of 512 bits (using a fixed algorithm) *[why?]*
2. Split into 512-bit blocks ***b*₀**, ***b*₁**, ... ***b*_{n-1}**
3. ***y*₀** = constant initialization vector., ***y*₁** = ***h*(*y*₀, ***b*₀**)**, ..., ***y*_i** = ***h*(*y*_{i-1}, ***b*_{i-1}**)**
4. Return ***y*_n**



MAC / PRF / Hash / Oh My!

Message Authentication Code (MAC): *Effectively the same thing as a PRF*

Cryptographic Hash Function: *Not a strong PRF, but can be used to construct a MAC/PRF (e.g. HMAC). Example: SHA-256.*

Stand alone, a hash function is (usually) vulnerable to length extension attacks

- Given $\mathbf{z} = \mathbf{H}(\mathbf{m})$ for some unknown \mathbf{m} , calculate $\mathbf{H}(\mathbf{m} \parallel \text{padding} \parallel \mathbf{v})$ for attacker selected \mathbf{v} . You'll do this in Project 1.

So Far

The Security Mindset

What is Cryptography?

Message Integrity

Next Week...

Confidentiality