Security: An Overview of Cryptographic Techniques

15-640/440

With slides from: Debabrata Dash, Nick Feamster, Gregory Kesden, Vyas Sekar and others

Cryptography, Cryptographic Protocols and Key Distribution

- Authentication
- Mutual Authentication
- Private/Symmetric Keys
- Public Keys
- Key Distribution

What do we need for a secure communication channel?

- Authentication (Who am I talking to?)
- Confidentiality (Is my data hidden?)
- Integrity (Has my data been modified?)
- Availability (Can I reach the destination?)

What is cryptography?

"cryptography is about communication in the presence of adversaries."

- Ron Rivest

“cryptography is using math and other crazy tricks to approximate magic”

- Unknown TA

What is cryptography?

Tools to help us build secure communication channels that provide:

1) Authentication
2) Integrity
3) Confidentiality

Cryptography As a Tool

- Using cryptography securely is not simple
- Designing cryptographic schemes correctly is near impossible.

Today we want to give you an idea of what can be done with cryptography.

Take a security course if you think you may use it in the future

The Great Divide

	Symmetric Crypto (Private key) (E.g., AES)	Asymmetric Crypto (Public key) (E.g., RSA)
Shared secret between parties?	Yes	No
Speed of crypto operations	Fast	Slow

Symmetric Key: Confidentiality

<u>Motivating Example:</u>

You and a friend share a key K of L random bits, and want to secretly share message M also L bits long.

<u>Scheme:</u>

You send her the *xor(M,K)* and then she "decrypts" using *xor(M,K)* again.

1) Do you get the right message to your friend?

2) Can an adversary recover the message M?

3) Can adversary recover the key K?

Symmetric Key: Confidentiality

- One-time Pad (OTP) is secure but usually impactical
  - Key is as long at the message
  - Keys cannot be reused (why?)

In practice, two types of ciphers are used that require constant length keys:

Stream Ciphers:

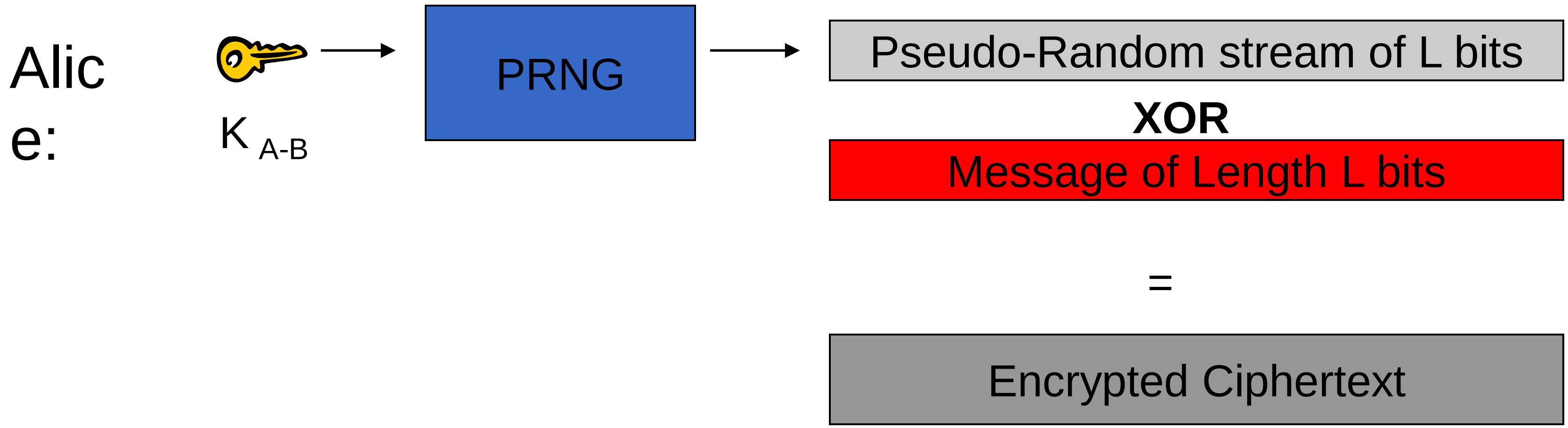
Ex: RC4, A5

Block Ciphers:

Ex: DES, AES, Blowfish

Symmetric Key: Confidentiality

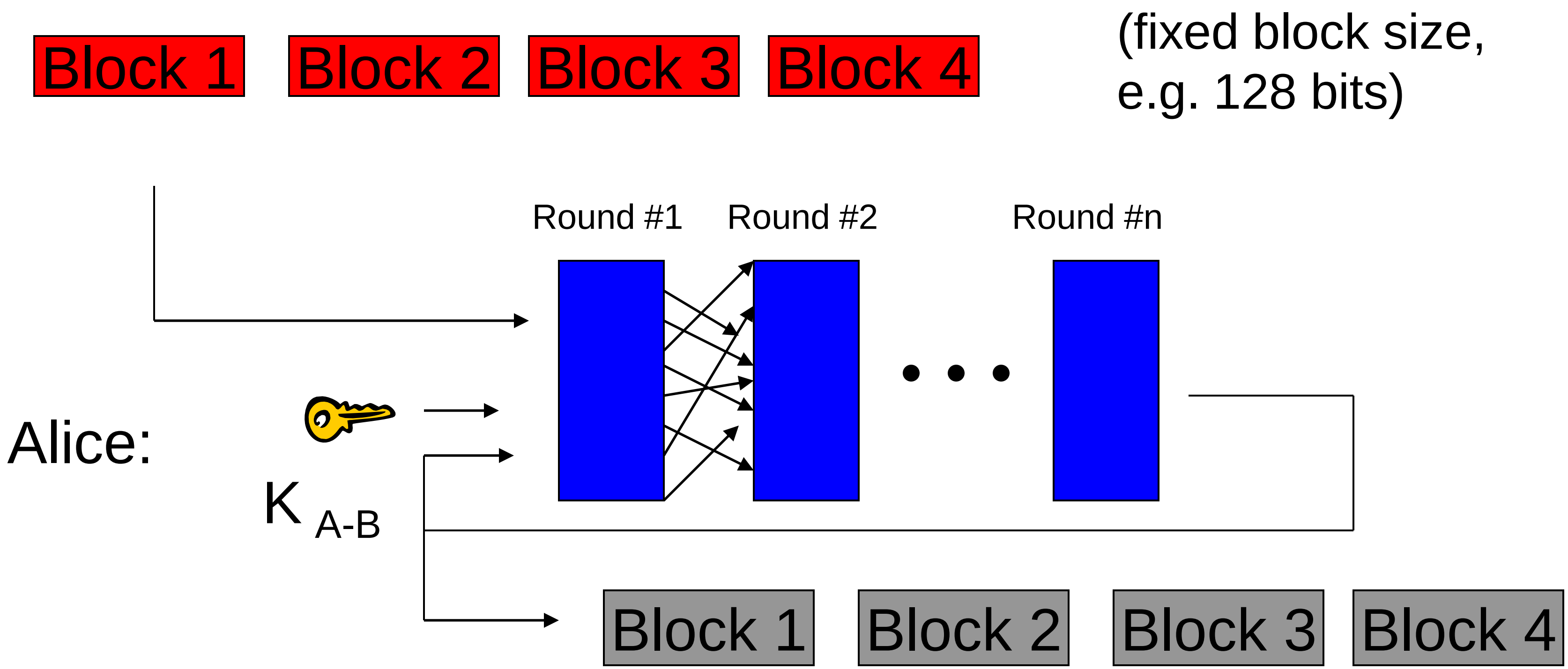
- Stream Ciphers (ex: RC4)



Bob uses K_{A-B} as PRNG seed, and XORs encrypted text to get the message back (just like OTP).

Symmetric Key: Confidentiality

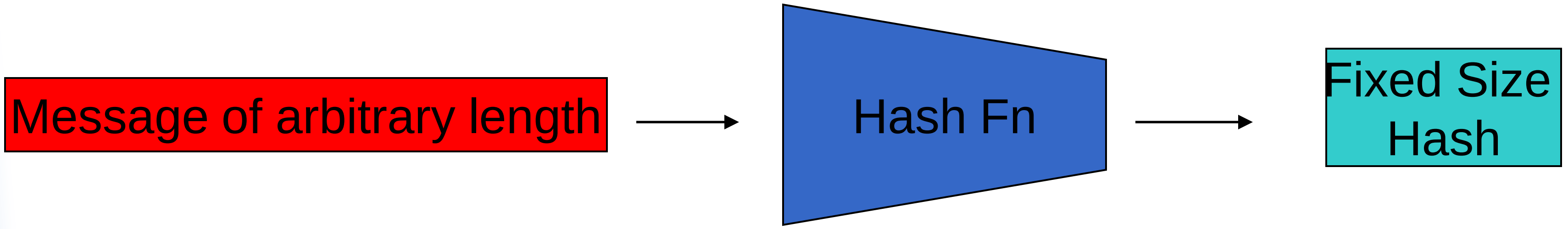
- Block Ciphers (ex: AES)



Bob breaks the ciphertext into blocks, feeds it through decryption engine using $K_{A\text{-}B}$ to recover the message.

Cryptographic Hash Functions

- Consistent
  - hash(X) always yields same result
- One-way
  - given Y, can't find X s.t. hash(X) = Y
- Collision resistant
  - given hash(W) = Z, can't find X such that hash(X) = Z

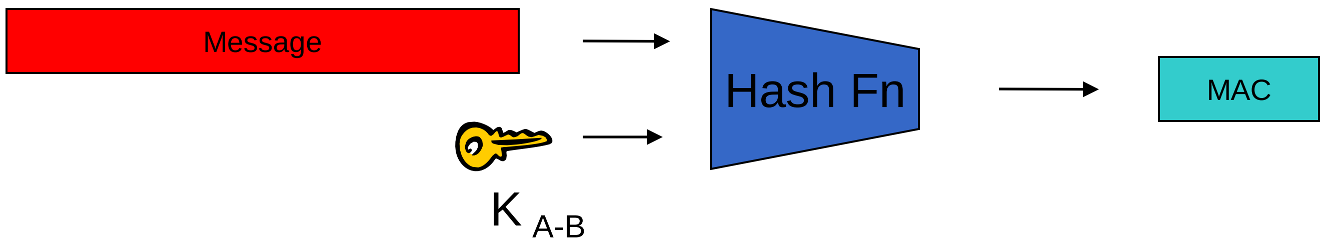


Symmetric Key: Integrity

- Hash Message Authentication Code (HMAC)

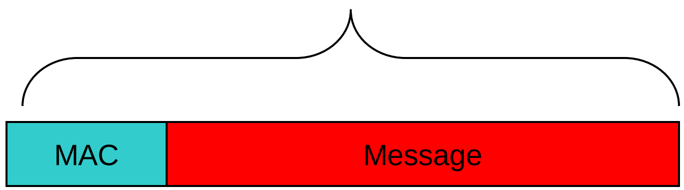
Step #1:

Alice creates MAC



Step #2

Alice Transmits Message & MAC



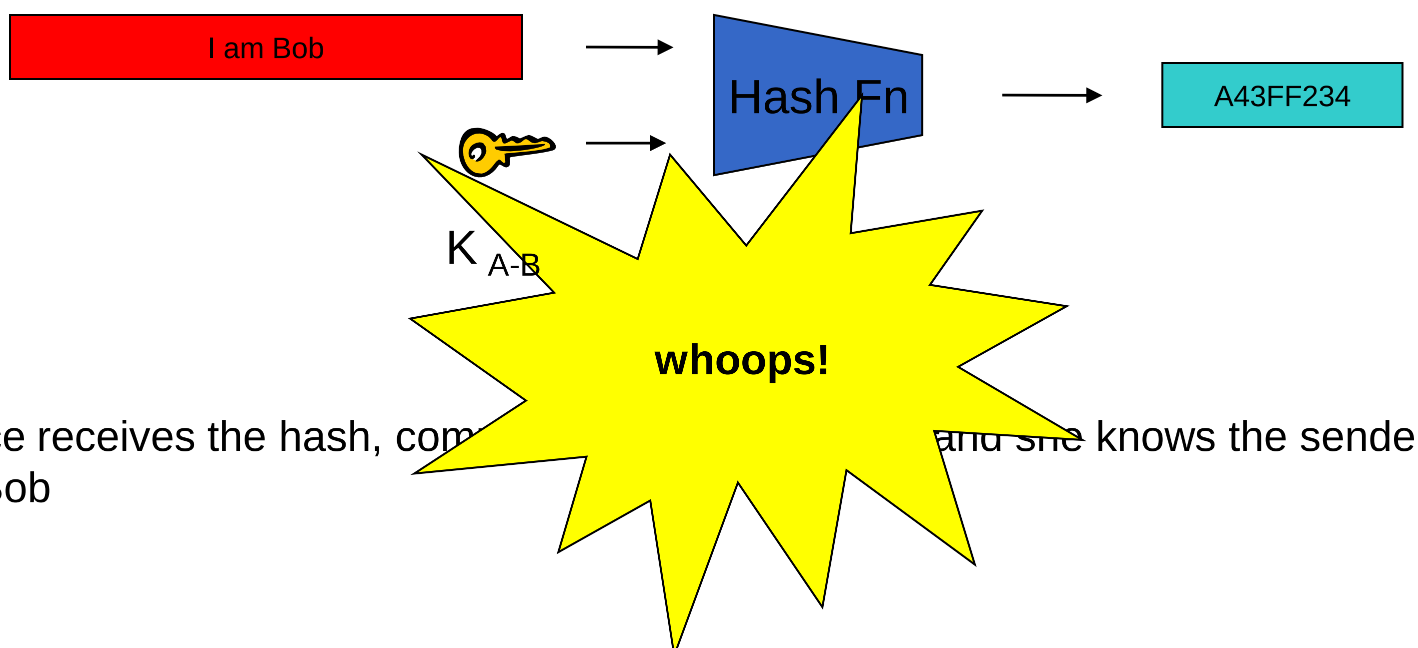
Step #3

Bob computes MAC with message and $K_{A\text{-}B}$ to verify.

Why is this secure?
How do properties of a hash function help us?

Symmetric Key: Authentication

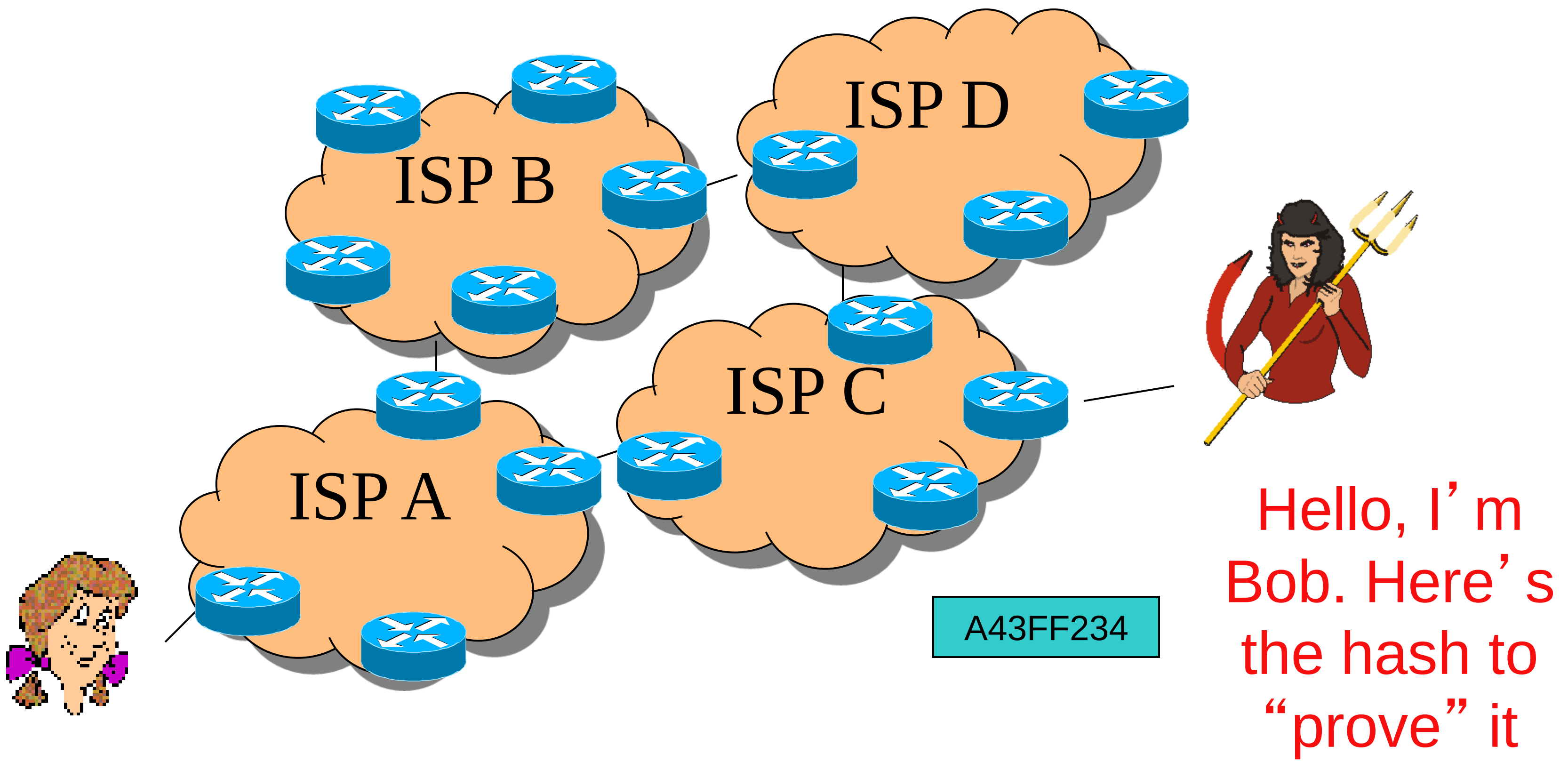
- You already know how to do this!
  (hint: think about how we showed integrity)



Alice receives the hash, com… and she knows the sender is Bob

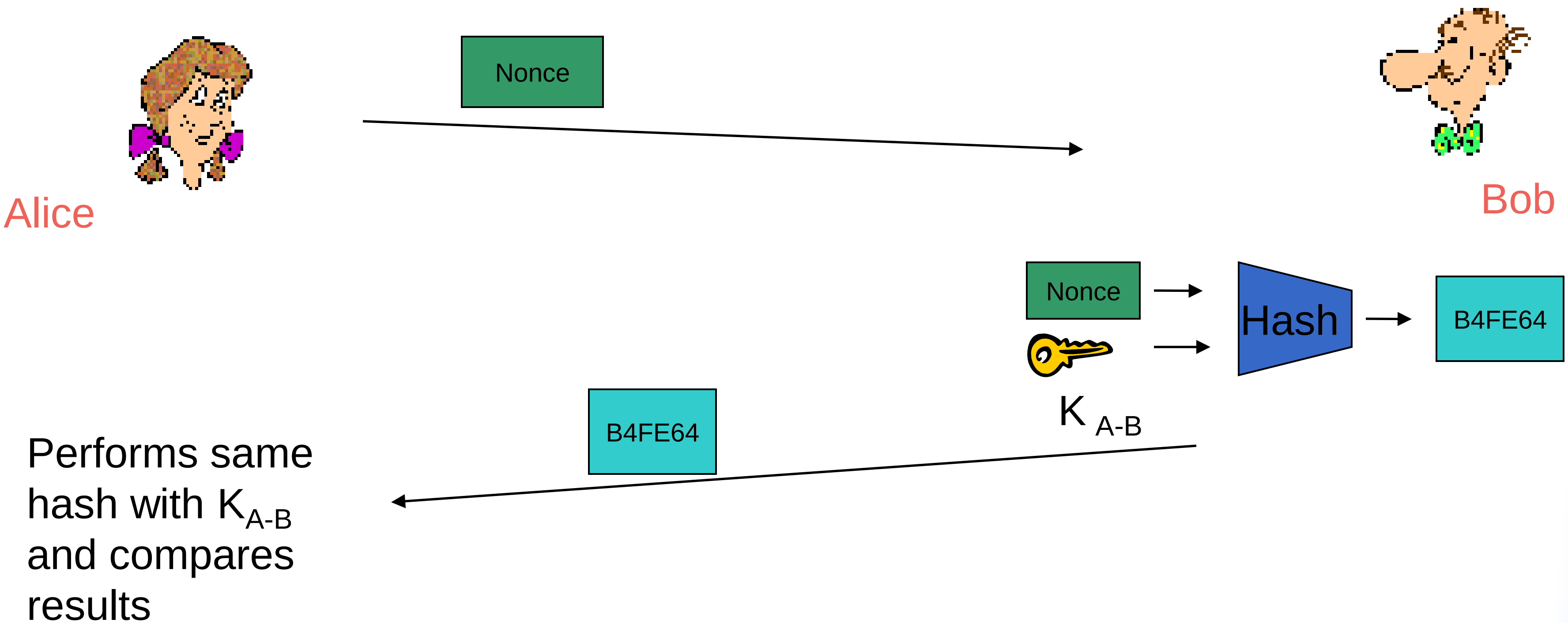
Symmetric Key: Authentication

What if Mallory overhears the hash sent by Bob, and then "replays" it later?



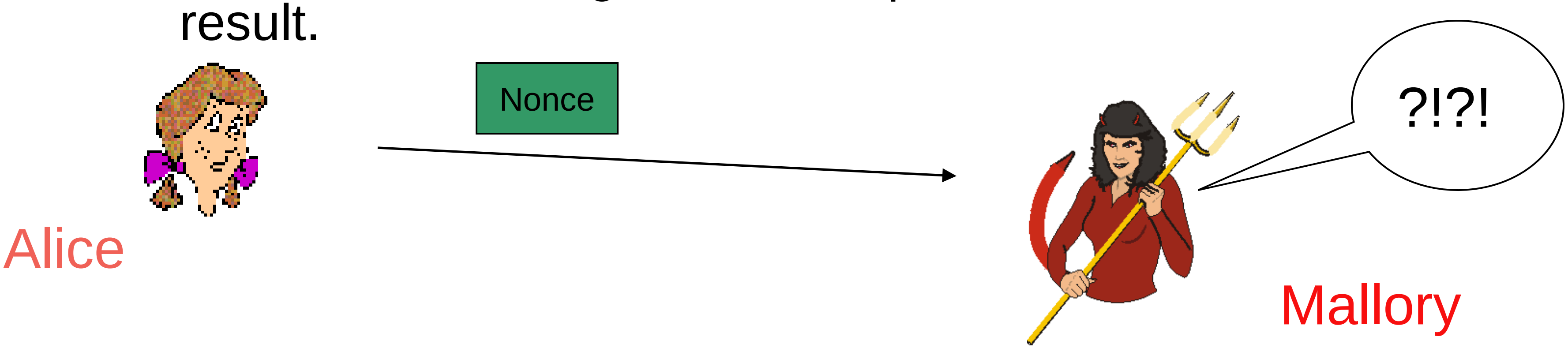
Symmetric Key: Authentication

- A "Nonce"
  - A random bitstring used only once. Alice sends nonce to Bob as a "challenge". Bob Replies with "fresh" MAC result.



Symmetric Key: Authentication

- A “Nonce”
  - A random bitstring used only once. Alice sends nonce to Bob as a “challenge”. Bob Replies with “fresh” MAC result.



If Alice sends Mallory a nonce, she cannot compute the corresponding MAC without $K_{A\text{-}B}$

Symmetric Key Crypto Review

- Confidentiality: Stream & Block Ciphers
- Integrity: HMAC
- Authentication: HMAC and Nonce

Questions??

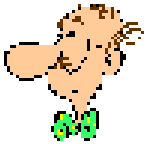
Are we done? Not Really:

1) Number of keys scales as $O(n^2)$

2) How to securely share keys in the first place?

Asymmetric Key Crypto:

- Instead of shared keys, each person has a "key pair"



 K_B Bob's <u>public</u> key

 K_B^{-1} Bob's <u>private</u> key

- The keys are inverses, so: $K_B^{-1}(K_B(m)) = m$

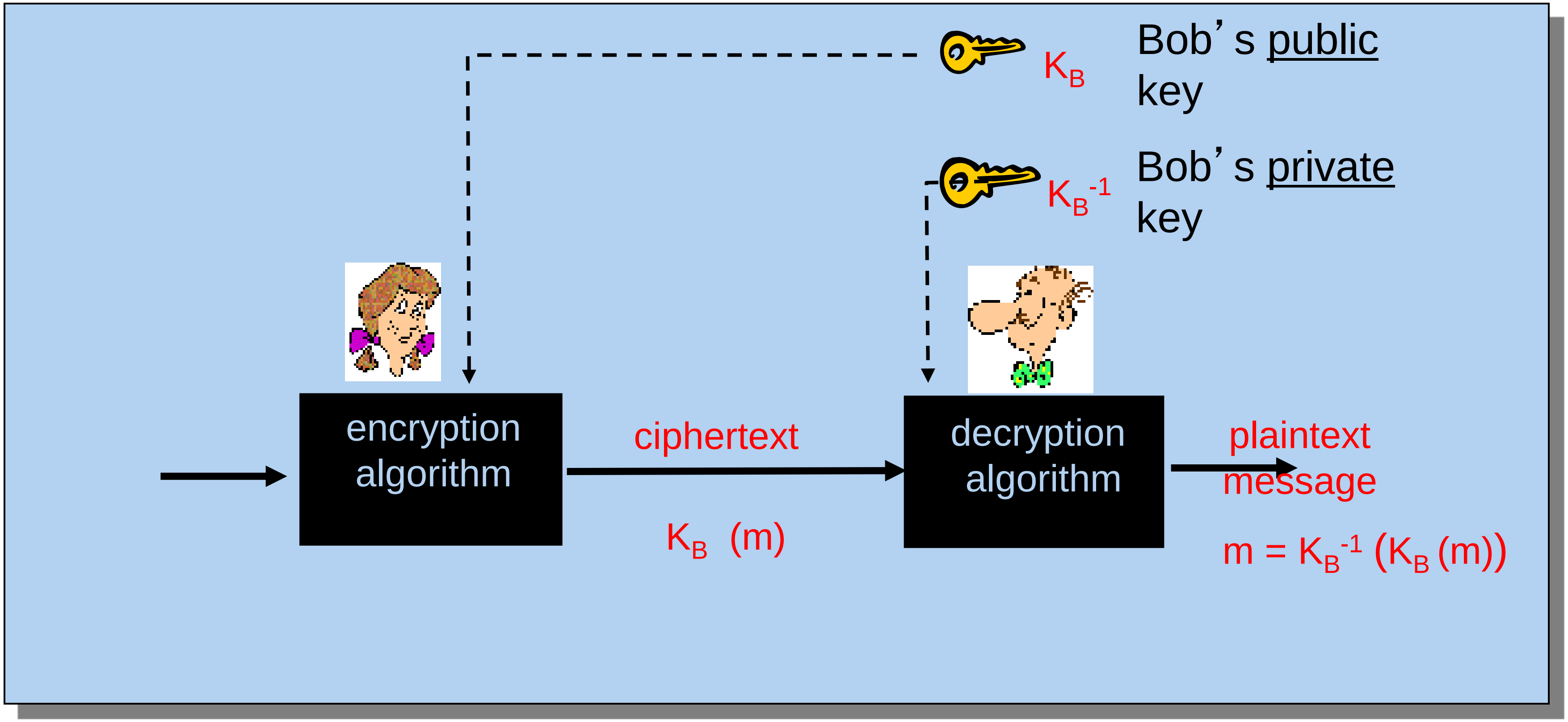
Asymmetric Key Crypto:

- It is believed to be computationally unfeasible to derive K_B^{-1} from K_B or to find any way to get M from $K_B(M)$ other than using K_B^{-1} .

=> K_B can safely be made public.

Note: We will not explain the computation that $K_B(m)$ entails, but rather treat these functions as black boxes with the desired properties.

Asymmetric Key: Confidentiality



Asymmetric Key: Sign & Verify

- If we are given a message M, and a value S such that $K_B(S) = M$, what can we conclude?

- The message must be from Bob, because it must be the case that $S = K_B^{-1}(M)$, and only Bob has K_B^{-1} !

- This gives us two primitives:
  - Sign (M) = $K_B^{-1}(M)$ = Signature S
  - Verify (S, M) = test($K_B(S)$ == M)

Asymmetric Key Review:

- Confidentiality: Encrypt with Public Key of Receiver
- Integrity: Sign message with private key of the sender
- Authentication: Entity being authenticated signs a nonce with private key, signature is then verified with the public key

But, these operations are computationally expensive*

Biometrics

- Nice in some respects
  - No need to distribute
  - Reducible to digital form
  - Unique in practice
- Hard to duplicate?
  - Used via binary representation
  - Warm gelatin fingers or slip-on finger-pads molded to prints?
  - Artificial eyeballs made to match scans?
  - Pictures? Videos w/blinking?
- Change over time?
  - Injury?
  - Aging?
- **Not replaceable or revocable**
  - What happens when “stolen?”
  - Are you “Deleted”?!?!?
  - (Well, you do have 10 fingers, two retinas, one nose, etc)

Multi-Factor, Human Factors

- Best systems use more than one factor
  - Something you know
  - Something piece of you
  - Biometrics + Password/Q&A Challenge, Etc
  - More natural factors better than fewer unnatural challenges
  - More weak factors may be stronger than fewer stronger factors

- Human factors are critical
  - Too many password restrictions? Too many passwords?
    - Write them down on Post-Its Notes!

Summary – Part II

- Symmetric (pre-shared key, fast) and asymmetric (key pairs, slow) primitives provide:
  - Confidentiality
  - Integrity
  - Authentication
- "Hybrid Encryption" leverages strengths of both.
- Great complexity exists in securely acquiring keys.
- Crypto is hard to get right, so use tools from others, don't design your own (e.g. TLS).