



# Computer Network Security

COMP 178 | Spring 2025 | University of the Pacific | Jeff Shafer

## Cryptography:

Key Exchange,  
Public Key Cryptography,  
Authentication

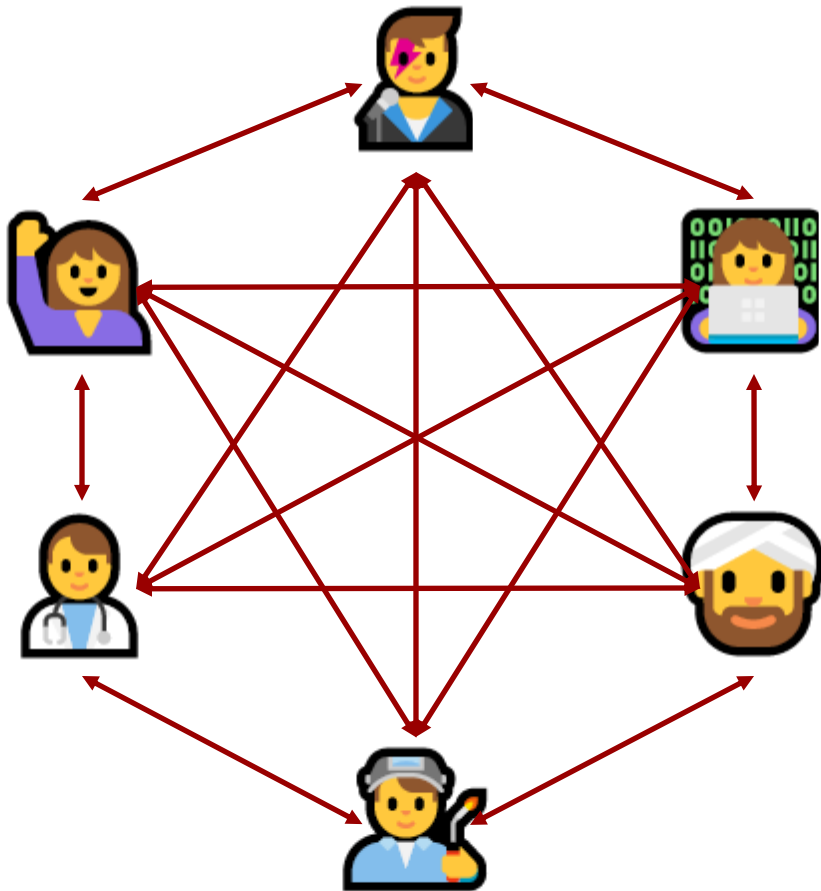
---



# Key Exchange



# Challenge – Exchanging Keys



$$\text{Exchanges} = \frac{n(n-1)}{2} = \frac{6(6-1)}{2} = 15$$

The more parties in communication, the more keys that need to be *securely* exchanged

Do we have to use out-of-band methods? (e.g., phone?)

# Key Exchange

➤ Insecure communications channel

➤ *Eve can see everything!*

➤ Alice and Bob agree on a **shared secret** (“key”) that Eve doesn’t know

➤ *Despite Eve seeing everything!*



(alice)



(bob)



(eve)

## New Directions in Cryptography

Invited Paper

Whitfield Diffie and Martin E. Hellman

**Abstract** Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.

### 1 INTRODUCTION

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The development of computer controlled communication networks promises effortless and inexpensive contact between people or computers on opposite sides of the world, replacing most mail and many excursions with telecommunications. For many applications these contacts must be made secure against both eavesdropping and the injection of illegitimate messages. At present, however, the solution of security problems lags well behind other areas of communications technology. Contemporary cryptography is unable to meet the requirements, in that its use would impose such severe inconveniences on the system users, as to eliminate many of the benefits of teleprocessing.

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from

communications over an insecure channel order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as a private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channel without compromising the security of the system. In *public key cryptosystem* enciphering and deciphering are governed by distinct keys,  $E$  and  $D$ , such that computing  $D$  from  $E$  is computationally infeasible (e.g., requiring  $10^{100}$  instructions). The enciphering key  $E$  can thus be publicly disclosed without compromising the deciphering key  $D$ . Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver is able to decipher it. As such, a public key cryptosystem is multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver public enciphering key and deciphers the messages he receives using his own secret deciphering key.

We propose some techniques for developing public key cryptosystems, but the problem is still largely open.

*Public key distribution systems* offer a different approach to eliminating the need for a secure key distribution channel. In such a system, two users who wish to exchange a key communicate back and forth until they arrive at a key in common. A third party eavesdropping on this exchange must find it computationally infeasible to compute the key from the information overheard. A possible solution to the public key distribution problem is given in Section III, and Merkle [1] has a partial solution of a different form.

A second problem, amenable to cryptographic solution which stands in the way of replacing contemporary business communications by teleprocessing systems is authentication. In current business, the validity of contracts guaranteed by signatures. A signed contract serves as gal evidence of an agreement which

Whitfield Diffie and Martin Hellman,  
“New directions in cryptography,”  
in *IEEE Transactions on Information  
Theory*, vol. 22, no. 6, Nov 1976.

Proposed public key cryptography.  
Diffie-Hellman key exchange.



Manuscript received June 3, 1976. This work was partially supported by the National Science Foundation under NSF Grant ENG 10173. Portions of this work were presented at the IEEE Information Theory Workshop, Lenox, MA, June 23-25, 1975 and the IEEE International Symposium on Information Theory in Ronneby, Sweden, June 21-24, 1976.

W. Diffie is with the Department of Electrical Engineering, Stanford University, Stanford, CA, and the Stanford Artificial Intelligence Laboratory, Stanford, CA 94305.

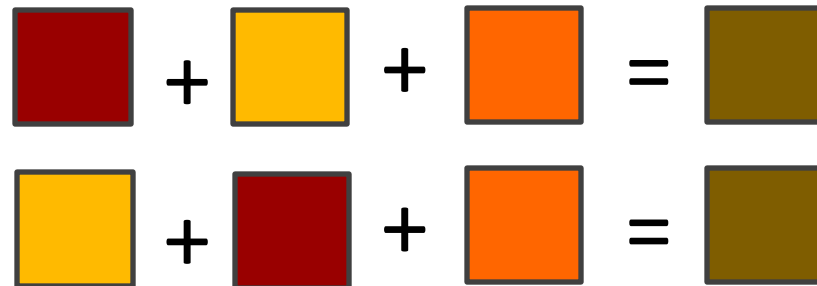
M. E. Hellman is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305.

# Diffie-Hellman *Color Analogy*

(1) It's easy to mix two colors:



(2) Mixing two or more colors in a *different order* results in the *same color*:



(3) Mixing colors is *one-way*

(Impossible to determine which colors went in to produce final result)

<https://www.crypto101.io/>

# Diffie-Hellman *Color Analogy*



(alice)



+



=



(eve)



(bob)






+



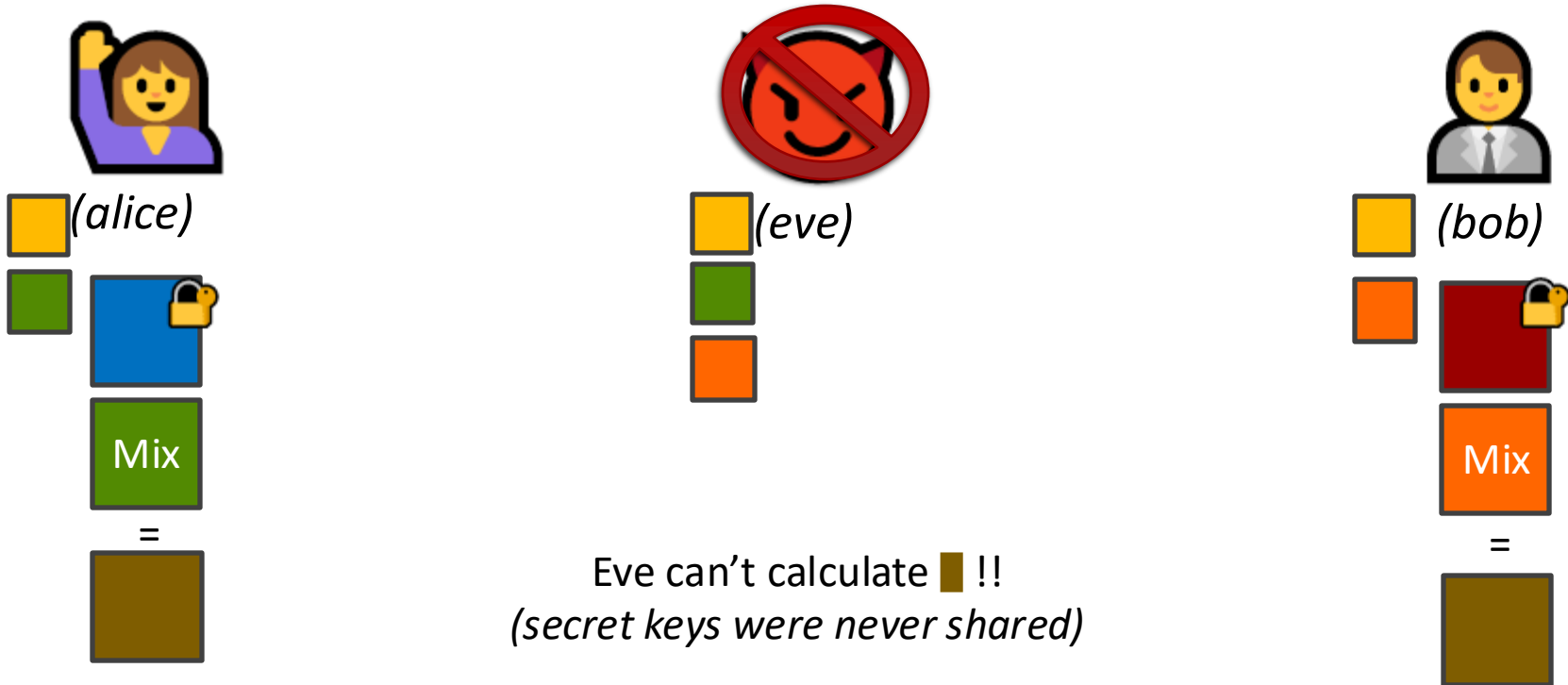
=



- (1) Start with public color  – share across network
- (2) Alice picks *secret* color  and mixes it to get 
- (3) Bob picks *secret* color  and mixes it to get 



# Diffie-Hellman *Color Analogy*



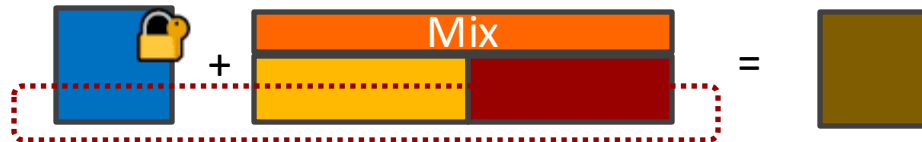
- (4) Alice and Bob exchange their mixed colors (green, orange)
- (5) Eve will see the mixed colors too (green, orange)
- (6) Alice adds her secret color (blue) to Bob's mix (orange) = (brown)
- (7) Bob adds his secret color (red) to Alice's mix (green) = (brown)



# Diffie-Hellman *Color Analogy*



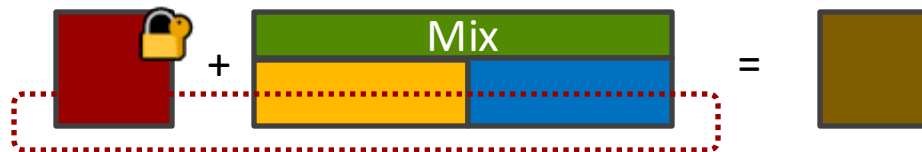
(alice)



(order doesn't matter)



(bob)



# Diffie-Hellman *Math*

$$y \equiv (g^x) \bmod (p)$$

$y$  is defined as equal to  $g^x$  modulo  $p$

$p$  = prime number (modulus)

$g$  = base integer

$x$  = random integer

**Assumption: Computing  $y$  is easy!**  
**But computing  $x$  given  $y$ ,  $g$ , and  $p$  is very hard!**  
***Discrete Logarithm Problem***

# Diffie-Hellman Math



(alice)



+



=



$$m_A \equiv (g^{r_A}) \bmod (p)$$

$$m_B \equiv (g^{r_B}) \bmod (p)$$



(bob)








+



=

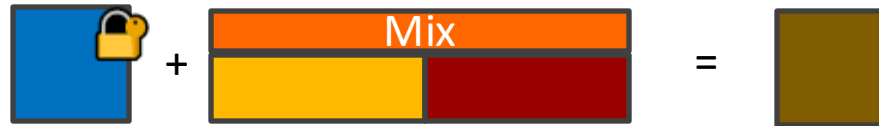


- (1) Public color  is a large prime number  $p$  and base  $g$
- (2) Alice *secret* color  is random integer  $r_A$
- (3) Bob *secret* color  is random integer  $r_B$
- (4) Alice mixed color  is  $m_A$
- (5) Bob mixed color  is  $m_B$
- (6) Exchange  $m_A$  and  $m_B$

# Diffie-Hellman *Math*



(alice)



$$s_A \equiv (m_B)^{r_A} \bmod (p)$$



(bob)



$$s_B \equiv (m_A)^{r_B} \bmod (p)$$

$$s_A = s_B$$

# Diffie-Hellman *Math*

- ↗ Doesn't have to be *modular division*
  - ↗ Could be *elliptic curves*
  - ↗ Could be *supersingular isogeny key exchange*
  - ↗ Could be *<other math words>...*

# Public Key Cryptography



# Public Key Cryptography

- **Asymmetric cryptography**
- Sending data to Alice?
  - Use her *public key*
- Alice receives your data?
  - She decrypts it with her *private key*



# A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

R.L. Rivest, A. Shamir, and L. Adleman\*

## Abstract

An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences:

1. Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key.
2. A message can be "signed" using a privately held decryption key. Anyone can verify this signature using the corresponding publicly revealed encryption key. Signatures cannot be forged, and a signer cannot later deny the validity of his signature. This has obvious applications in "electronic mail" and "electronic funds transfer" systems.

A message is encrypted by representing it as a number  $M$ , raising  $M$  to a publicly specified power  $e$ , and then taking the remainder when the result is divided by the publicly specified product,  $n$ , of two large secret prime numbers  $p$  and  $q$ . Decryption is similar; only a different, secret, power  $d$  is used, where  $e \cdot d \equiv 1 \pmod{(p-1) \cdot (q-1)}$ . The security of the system rests in part on the difficulty of factoring the published divisor,  $n$ .

**Key Words and Phrases:** digital signatures, public-key cryptosystems, privacy, authentication, security, factorization, prime number, electronic mail, message-passing, electronic funds transfer, cryptography.

CR Categories: 2.12, 3.15, 3.50, 3.81, 5.25

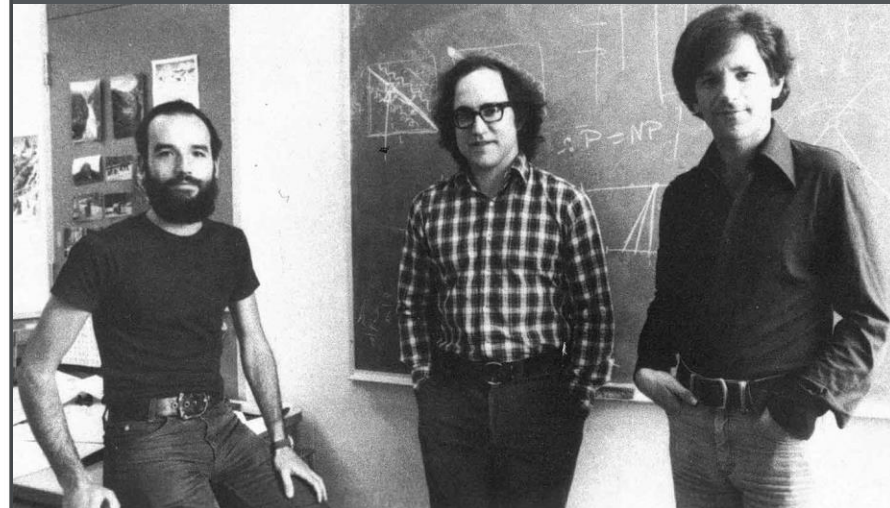
\*General permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to nonprofit libraries acting for them provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, or systematic or multiple reproduction.

This research was supported by National Science Foundation grant MCS76-14294, and the Office of Naval Research grant number N00014-67-A-0204-0063.

Author's Address: Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 E-mail addresses: rivest@theory.lcs.mit.edu

Ron L. Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* (February 1978)

**RSA encryption method**  
**First public key method**



# Public-Key Algorithms

## ➤ Key exchange algorithms

- Allows two parties to agree on a shared secret across an insecure medium
- Example: Diffie-Hellman

## ➤ Encryption algorithms

- Allows sender to encrypt *without* having to agree first on a shared secret
- Example: RSA

## ➤ Signature algorithms

- Allows sender to sign information using sender's *private* key and receiver to validate using sender's *public* key

# Public-Key Encryption

Public key encryption is *awesome!*

Should we use it everywhere?

# Caveat – Performance

- Rarely use public key encryption by itself
- Reasons:
  - **Size** (RSA can't encrypt anything larger than its modulus, i.e. 4096 bits)
  - **Performance**
    - RSA 2048 encryption: 0.08 **mega**cycles/operation (256B)
    - RSA 2048 decryption: 3.25 **mega**cycles/operation
    - AES-GCM: 2-4 cycles per byte
    - <https://www.cryptopp.com/benchmarks.html>

Hybrid cryptosystem – Use public-key algorithms to coordinate keys, and then use symmetric ciphers (shared key) for bulk operations



# Authenticated Encryption

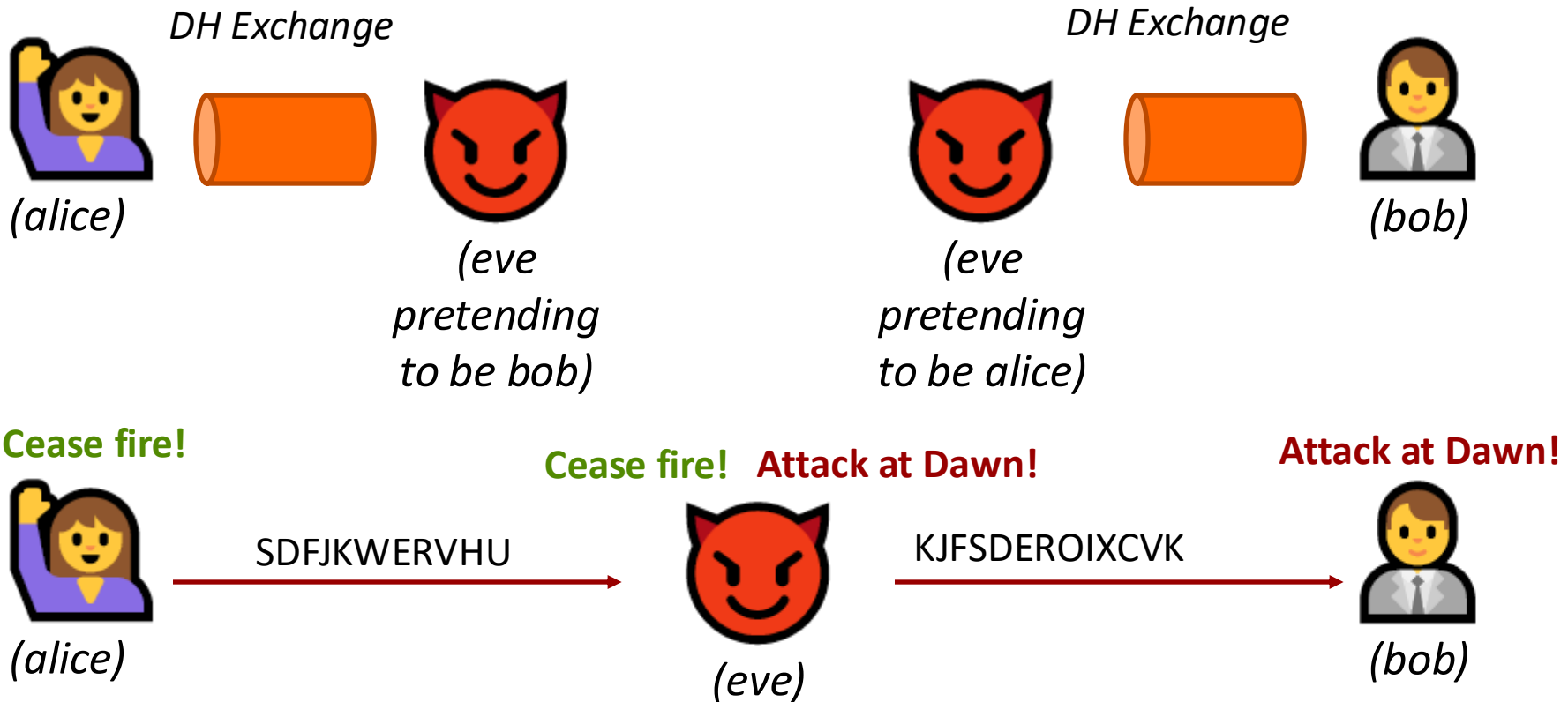


# Motivation

What if the attacker *actively manipulates* data instead of passively observing it?

# Motivation

How do we protect against this scenario?





# Warning!

*Encryption without authentication is almost certainly wrong...*

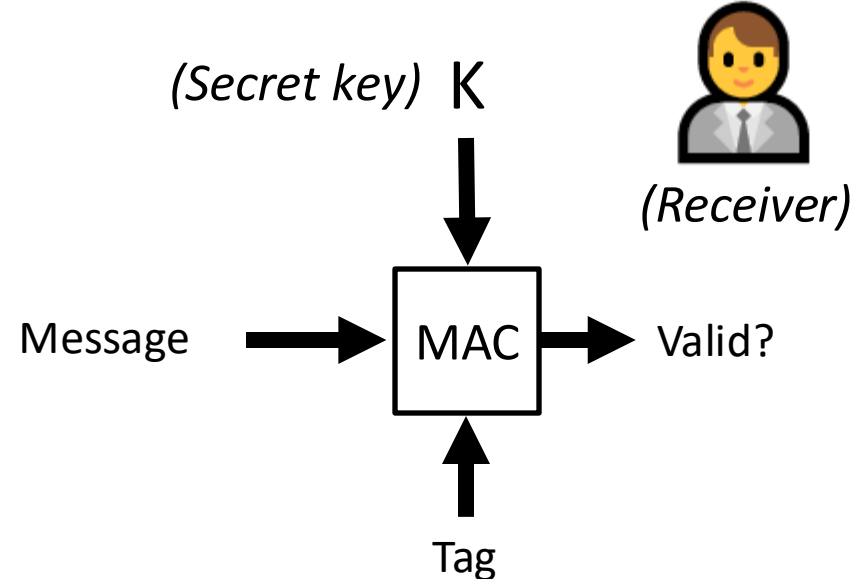
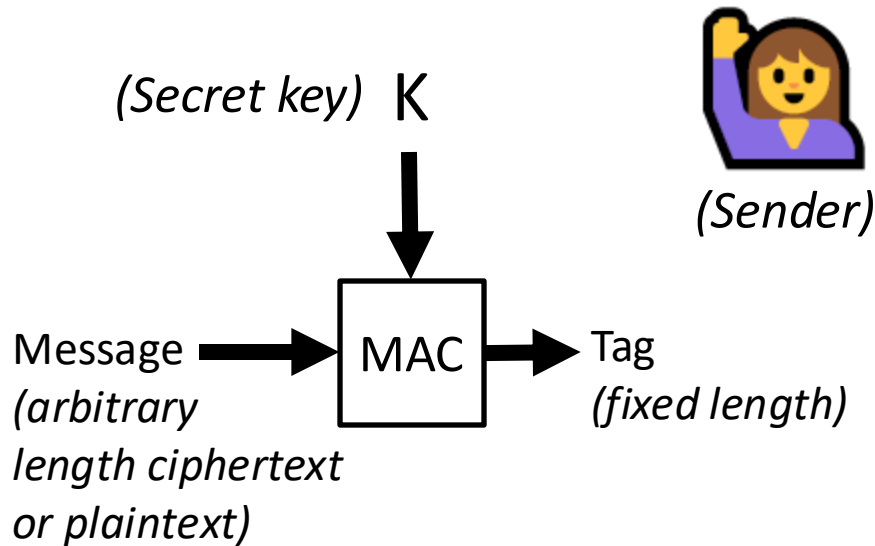
Attackers don't need to *decrypt* to *modify* ciphertext

# Authentication

- Goal: Add information to message that only the real sender (not Eve!) could have computed
- Authentication for symmetric-key encryption
  - **“Message Authentication Codes”**
  - MACs are generated and verified with the *same* key
- Authentication for public-key encryption
  - **“Signatures”**
  - Signatures are generated with *private* key and verified with *public* key

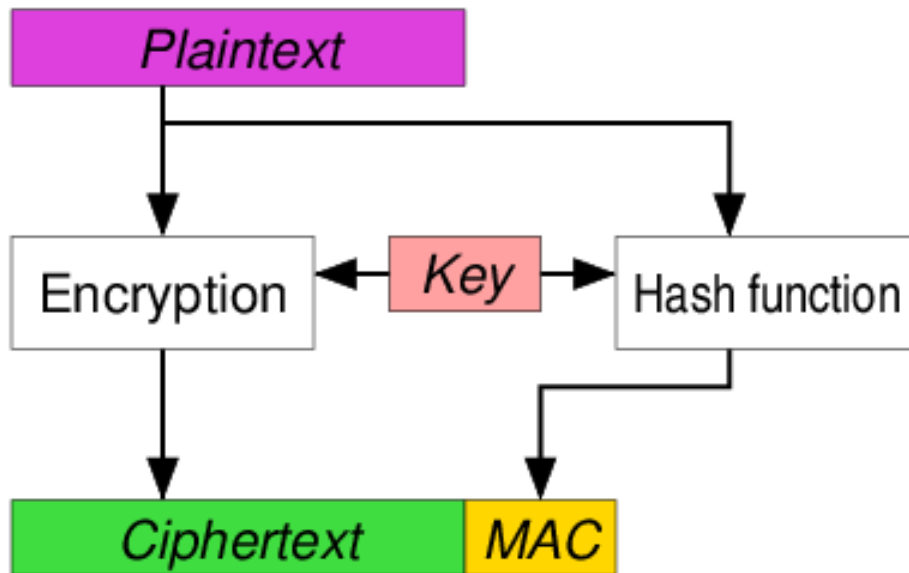
# Message Authentication Code (MAC)

- Small piece of information used to verify message integrity / authenticity ("Tag")
- Key is *shared secret* between Alice and Bob



# Message Authentication Code (MAC)

How to combine ciphertext with a MAC?



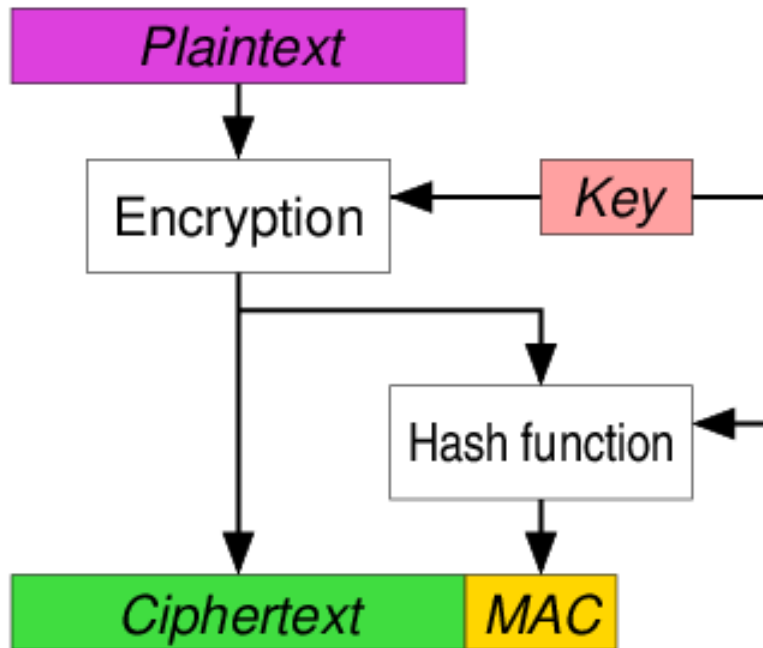
## ➤ Authenticate and Encrypt

- Used by SSH
- Authenticate and encrypt plaintext *separately*
- $C = E(K_C, P)$  and  $t = \text{MAC}(K_M, P)$
- Send C and t



# Message Authentication Code (MAC)

How to combine ciphertext with a MAC?



## ➤ Encrypt, then Authenticate

- Used by IPSec
- Standard ISO/IEC 19772:2009
- Encrypt plaintext, then authenticate ciphertext
- $C = E(K_C, P)$  then  $t = \text{MAC}(K_M, C)$
- Send C and t

# Message Authentication Code (MAC)

How to combine ciphertext with a MAC?

- **Which to choose?**
  - Authenticate and Encrypt
  - Authenticate, then Encrypt
  - Encrypt, then Authenticate – **Modern Best Practice**
- *Consider what the receiver does to reverse process*
- When you receive a message, the **very first thing** you do should be to authenticate it
  - Anything else risks **CERTAIN DOOM** (eventually)

<https://moxie.org/2011/12/13/the-cryptographic-doom-principle.html>



# AEAD

➤ We can do better still! What if authentication was *part of* our encryption scheme, and not a separate step?

➤ **Authenticated Encryption with Associated Data (AEAD)**

➤ Messages have two parts – example: emails

➤ Content (encrypt!)

➤ Metadata (authenticate, but plaintext)



# AEAD Modes

- Galois Counter Mode (**GCM**) – **Good!**
  - Not patent encumbered
  - SSH, TLS 1.2, OpenVPN
  - Standardized in ISO/IEC 19772:2009
  - Can be used by itself (authentication-only): **GMAC**
- Many other AEAD modes
  - EAX, OCB 2.0, CCM, Key Wrap, ...

# Modes of Operation

Remember our Block Cipher *Modes of Operation*?

## Encryption-Only No Authentication

- Counter (CTR) – **Best!**
- Cipher Block Chaining (CBC) – **Good**
- Electronic Code Book (ECB) – **Don't use!**
- Also ran: CFB, OFB, XTS, ...

## MACs – Message Integrity Only, No Encryption

- GMAC - **Good**
- HMAC – **Good**
  - *But why are you just authenticating and not encrypting?*
- Also ran: ALG1-6, CMAC

# Modes of Operation

Remember our Block Cipher *Modes of Operation*?

## Authenticated Encryption (Encrypt + Auth)

- GCM – **Good!**
- CCM – **Good!**
- Also-ran: EAX, OCB 2.0, Key Wrap, ...



# Repeating the Warning...

*Encryption without authentication is almost certainly **wrong**...*

Attackers don't need to *decrypt* to *modify* ciphertext

# Authentication

- Goal: Add information to message that only the real sender (not Eve!) could have computed
- Authentication for symmetric-key encryption
  - **“Message Authentication Codes”**
  - MACs are generated and verified with the *same* key
- Authentication for public-key encryption
  - **“Signatures”**
  - Signatures are generated with *private* key and verified with *public* key

# Signatures

- RSA-based signatures
- Digital Signal Algorithm (**DSA**)
- Elliptic Curve Digital Signature Algorithm (**ECSDA**)