

IT 4823

Information Security Administration

Cryptography II



Notice: This session is
being recorded.



Copyright © 2016 by Bob Brown



About Note Taking and Studying

- Read the assignments *before* class.
- In class, you should mostly be listening.
- Make short notes **on paper** of anything you think is important, especially if it's not on the slide.
- Merge your notes onto the slides the following day.
- Each class day, review the slides and notes from the class one week ago.

Three Cryptographic Technologies

- **Symmetric key cryptography:** The same key is used to encrypt and decrypt.
- **Public key cryptography:** Keys are used in pairs; one encrypts and the other decrypts.
- **Cryptographic hashes:** Generate a fixed length “fingerprint” that identifies a variable-length message.

A Very Strange Lock



- If locked with B, can only be unlocked with V. (B won't unlock it.)
- If locked with V, only unlocked with B.
- So, B and V are inverses of one another with respect to this lock.

Idea...

- Give each of your friends a lock and a B key
- You keep the *only* V key.
- Your friends can send you a locked box (locked with the B key) that only you can open because you have the only V key.



Public Key Cryptography

- What if we had two keys... one to encrypt and one to decrypt?
- This solves the key exchange problem because we don't have to exchange keys at all. (Or, we can use public key crypto to exchange symmetric keys; more on that later.)
- ... but it seems hard.

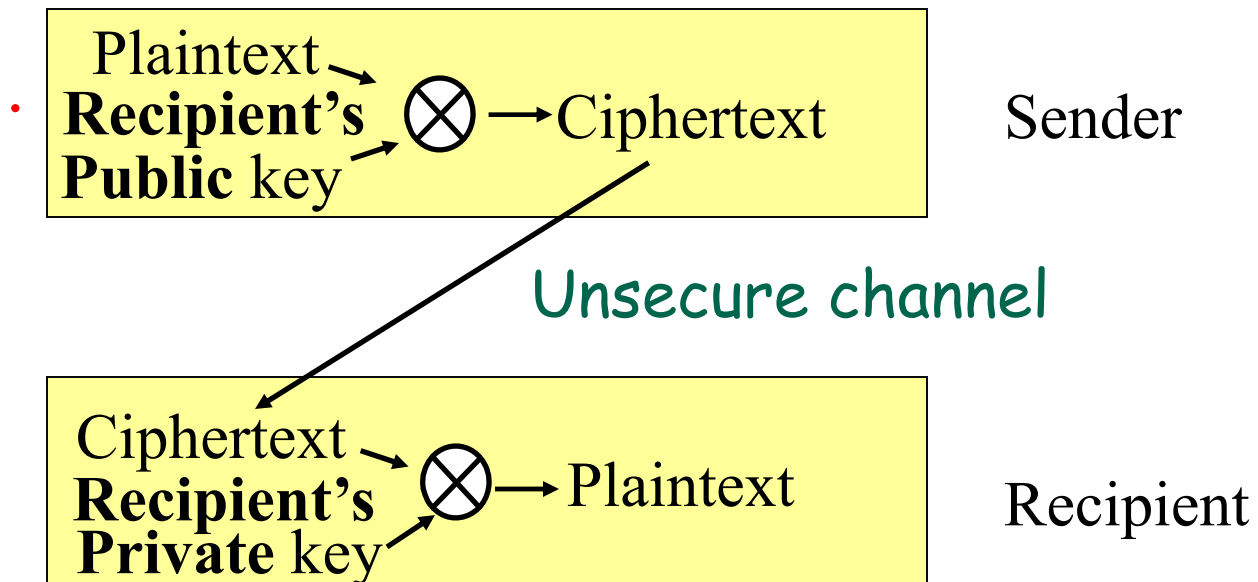
Public Key Cryptography

- Also known as “asymmetric key cryptography.”
- Encrypt with one key (publicly available)
- Decrypt with a different key, known only to the recipient.
- How? Trapdoor functions!
Functions that are easy to compute, but for which the inverse is intractable.
- 293×307 is easy. Find the prime factors of 89,851!
- Now try finding prime factors of a 100-digit number.

Public Key Cryptography

- With public key cryptography, one creates a pair of keys, k_v (private) and k_b (public).
- A ciphertext message, C is formed from a plaintext message M : $C = f(k_b, M)$
- Deriving M from C and k_b is intractable.
- Deriving k_v from k_b is intractable.
- However, $M = f(k_v, C)$ is straightforward to compute, but computationally intensive.

Public Key Cryptography



Note: both keys are *recipient's* keys; there is no need for the sender to have a key at all unless a reply is wanted.

The Math of Public Key Cryptography

Choose two large prime numbers, p and q .
Let $n = pq$; We will call n the “modulus.”
let $\phi = (p-1)(q-1)$

Choose e such that $1 < e < n$ and e and ϕ are relatively prime (have no common factors).
Call e the “public exponent.”

Compute d such that $ed \bmod \phi = 1$
Call d the “private exponent.”

The public key is: $k_b = (e, n)$

The private key is: $k_v = (d, n)$

The Math of Public Key Cryptography

Encryption: compute $c = m^e \bmod n$
 e and n are components of the public key.

Decryption: compute $m = c^d \bmod n$
 d and n are components of the private key.

$$c^d \bmod n = ((m^e) \bmod n)^d \bmod n = m^{ed} \bmod n$$

For $m < n$, $m^{ed} \bmod n = m$

Example

Choose $p = 5$, $q = 11$ (both prime)

$$n = pq = 55$$

$\phi = (p-1)(q-1) = 40$; let $e=9$ (relatively prime to ϕ)
 d will be 49 ($9 \times 49 = 441$; $441 \bmod 40 = 1$)

Let m , the message, be 6 (Must be less than n)

Encrypt:

$$c = 6^9 \bmod 55 = 10,077,696 \bmod 55 = 46$$

Decrypt:

$$m = 46^{49} \bmod 55 = 6 \text{ (} 46^{49} \text{ is an 82-digit number!)}$$

To break: Factor n to obtain p , q , then d

PK Crypto: Computationally Secure

- **To break:** Factor n to obtain p , q , then d
- Brute force attacks against public key crypto are ineffective because the keys are thousands of bits long.
- There are algorithms for finding the prime factors of large numbers.
- The Good Thing, and what makes public key crypto work is that finding prime factors of very large numbers takes a very long time indeed!

Computationally Secure? Maybe.

- A fast method for finding the prime factors of large numbers might be developed. (It is only “thought” to be hard.)
- An agency like the NSA may already have done so in secret.
- When quantum computing is sufficiently developed, it will be possible to use Shor’s algorithm to find prime factors of large numbers.

Equivalent Strength

- RSA, the company, says a 1024-bit public key is about as strong as an 80-bit symmetric key, *i.e.* not very strong, and 2048 bits of public key is about equivalent to 112 bits of symmetric key.
- NIST says a 15,360 bit public key is about as strong as a 256-bit symmetric key; thousands of years to crack.
- Elliptic curve cryptography seems to offer more secure asymmetric crypto with shorter keys.

Computational Effort

- Public key encryption can be 10,000 times more computational work than secret key encryption.
- Why? We are dealing with difficult arithmetic (exponentiation) on very large (hundreds of digits) numbers.

Public Keys with Less Computation

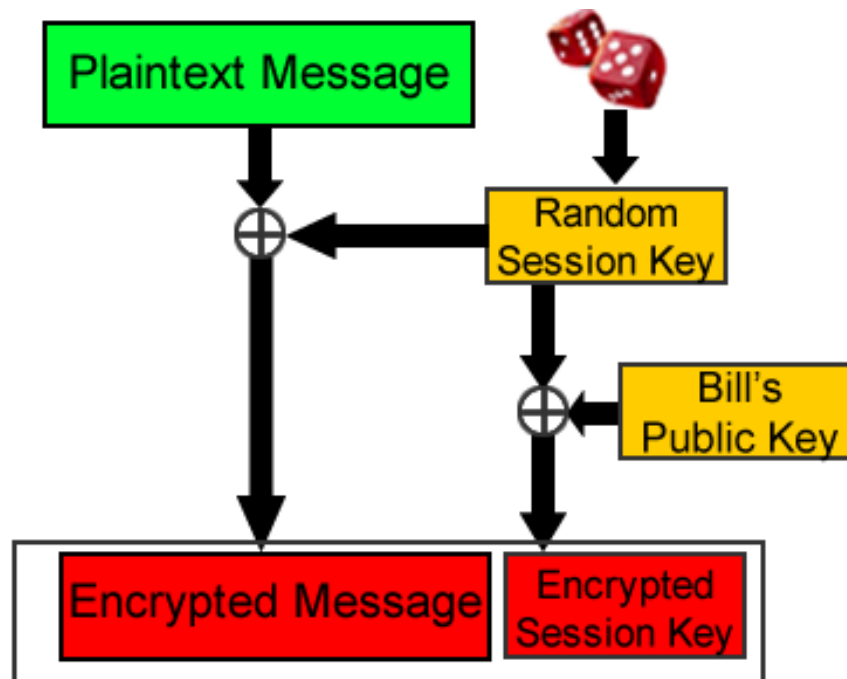
- Public key encryption is time-consuming... up to 10,000 times more work than secret key encryption.
- Solution: generate a random, one-time secret key, the *session key*.
- Encrypt the message with the secret key.
- Encrypt the secret key with the recipient's public key.
- Send encrypted message and encrypted key.

Key Exchange with Public Key Crypto

Alice wants to send a message m to Bill

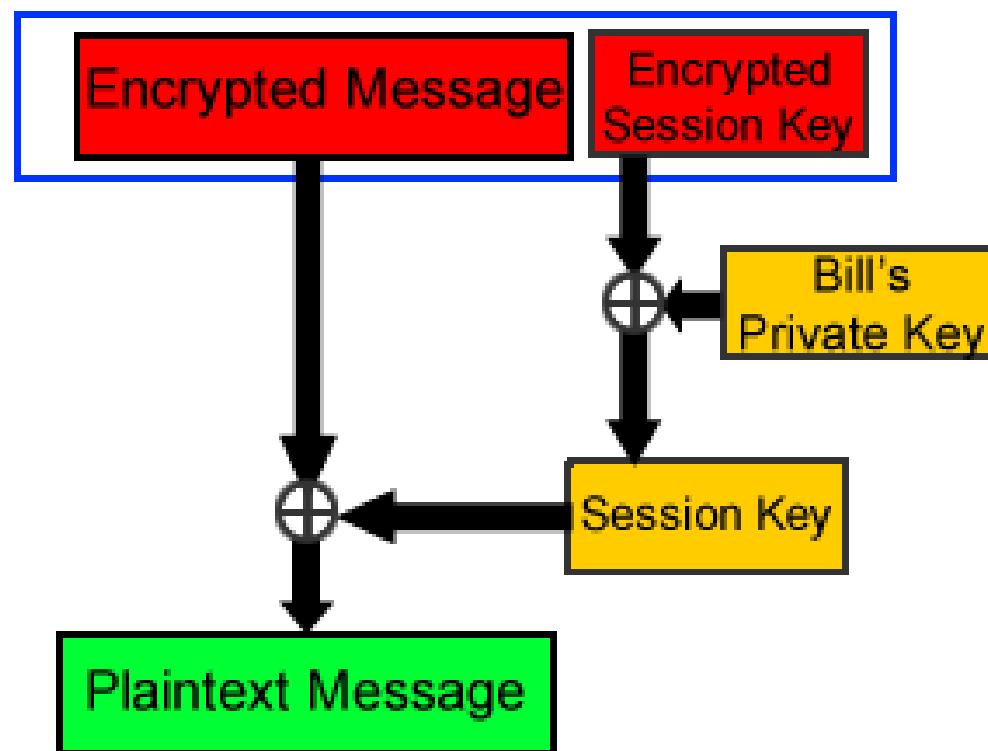
- Assume public key encryption
- Alice generates a random cryptographic key k_s and uses it to encipher m
 - k_s will be used for *this message only*
 - It's called a *session key*
- She enciphers k_s with Bill's public key k_B
- Alice sends $\{ m \} k_s // \{ k_s \} k_B$

Encryption with Session Key



The session key is a symmetric key; computation for encryption is (relatively) easy.

Decrypting a Hybrid Message



Principal Benefit

- Public key crypto is computationally expensive.
- Without session keys, one would have to encrypt the whole (possibly long) message with public key crypto.
- With session keys, only the (short) session key must be encrypted with public key crypto.
- The message can be encrypted with efficient symmetric key crypto.

Other Benefits

- Limits amount of traffic enciphered with single key. Standard practice, to decrease the amount of traffic an attacker can obtain
- Prevents some attacks: Example: Alice will send Bill message that is either “BUY” or “SELL”. Eve computes possible ciphertexts $\{ \text{“BUY”} \} k_B$ and $\{ \text{“SELL”} \} k_B$. Eve intercepts enciphered message, compares, and gets plaintext at once.
- Note: “real” PK encryption adds a random “padding” block to prevent this attack.

Digital Signatures

- Is there a way to use cryptography to verify origin integrity? (*Hint: Yes.*)
- The public and private keys of a key pair are cryptographic inverses of each other.
 - A message encrypted with the public key can only be decrypted with the corresponding private key.
 - *But*, a message encrypted with the private key can be decrypted using the corresponding *public* key!

Locking with the Private (V) Key

- If I lock a chest with the private (V) key...
- Anyone with a public (B) key can open it...
- But I am the only one who could have sent the contents because only I have the private (V) key. The contents are digitally signed!



Public Key Digital Signature

Public key cryptography works both ways; the keys are cryptographic inverses of one another!

Bob encrypts a message with his **private** key

Anyone can decrypt it using Bob's **public** key.

Only Bob could have encrypted it, so, it is “signed” by Bob. *Bob Brown*

Uhhh, but **anyone** can decrypt it; there is *no confidentiality*. (This is exactly analogous to a handwritten signature.)

Cryptographic Hash Functions

- **Idea:** Detect tampering with messages.
- **Needed:** A function that “characterizes” a message in some way. That’s a cryptographic hash function.
- **Characteristics of a strong hash function:**
 - $h(m)$ is easy to compute
 - Deriving m from $h(m)$ is infeasible
 - Given m , finding m' such that $h(m') = h(m)$ is infeasible (protects integrity by resisting collision attacks)

Simple Hash Example

DOG

hex 44 4f 47

Hash= $44+4f+47 = da$

BOG

hex 42 4f 47

Hash = $42+4f+47 = d8$

Different hash codes imply different messages.

Do identical hash codes mean that the messages are identical?

The MD5 and SHA Hashes

- The MD5 hash code is 128 bits; SHA is 160.
- So, there are 2^{128} or 2^{160} possible hashes (a lot!)
- The probability that two messages have the same hash code is very small.
- Can a message be especially crafted to have the same hash code as an arbitrary message?
(Not easily.)
- That is the strength of cryptographic hash algorithms.

Collision Attack

A **collision attack** is an attack that attempts to create a message with the same hash code as some other message.

A source program has a given MD5 hash.

Can you insert malicious code in the program, then change it in some other way such that it has the same MD5 hash as the “good” program?

<http://eprint.iacr.org/2004/199.pdf>

Hashing vs. Encryption

- Hash:
 - Not invertible (reversible)
 - Variable length input, fixed length output.
- Encryption:
 - Reversible; you can decrypt encrypted text.
 - One to one correspondence between plain text and ciphertext.
 - Large plain text will generate large ciphertext.

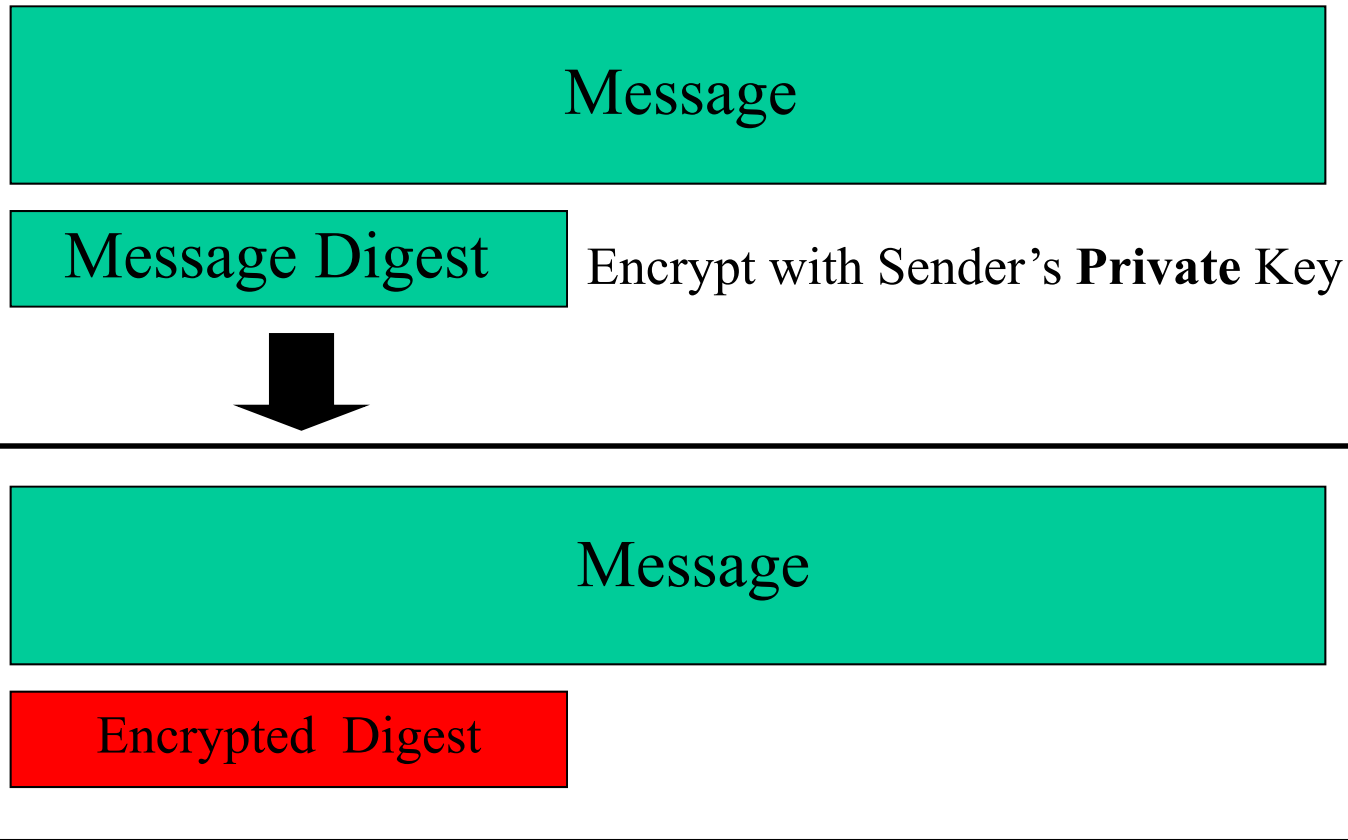
Digital Signature Improved

Bob computes a “digest” of a message using a hash function; this is smaller than the message and of fixed size, but has the property that changing the message changes the computed digest.

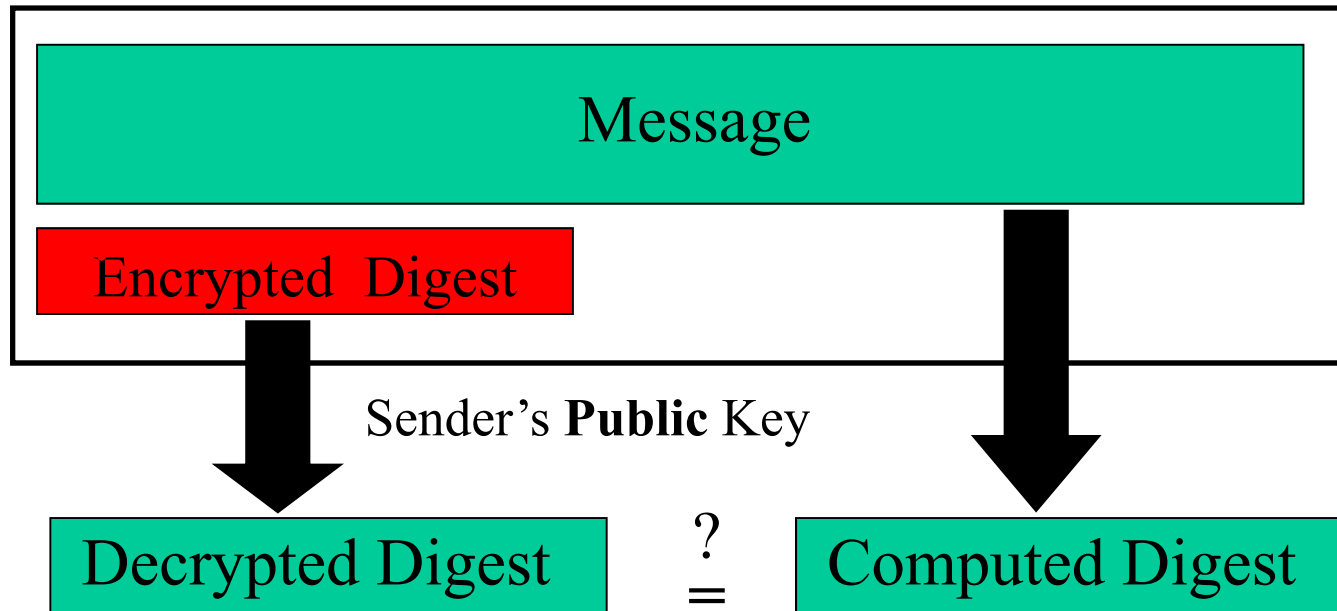
Bob encrypts the digest with his private key
Anyone can decrypt the digest using Bob’s public key, but only Bob could have encrypted it.
The message is digitally signed.

If the decrypted digest matches one computed from the message, the message has not been altered.

Digital Signature



Digital Signature



If the digests are equal, the sender is assured and no tampering has occurred.

Electronic Signatures and Law

- The E-Sign Law (2000) gives digital signatures the same legal weight as “wet ink” signatures.
- E-Sign does not specify a particular technology. (Good.)
- State law: UETA (Uniform Electronic Signature Transactions Act)
- “Electronic” signatures need not be cryptographic digital signatures.

Digital Signature and Confidentiality

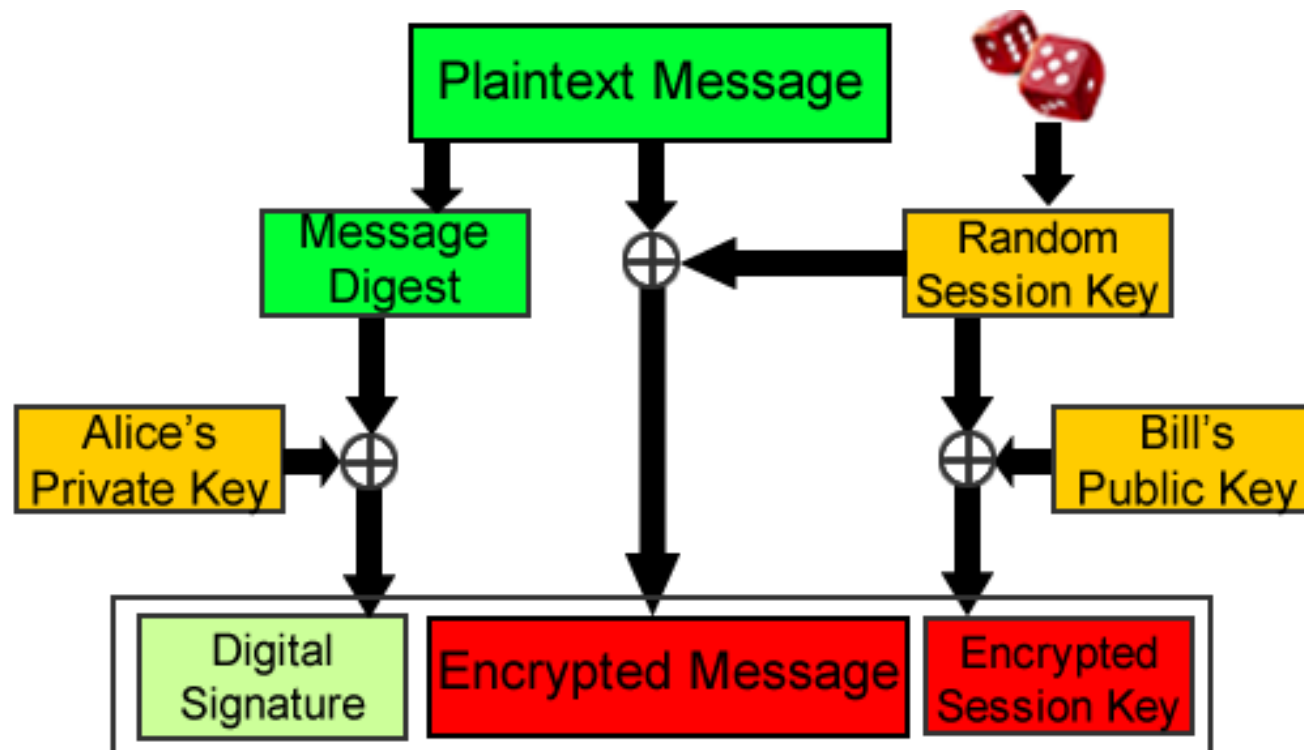
Alice computes a digest over a message and encrypts it with her **private** key.

Alice encrypts the message with a random session key and encrypts the session key with Bill's **public** key.

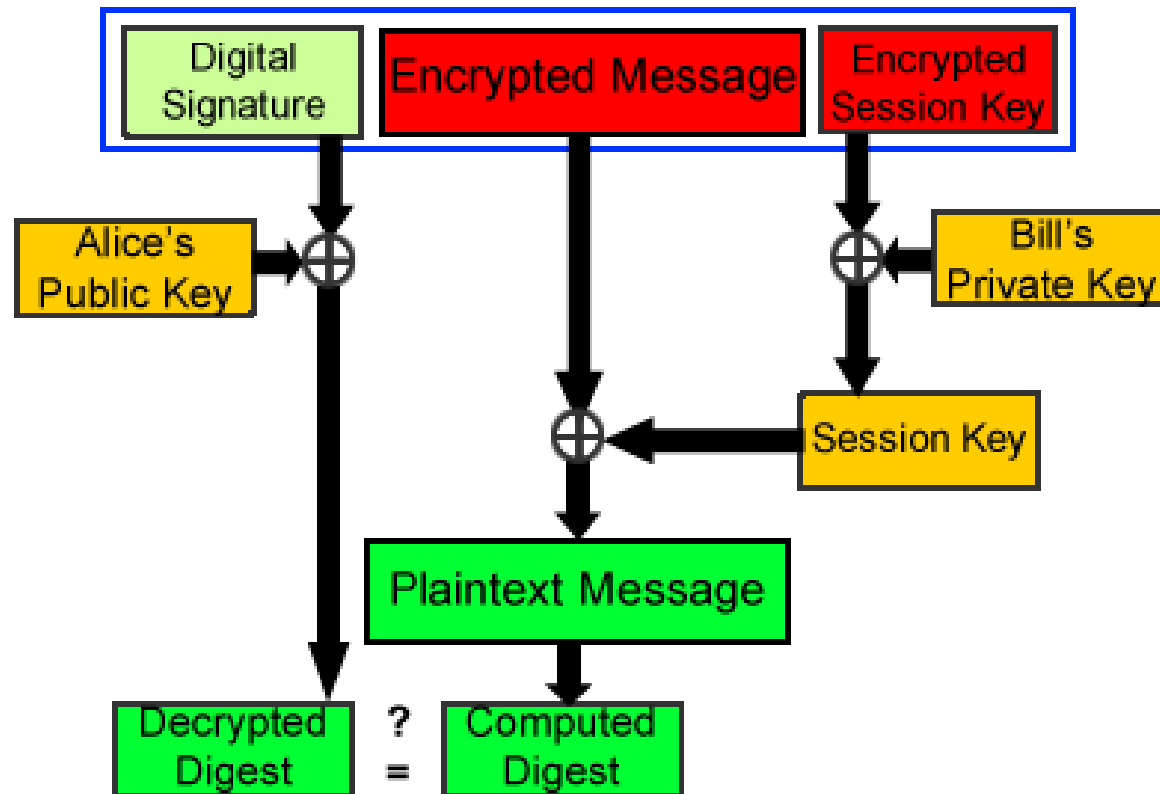
Only Bill can decrypt the message because his private key is needed to decrypt the session key. The message is confidential.

Bill decrypts and verifies the digest using Alice's public key. The message is signed.

Signature with Confidentiality



Decrypting the Message



Security Services

- Confidentiality
 - Only the owner of the private key knows it, so text enciphered with public key cannot be read by anyone except the owner of the private key.
- Integrity
 - Enciphered messages cannot be changed undetectably without knowing the sender's private key. (Data integrity.)
 - Message enciphered with private key came from someone who knew it. (Origin integrity and non-repudiation.)

Multiple Recipients

- This scheme can be adapted to allow a single message to be sent to multiple recipients.
- The session key must be encrypted separately, one time for each recipient, and identify the recipient.
- Each recipient uses his own private key to decrypt a copy of the session key.
- Only one copy of the message and message digest are needed.

How TLS Works

Generate a secret value at the browser

Encrypt the secret value with the server's public key to secure it; send to server.

Browser and server each derive a session key from the shared secret value

The session key is used for one communication session only.

When to Use Encryption

- Encryption is not a magic sauce.
- “Encryption is most useful when OS protections cannot work.” –*Steve Bellovin*.
 - Disk drives when attacker has physical access, backup media, etc.
 - “Cloud” storage
 - Data being transmitted over a network.
- Used inappropriately, encryption provides little protection and causes operational headaches.

Counter Example

Full-disk encryption of a database server:

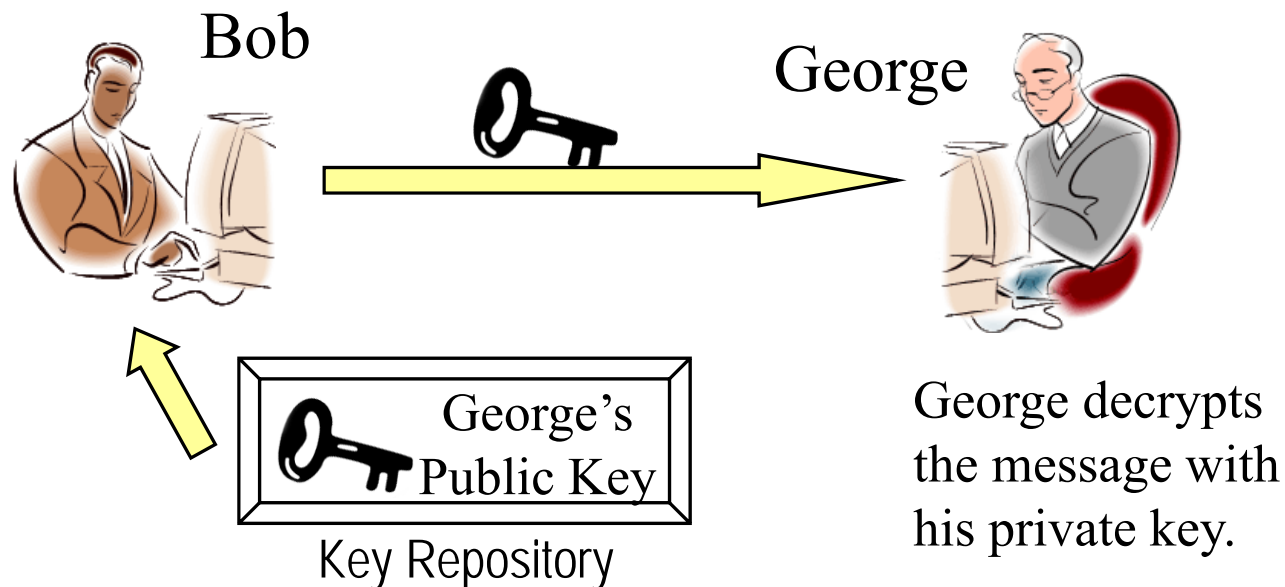
- Decryption key must be present for server to operate.
- Compromise of the database leaks plain-text data because the key is present.
- Server cannot restart automatically in case of a crash because key cannot be on server.
- *May* help when server/disks are retired, but disks should be sanitized anyway!

What Kind of Encryption

- Symmetric: When the subject that did the encryption will later decrypt and use the data. (No need for key exchange.)
- Asymmetric or hybrid: When two or more subjects are involved. (Use public key crypto for encryption or key exchange.)
- Cryptographic hash:
 - To check integrity of data or message
 - When stored value must be protected and compared for equality with challenge value, as with passwords.

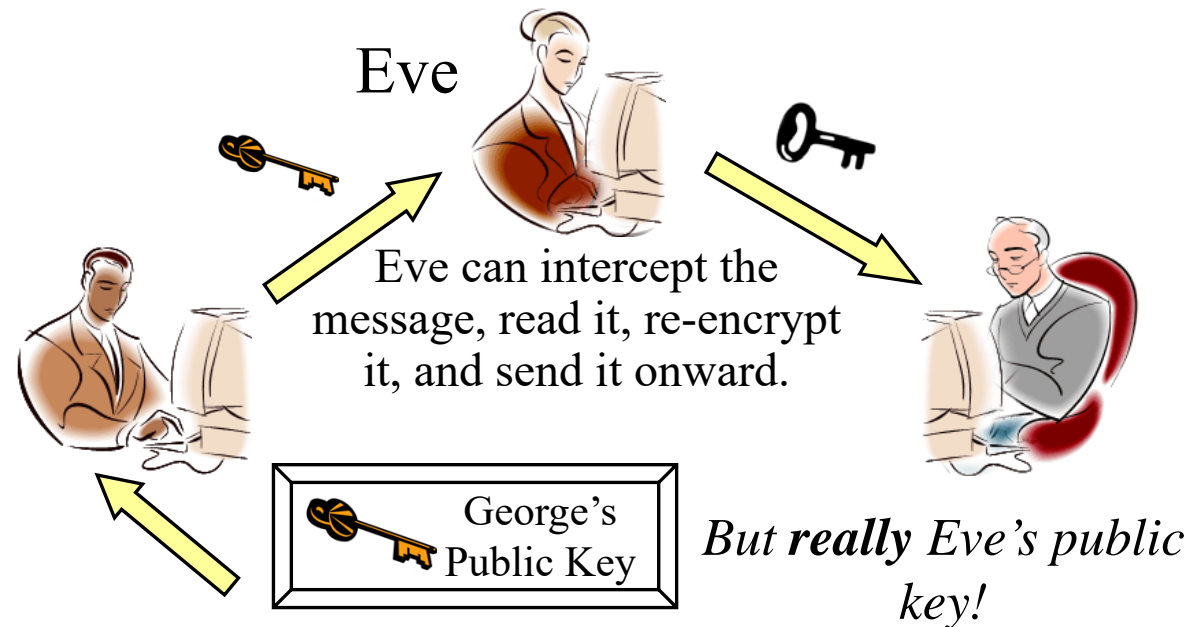
Public Key Infrastructure

Bob wants to communicate securely with George.
He gets George's public key from a key repository
and uses it to encrypt his message.



Man-in-the-Middle Attack

Evil Eve the Eavesdropper (and the man in the middle) sneakily replaces George's public key in the repository with her own.



Digital Certificates

- Goal: **bind identity to a public key**
- Principals must be identified by an acceptable name, like an email address or DNS name.

Certificates

- Create token (message) containing
 - Identity of principal (here, George)
 - Corresponding public key
 - Type of hash used
 - Other information (*e.g.* identity of signer)

Digitally signed by trusted authority

$$C_s = \text{George} \parallel k_{GB} \parallel T \parallel \text{Signer} \parallel \{\text{Digest}\} k_{CV}$$

The Digital Certificate

Contains:

- George's identity
- George's public key
- Identity of signer and other info
- Digital Signature

Everything is plain text except the digital signature, which is a cryptographic hash (digest) encrypted by the signer's *private* key.

George's Identity George@example.com
George's Public Key mQGiBD9aAvwRB
Hash, other information SHA-256
Signer's Identity Verisign
Digital Signature B157 ACE3 9788

Changing the Public Key

- Evil Eve substitutes her public key for George's, a man-in-the-middle attack.
- The digital signature is no longer valid (because the data have changed.)
- Evil Eve must gain access to the signer's private key (signing key) to forge a new digital signature. We hope this is hard!

George's Identity
George@example.com

Eve's Public Key
DfIZOkhURN

Hash, other information
SHA-256

Signer's Identity
Verisign

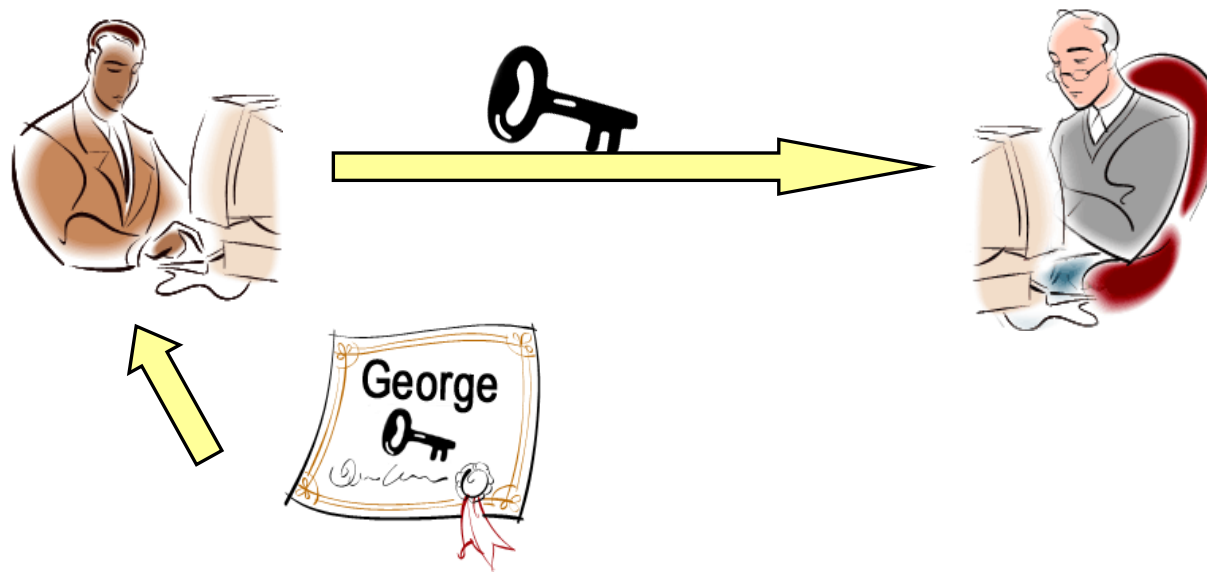
Digital Signature
B157 ACE3 9788

Validating a Digital Certificate

- Compute a cryptographic hash over all fields of the certificate except the digital signature.
- Decrypt the digital signature with the signer's public key, cached in the browser.
- Compare the digital signature – it's also a cryptographic hash – with the computed hash.
- If equal, the certificate is valid.
- Check the Certificate Revocation List (sometimes omitted!)

Public Key in a Certificate

If George's public key is contained in a digital certificate, signed by someone Bob trusts, it is *much harder* to tamper with the key.



Quantum Key Distribution

- QKD does not depend on public key cryptography to solve the key exchange problem.
- Instead, key information is encoded by polarizing individual photons.
- Intercepting the photons absorbs them; Eve cannot both copy the key and send it onward.
- So, the key cannot be intercepted, even on an unsecure channel.

Questions

