Introduction to Public/Private Key Infrastructure

- Encryption
- Digests
- Certificates

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Security Fundamentals

- Authentication
- Authorization
- Message Integrity
- Message Confidentiality

- This chapter covers integrity and confidentiality

Authentication is the act of determining the identity of a principal, usually via a secret of some sort (e.g. a password).

Authorization is the act of allowing or denying access to resources for authenticated principals.

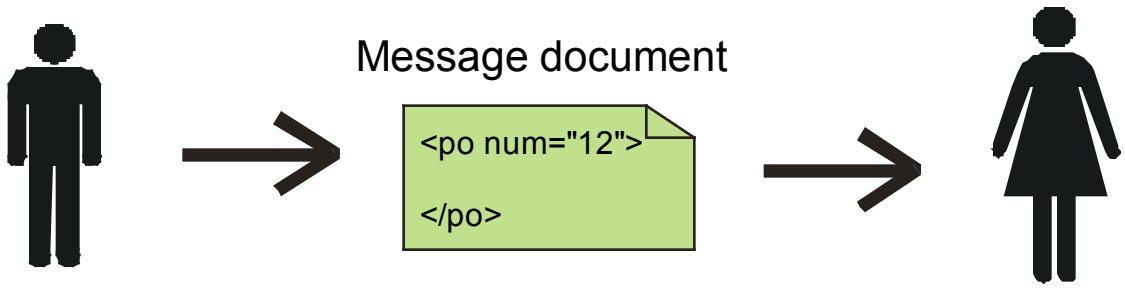
Message integrity refers to ensuring that a message has not been modified during transmission.

Message confidentiality refers to enciphering the message so it cannot be read except by the intended recipient.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Message Integrity

- We would like to ensure that messages have not been modified en-route, either intentionally or due to transmission errors



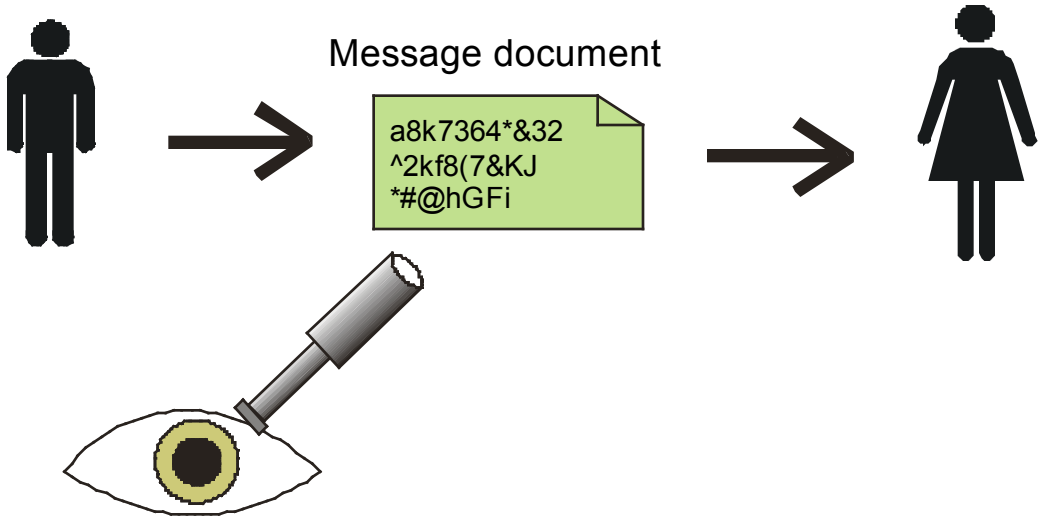
From the Wikipedia:

Integrity is the condition that exists when data is unchanged from its source and has not been accidentally or maliciously modified, altered, or destroyed

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Message Confidentiality

- We would perhaps like to ensure that if a message is intercepted en-route, the message will be indecipherable



2 - 4

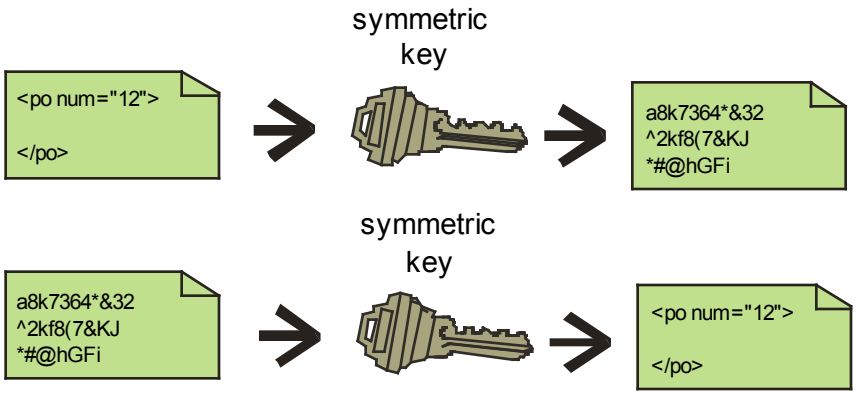
From the Wikipedia:

Confidentiality places restrictions on the accessibility and dissemination of information.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Encryption Fundamentals

- Encryption is the act of scrambling plain text so that it cannot be read
- There are several possible techniques, but two possibilities are popular in the Web service world:
  - **Symmetrical** encryption
  - **Asymmetrical** encryption using a public/private key infrastructure (PKI)



From the Wikipedia:

Encryption is a formula used to turn ordinary data, or "plaintext," into a secret code known as "ciphertext." Each algorithm uses a string of bits known as a "key" to perform the calculations. The larger the key (the more bits), the greater the number of potential patterns can be created, thus making it harder to break the code and de-scramble the contents.

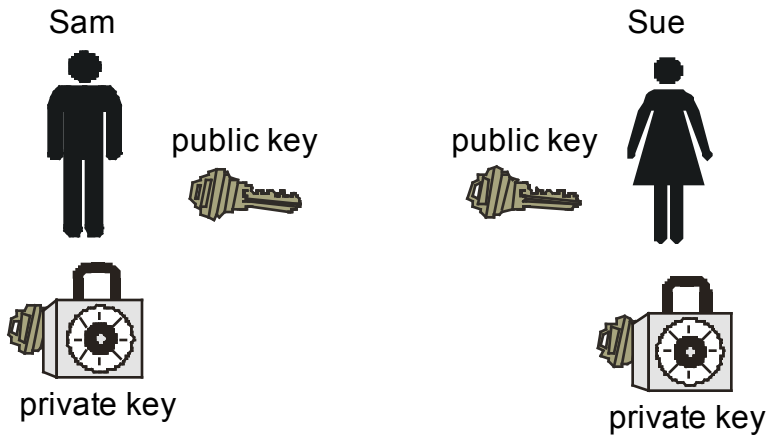
The figure on this page shows using a symmetric key, which must be in the possession of both the sender and the recipient. The downside of symmetric keys is that both parties must have the same key, which can be a problem in a Web-based system where the parties may not even know each other.

We will cover PKI coming up.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

PKI Fundamentals

- Public/private key encryption uses a pair of mathematically related numbers referred to as **keys**
- Clear-text that has been encrypted by one of the pair can only be decrypted by the other
- This technique relies on wide distribution of public keys and keeping private keys really private



2 - 6

From the Wikipedia:

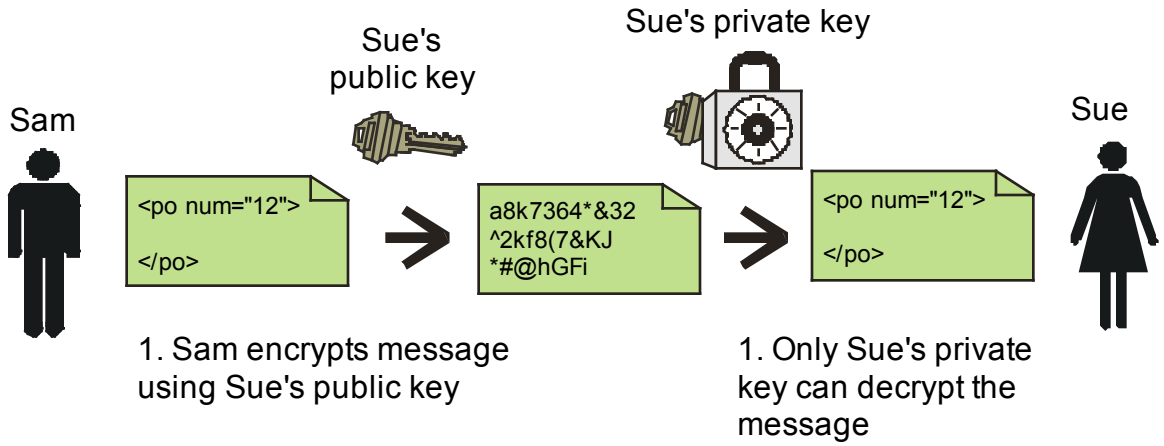
In cryptography, an asymmetric key algorithm uses a pair of cryptographic keys to encrypt and decrypt. The two keys are related mathematically; a message encrypted by the algorithm using one key can be decrypted by the same algorithm using the other. In a sense, one key "locks" a lock (encrypts); but a different key is required to unlock it (decrypt).

One drawback of asymmetric key algorithms is that they are much slower (factors of 1000+ are typical) than comparably secure symmetric key algorithms. In actual practice, both types are used, known as a hybrid cryptosystem. The receiver's public key encrypts a symmetric algorithm key, which is used to encrypt the main message. This combines the virtues of both algorithm types when properly done.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Using PKI for Confidentiality

- The sender signs the clear text with the recipient's public key
- Only the recipient can decrypt the cipher text



Sam must obtain Sue's public key and use it to encrypt the message -- only Sue's private key can decrypt it, so the message is safe from prying eyes.

Note that this only works if:

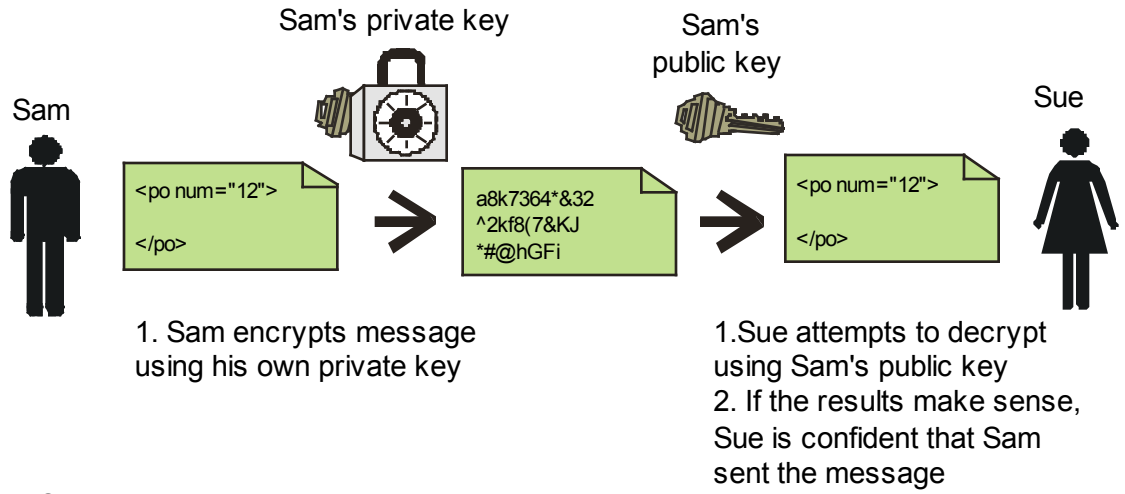
1. Sam is sure that Sue's public key really belongs to Sue.

2. Sue really keeps her private key private

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Using PKI for Authentication

- The sender signs the clear text with their own private key
- The recipient decrypts with the sender's public key -- if the results are intelligible, the recipient is confident of the sender's identity



Sue must obtain Sam's public key and use it to decrypt the message -- only Sam's private key can decrypt it, so if the result is not gibberish, then Sue can be pretty sure that Sam's private key encrypted the message.

Note that this only works if:

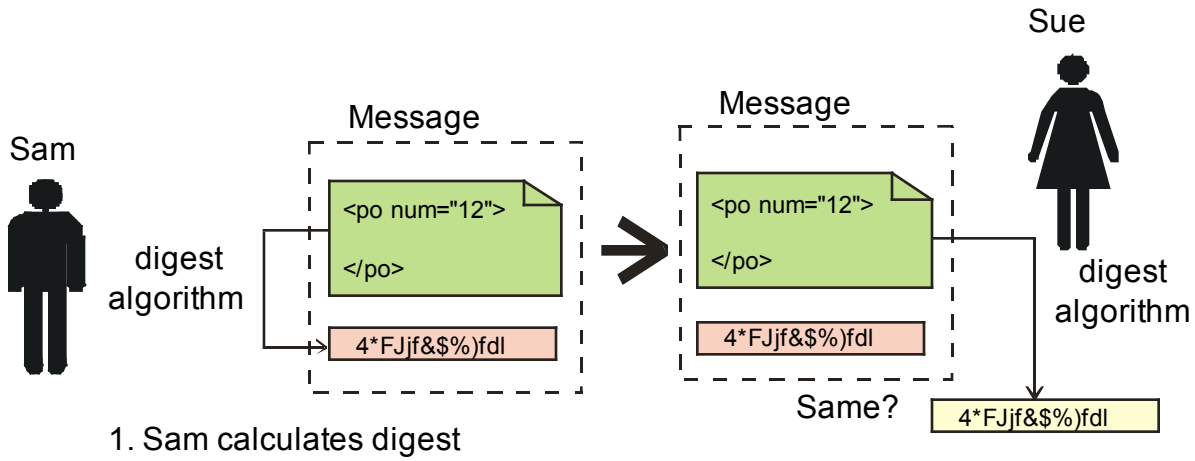
1. Sue is sure that Sam's public key really belongs to Sam.

2. Sam really keeps her private key private

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Message Digests

- You can use **message digests** to ensure **data integrity**



1. Sam calculates digest from clear text and includes digest with message

1. Sue calculate digest from clear text and compares it with sent digest
2. If they match, Sue is confident that message has not been altered during transit

From the Wikipedia:

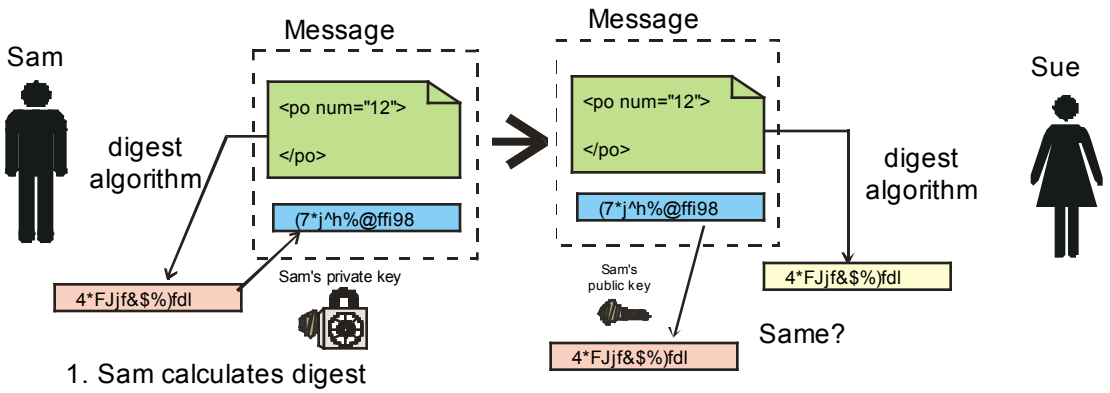
A digest is a condensed text string that has been distilled from the contents of a text message. Its value is derived using a one-way hash function.

There are several well-known digesting algorithms. Note that the message needs to specify which algorithm was used so that the recipient can re-create the digest.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Signed Message Digests

- Messages with signed digests ensure integrity **and** perform authentication (but not confidentiality)



1. Sam calculates digest from clear text
2. Sam encrypts (signs) digest with own private key and includes signed digest in message

1. Sue calculates digest from clear text
2. Sue uses Sam's public key to decrypt signed digest
3. Sue compares unsigned digest with calculated one
4. If they match, Sue is confident that the message came from Sam and that it has not been altered

Using signed message digests gives you integrity and authentication without needing to encrypt the entire message body, which could be large (remember that asymmetric encryption and decryption is computationally intensive).

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Digital Certificates

- A certificate is a digitally signed document that authenticates a public key
- The certificate contains a public key and other information -- the entire certificate is digested and the digest signed with the private key of a trusted, well-known **certificate authority**



From the Wikipedia:

A digital certificate is the digital equivalent of an ID card used in conjunction with a public key encryption system. Also called "digital IDs," digital certificates are issued by a trusted third party known as a "certification authority" (CA) such as VeriSign (www.verisign.com) and Thawte (www.thawte.com). The CA verifies that a public key belongs to a specific company or individual (the "subject"), and the validation process it goes through to determine if the subject is who it claims to be depends on the level of certification and the CA itself.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Self-Signed Certificates

- During development, it's convenient to use a **self-signed** certificate
- Browsers typically display a warning if you browse to a Web site that uses self-signed certificates



Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Obtaining a Certificate from a CA

- To obtain a certificate from a certificate authority, you typically:
    - Generate your public/private key pair
    - Generate a **Certificate Signing Request** (CSR) and send that the CA
- The CA will verify your identity in some fashion and then issue the certificate
- If you are going to use the certificate on a Web server, you then install the certificate in a server-dependent fashion

---

The most common format for CSRs is the PKCS#10 specification.

Of course, most CAs do not provide their service for free, so you will typically need to pay for your certificate. In addition, there are different "levels" of certificates based on how stringently the CA verifies identity.

If you are using PKI in Java, you also typically install the certificate into a "keystore" and specify the keystore when the application runs.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Introduction to SSL/TLS

- The **Secure Sockets Layer** (SSL) provides encryption of traffic sent via TCP sockets
- **Transport Layer Security** (TLS) is the successor to SSL
- HTTPS is the combination of HTTP and SSL/TLS
- Most Web browsers and Web servers can exchange keys to use HTTPS to provide authentication and confidentiality



Java programs can use SSL/TLS via the Java Secure Socket Extension (JSSE) API.

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Chapter Summary

In this chapter, you learned about:

- The basics of encryption and digital signatures

Copyright © Descriptor Systems, 2001-2016. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum