

Class Eleven

Topics:

- **Principals of Cryptography**
- **Public & Private Key Systems**
- **Digital Signatures**
- **Public key certificates and public key infrastructure (PKI)**
- **Commercial uses of Cryptography**
- **Beware of Snake Oil!**
- **EXTRA: Using GNU Privacy Guard**

8/24/2007

Class 11

1

What is Cryptography?

- Crypto = secret; Graphy = writing
- The art of keeping messages secure
- "Exchanging a lot of small secrets for one big secret"
- Cryptography helps provide:
 - Authentication
 - Data Integrity
 - Privacy
 - Non-Repudiation
- Related areas of study:
 - Cryptanalysis - science of uncovering secrets
 - Cryptology - mathematics of cryptography and cryptanalysis

8/24/2007

Class 11

2

Where is Cryptography used?

- SSL, secure Web access
- IPSec, for secure VPN access for remote users to a private central network
- WEP/WPA, for wireless access security (with varying degrees of success)
- Microsoft Encrypting File System (EFS), keeping stored files secure especially on laptop computers that may be easily stolen
- Code Signing, to verify the author of an executable
 - Microsoft's "Authenticode"
 - Open Source Software – E.g., GnuPG digital signatures to verify software distribution
- Many authentication methods (Kerberos, RADIUS, CHAP)

These are cryptographic applications

8/24/2007

Class 11

3

Cryptography Definitions

The Basics

- **Plaintext** - The "in the clear" message, prior to encryption
- **Ciphertext** - Plaintext which has been encrypted
- **Key** - A secret or protected value used by the encryption algorithm to create ciphertext from plain text.
- **Algorithm** - The method used to encrypt plaintext. The method is not assumed to be secret.

"Key" Definitions

- **Keyspace** - The set of all possible keys for a given encryption algorithm.
- **Key Management** - The process of generating, distributing, storing, and destroying cryptographic keys.

Types of Cipher

- **Permutation Cipher** - A cipher which "shuffles" or recombines the elements of plaintext (characters or bits) to make ciphertext.
- **Substitution Cipher** - A cipher which substitutes plaintext (bits, bytes, or other patterns) with something else to make ciphertext.

Related Topics

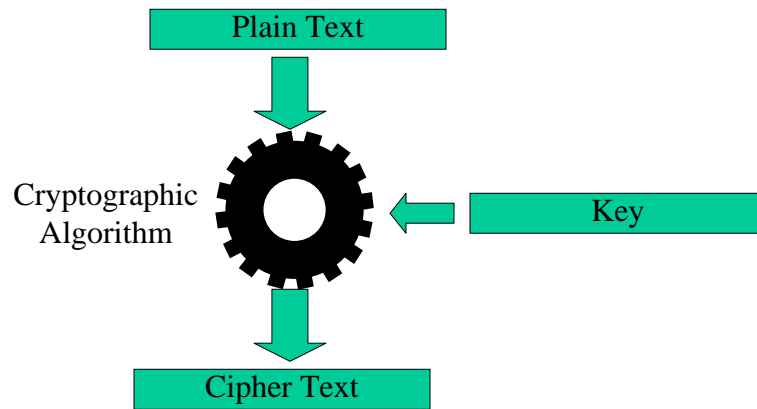
- **Code** - Cryptosystem that deals with linguistic units - words, phrases, etc. as opposed to a **cipher** which encodes symbols (letters, bytes, bits)
- **Steganography** -hiding a secret message in another innocuous message

8/24/2007

Class 11

4

How Cryptography Works

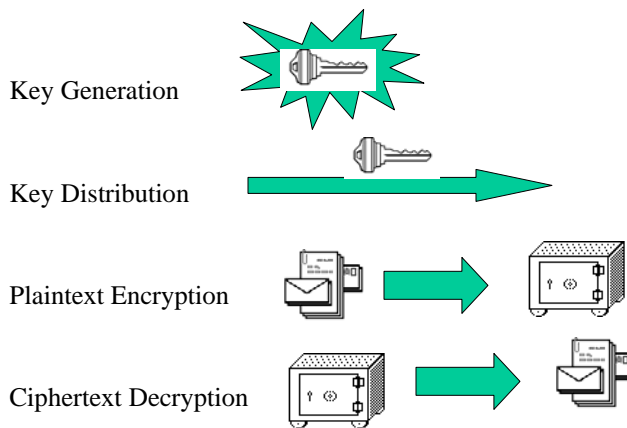


8/24/2007

Class 11

5

Steps in Cryptography



8/24/2007

Class 11

6

Cryptographic System Components

E-mail, Secure Credit Card Payment, etc.

Key Exchange, Signature Verification,
etc.

Block, Stream, etc.

DES, RSA, Blowfish, etc.

APIs	Application
APIs	Protocol
APIs	Mode
	Algorithm

8/24/2007

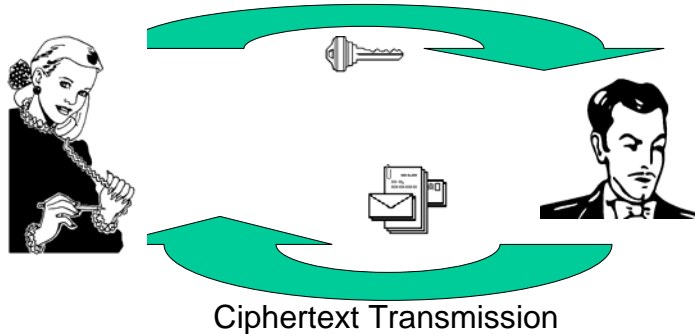
Class 11

7

Types of Cryptography:

Traditional Symmetric or Private Key:

Key Distribution



8/24/2007

Class 11

8

Cryptographic Algorithms

Symmetric Key

- Data Encryption Standard (DES)
- IDEA
- Rijndael (AES standard)
- Skipjack (used in Clipper chip)

Public Key

- RSA public key
- Digital Signature Algorithm (DSA)
- Others (El Gamal, Knapsack, etc.)

Secure Hash

- MD5
- Secure Hash Algorithm - 1 (SHA-1)
- Secure Hash Algorithm - 256 (SHA - 256)

8/24/2007

Class 11

9

DES in Detail

- Began in early 1970's as IBM's Lucifer Algorithm
- Approved as Federal Standard in FIPS Pub 46 and other FIPS documents
- Public domain
- Designed for fast hardware implementation
- Combines substitution and permutation against 64 bit blocks.
- Increasingly vulnerable due to key length
- Ways to strengthen DES:
 - Triple DES (3DES)
 - DESX
- US Federal effort to find DES alternative was Advanced Encryption Standard (AES)
- Replacement announced September 2000 was Rijndael, submitted by Belgians Joan Daemen and Vincent Rijmen

8/24/2007

Class 11

10

Symmetric Key Issues

- 2 parties have same key - means identity of a party can never be uniquely determined solely from cryptographic means. Digital signatures are not possible!
- Keys must be distributed securely to identified individuals. Secure distribution is very difficult and never certain.
- Managing keys in a large organization requires hierarchies of keys encrypting keys encrypting keys...
- You must make pre-arrangements to send an encrypted message to someone. You must send them a specific key by secure means.

8/24/2007

Class 11

11

Cryptographic Modes

- Defined initially for DES (FIPS PUB 81)
- Relevant to any Block cipher
- Generally defined only for symmetric key ciphers – not relevant to public/asymmetric key ciphers
- Stream Modes:
 - Cipher Feedback (CFB)
 - Output Feedback (OFB)
- Block Modes
 - Electronic Codebook (ECB)
 - Cipher Block Chaining (CBC)
- Modes have different characteristics:
 - Cryptographic vulnerabilities
 - Error propagation
 - Speed of encryption/decryption
 - Ability to provide non-sequential data access

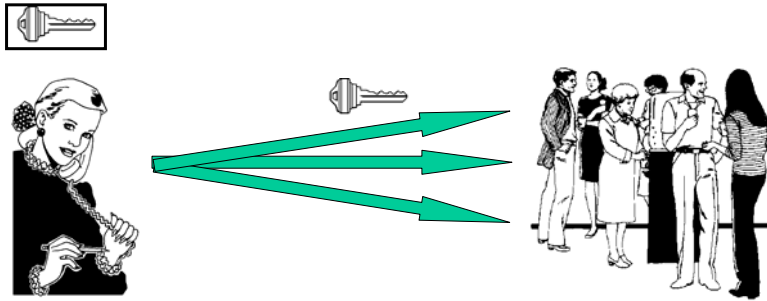
8/24/2007

Class 11

12

Types of Cryptography

Asymmetric or Public Key: Key Distribution



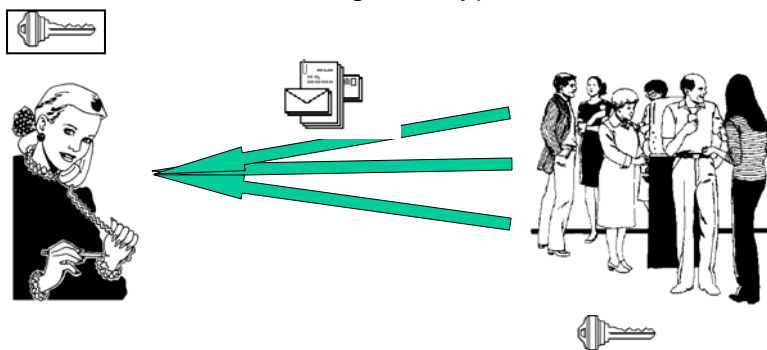
8/24/2007

Class 11

13

Types of Cryptography

Asymmetric or Public Key: Message Encryption



8/24/2007

Class 11

14

Public Key vs. Private Key Algorithms

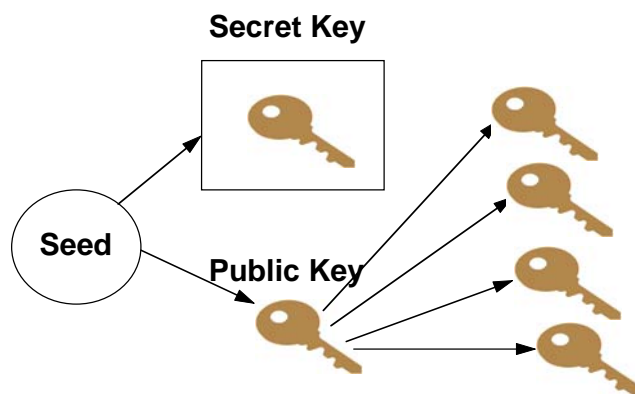
- Private \Rightarrow Symmetric Key
 - Encryption key is same as decryption key
 - Traditional, used for centuries
 - Key management is problematic
 - Computationally fast
- Public \Rightarrow Asymmetric Key
 - Public discovery in mid 1970's (NSA claims earlier discovery)
 - Encryption key is separate from decryption key
 - Encryption key is publicly distributed, decryption key is secret
 - Simplifies key management
 - Inverse process provides for digital signature
 - Susceptible to "chosen plaintext" attacks
 - Slow, very compute intensive

8/24/2007

Class 11

15

Public Key Protocols - Key Generation & Distribution

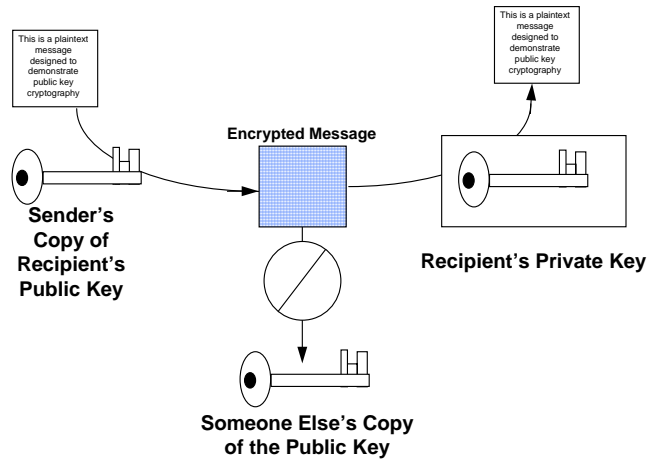


8/24/2007

Class 11

16

Public Key Protocols - Encryption and Decryption



8/24/2007

Class 11

17

Secure Hash

- Create a smaller data element out of a bigger one, such that one cannot:
 - Easily re-create original back out of hash
 - Modify the original to give the same hash
 - Find some other, unrelated text which hashes to the same value (so-called "Birthday Attacks").
- These characteristics define the difference between a secure hash and an ordinary checksum
- Secure Hash Algorithms
 - MD4
 - MD5
 - SHA1

8/24/2007

Class 11

18

Digital Signatures - A form of message authentication

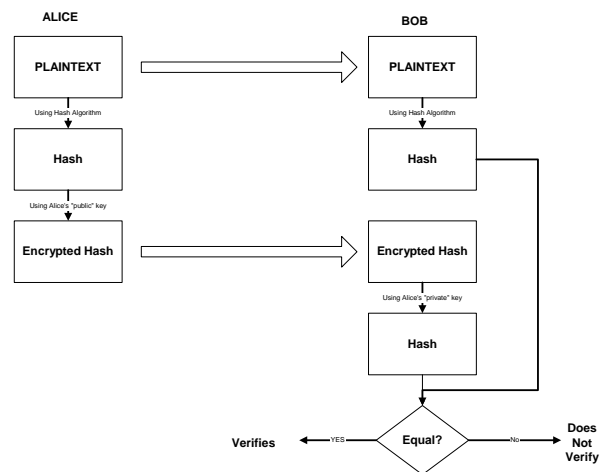
- A Public Key Protocol
 - Key Generation - Same as with encryption
 - Key Distribution - This time the secret key, or the decryption key is distributed. The public or encryption key is kept secret.
 - Message Signing - The sender encrypts with the public key.
 - Signature Validation - The recipient decrypts using the private key. Successful decryption verifies that only the key owner could have created the message.
- To save space and computation time, usually only a message digest or secure checksum is signed. The recipient validates the checksum against that of the plaintext message to verify signature.

8/24/2007

Class 11

19

Digital Signatures Illustrated



8/24/2007

Class 11

20

Public Key Infrastructure: How to distribute and verify public keys

- Public key cryptography changes focus of attacks from theft of secret key to impersonation ("man in the middle")
- Use of public key methods for signature raises the risk of attack by spoofing, rather than breach of confidentiality present with symmetric keys
- A public key is just a set of bits. Administrative procedures and technical controls are required to bind a public key to an identity.
- A trusted entity verifies that the key belongs to the entity (individual or corporate) and asserts this trust by signing the key
- If the recipient of the public key wants to verify it belongs to the correct party, they verify that the key is signed by a trusted entity.

Certification attempts to alleviate impersonation attacks by binding a public key to a real identity

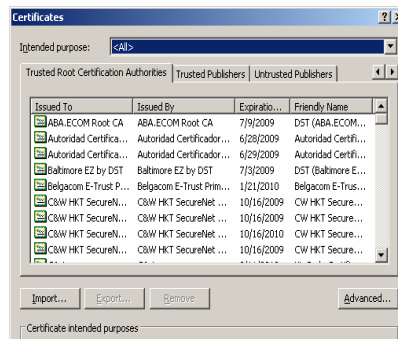
8/24/2007

Class 11

21

Certification Authorities

- X.509 is a widely accepted standard for key certificates
- A certificate authority binds a key to an identity by signing both
- Many organizations act as certification authorities, signing your key for a fee
- An organization may also "self certify" its keys
- Your Web browser has a "built in" list of trusted certificate authorities
 - Used to verify server certs for SSL
 - If the server cert is not signed by a trusted certificate authority, you will get a warning message



8/24/2007

Class 11

22

X.509 Format

- Certificate
 - Version
 - Serial Number
 - Algorithm ID
 - Issuer
 - Validity
 - Not Before
 - Not After
 - Subject
 - Subject Public Key Info
 - Public Key Algorithm
 - Subject Public Key
 - Issuer Unique Identifier (Optional)
 - Subject Unique Identifier (Optional)
 - Extensions (Optional)
- Certificate Signature Algorithm
- Certificate Signature

8/24/2007

Class 11

23

Getting an X.509 cert

- Create a Certificate Signing Request (CSR):
 - Using OpenSSL (for Apache Web servers):

```
openssl req -new -nodes -keyout server.key -out server.csr
```
 - This will create the public/private key pair. The private key stays on your server. The public key becomes part of the CSR
- Submit the CSR to the Certificate Authority (CA) via email or other means
- When the CA validates your identity (and is paid!), the CA then generates the X.509 cert and makes it available to you
- When you obtain your cert (via download, etc.) you configure your Web server to use this certificate for all HTTPS/SSL traffic
- Assuming your CA is one of those provided by default in Web browsers, your users will experience seamless cryptographic security

8/24/2007

Class 11

24

Public Key Infrastructure (PKI)

- Management of public keys within an organization:
 - Policies regarding key use
 - Key generation and assignment
 - Publication of keys
 - Key revocation and expiration
 - Key certification
 - Key escrow or recovery
 - Need to maintain old keys to verify old messages (key archival)
- Certificate authorities and certificate management are essential components of PKI

8/24/2007

Class 11

25

Public Key Infrastructure (PKI)

- Components of a PKI:
 - Registration Authority
 - Certification Authority
 - Certificate Repository
 - Certification Practices Statement
- PKI is often integrated with directory services (esp. LDAP)
- See IETF Public Key Working Group
(<http://www.ietf.org/html.charters/pkix-charter.html>)

8/24/2007

Class 11

26

Cryptographic Design Vulnerabilities

- Improper use of secure algorithms
- Combining secure algorithms in a way that negates security
- Bad random number generation
- Flawed software (buffer overflows, poor error checking, etc.)
- Tamper-resistant hardware that isn't
- Flawed trust models
- Trust models that don't apply to the implementation environment
- Social Engineering
- Proprietary algorithms

8/24/2007

Class 11

27

Breaking Secret Codes

• Passive vs. Active Attacks

- Protocol Subversion
 - “Man in the Middle”, third party takeover of key exchange
- Brute Force Key Generation
- Dictionary attack
- Traffic Analysis
- “Rubber Hose” Cryptography
 - Bribery
 - Social Engineering

- Ciphertext Only
- Known Plaintext
- Chosen Plaintext

8/24/2007

Class 11

28

Bad Cryptography Examples

- German WWII Enigma
 - “Cillies”, common operator errors made cracking the codes easier. Included using easy-to-guess message keys like adjacent keyboard letters
 - Same message sent in weak crypto as strong, giving a known plaintext to crack
 - Boarding a German submarine and confiscating a working machine helps too.
- Windows NT Password Encryption
 - Split 14 character password into two groups of seven
 - Padded with nulls
- CSS DVD Encryption
 - Encryption design with 40 bits to be US exportable
 - Encryption algorithm itself is weak, allowing attack based on 25 bits
 - Once of the licensees had the key in plaintext!
 - Encryption broken in full, with associated software tool by a Norwegian teenager
 - See <http://web.lemuria.org/DeCSS/crypto.gq.nu/> for cryptoanalysis

8/24/2007

Class 11

29

Chosing a Product: Avoid Snake Oil

- “Trust Us We Know What We Are Doing”
- Technobabble
- Secret Algorithms
- Revolutionary Breakthroughs and Leading Edge Technology
- Security Experts, Rave Review and Other Useless Certificates
- Unbreakable!
- “Uses One Time Pads”
- Claims Competing Products are Insecure
- Recoverable Keys
- Exportable from USA
- “Military Grade”
- Cryptographic strength enforced by litigation (my addition to this list)

<http://www.interhack.net/people/cmcurtin/snake-oil-faq.html>

8/24/2007

Class 11

30

Extra Material – using GnuPG

Installing GnuPG

- GnuPG is an open source free program to perform basic public key cryptography, including:
 - Key Generation
 - Encryption and decryption of data
 - Generation of and verification of digital signatures
- Primarily designed for Unix, but Windows installable binaries are available
- May be found at <http://www.gnupg.org>
- For Windows systems, you want to download:
 - gnupg-w32cli-1.4.2.2.exe
 - gnupg-w32cli-1.4.2.2.exe.sig
 - Sha1sum.exe
- Best place to download is from <http://ftp.gnupg.org/gcrypt/binary/?C=S;O=D>
- Double click on gnupg-w32cli-1.4.2.2.exe and select the default options on the installer
- In Windows, GnuPG must be run from the command line. Unless the install directory happens to be in your path, you must do the following:
 - Open a command window
 - Change directory to C:\Program Files\GNU\Gnupg

8/24/2007

Class 11

31

Extra Material – using GnuPG

Verifying your GnuPG installation

- To verify your first time GnuPG download using sha1sum:
 - Obtain the SHA1 digest of the installer by typing:
`sha1sum gnupg-w32cli-1.4.2.2.exe`
 - Compare it to the published value on the Web site:
http://www.gnupg.org/en/download/integrity_check.html
 - The values should be the same

8/24/2007

Class 11

32

Extra Material – using GnuPG

Generating your key pair

- To your public private key pair
 - First, come up with a very long, random pass phrase
 - One that is high in entropy
 - Once you are sure you have a good pass phrase, start the key generation process by typing:
`gpg --gen-key`
- The program will ask a few questions:
 - What type of key do you want? (It's OK to pick the default)
 - What keysize do you want? (It's OK to pick the default)
 - When will the key expire (if at all)?
 - Your "real name"
 - Your email address
 - Some sort of comment about the key
 - And the all-important passphrase

8/24/2007

Class 11

33

Extra Material – using GnuPG

Turning your key into ASCII for emailing

- To your turn your public key into an ASCII file
`gpg --armor --output "key.txt" --export "Vincent LeVeque"`
- This exports the public key as the file "key.txt", for you to email as an attachment
- The binary key is turned into ASCII characters (called "armor")
- The key.txt file looks something like

```
-----BEGIN PGP PUBLIC KEY BLOCK-----  
Version: GnuPG v1.4.2.2 (MingW32)
```

```
mQGiBEQeUO4RBADwWbLOAKLCAeX0eF0KsmEXpdy93reL76W5KWltFUMqhQWVXRK  
SyUsSQ2FtwXuCYMAyCiu4bRHh/5q/4ncmdkQGaxK8wrvfKBIcJ0AWTCkyaIOBTjs  
b0sbJdLTnnhGFNLM7Mc9Q3rpZnsvN6wiSA9JcVD9rkVfRv/jHPNlVcmAwCgkUoL  
Ng6e4erXeVbjXfsPIAJhb6cD/iKZZ4ElntaSdso0t8IZv6i6GwWuoQTgQPN6rHRO  
wLZhXa5xLC2MvdJmTg2gXMPG4yC2LfTbNY4jSjP4541kyHTfw9VbJW5fwnorNbw5  
JJDoCDe2ZS0F1dy8BufeilWLz/35KkhAhgLfPWe/34zNVH+bXYBahrUtAGHehz2I  
xmCUBADFrXKSNbxT2CzSpTwiNyeXO2gXwupnqY41d08fthbW7sWmJ8iBPU/mxuGR
```

8/24/2007

Class 11

34

Extra Material – using GnuPG

Encrypting a file

- To import someone else's public key
 - Get the key from a trusted email of other good source
 - Type:

```
gpg --import "BobSmith.txt"
```
 - Use edit-key to set the trust level of the key (how genuine do you find the key?)

```
gpg --edit-key "Bob Smith"
```
- To encrypt a file with someone else's public keys

```
gpg --armor --recipient "Bob Smith" --output "message-to-bob.asc"  
--encrypt "message-to-bob.txt"
```
- The file `message-to-bob.asc` can then be sent to Bob Smith, who can decrypt it with his `gpg` private key

8/24/2007

Class 11

35

Extra Material – using GnuPG

Decrypting a file

- To decrypt a file encrypted with your public key:

```
gpg --decrypt-files "message-to-bob.asc"
```
- You will be prompted to enter your passphrase to complete the decryption.

Signing a file

- To sign a file with your "private" key but not encrypt it:

```
gpg --local-user "Vincent LeVeque" --clearsign "message.txt"
```
- You will then need to enter your passphrase to complete the process.
- You will end up with the file `message.txt.asc`, having the plaintext message and the ASCII-armoured digital signature.

8/24/2007

Class 11

36

Extra Material – using GnuPG

Verify a signed file

- To verify a signed file:
`gpg --verify "message.txt.asc"`
- If it works, you will see something like:

```
gpg: Signature made 03/25/06 16:42:03 using DSA key ID 34C7CB3D
gpg: Good signature from "Vincent LeVeque (first key) <vleveque@sbcglobal.net>"
```