Introduction to Cryptography

CSH6 Chapter 7
"Encryption"
Stephen Cobb &
Corinne Lefrançois

1

Topics

- Basic Concepts & Terminology
- Types of Algorithm
- Cryptanalysis

CSH6 Chapter 7 pp 7.1-7.16

2

Encryption in INFOSEC

- Foundation technology
- Underlies almost everything in INFOSEC
- Ensures or supports
  - Confidentiality
  - Control and possession
  - Integrity
  - Authenticity
  - Non-repudiation

3

Basic Concepts & Terminology (1)

- *Plaintext* (aka *cleartext)*: original, readable data
- *Ciphertext*: scrambled form of plaintext
- *Encryption*: reversible conversion of plaintext into ciphertext
- *Decryption*: conversion of ciphertext back into plaintext
- *Crack* (aka *break) code*: decrypt ciphertext without knowing key

4

Basic Concepts & Terminology (2)

- *Key*: secret allowing encryption and decryption to be restricted to possessors of key
- *Symmetric encryption:* encryption requiring a shared key for both encryption and decryption
- *Asymmetric encryption:* algorithm using a different key for decryption than for encryption

5

Basic Concepts & Terminology (3)

- Keylength: number of bits in key
- Keyspace: number of possible keys
- Keyspace = $2^{keylength}$ *Keylength in bits*

$$2^n \approx 10^{n(\log_{10} 2)}$$

$$\approx 10^{0.30103n}$$

6

Topics

- Basic Concepts & Terminology
- Types of Algorithm
- Cryptanalysis

7

Monoalphabetic Substitution Ciphers (1)

- "Secret decoder ring" or Caesar cipher
- Algorithm uses key = offset and algorithm = transposition;
  - e.g., if offset = 3, then A becomes D, B = E etc.
- Subject to cryptanalysis using known letter frequencies in specific languages
  - English: *etaionshrdlu*. . . .
  - For English alphabet, only 25 possible offsets; therefore maximum 25 tries to find the "key"

8

Monoalphabetic Substitution Cipher: Example

	A	B	C	D	E	F
1	Monoalphabetic Substitution Cipher Demonstration					
2						
3	Enter offset below:			"=IF(A6<>"",MOD(CODE(A6)-CODE("A")+A4,26)+CODE("A"),"")"		
4	5					
5	Cleartext	ASCII	Transformed ASCII code	Ciphertext		
6	T	84	89	Y		
7	H	72	77	M		
8	E	69	74	J		
9						
10	Q	81	86	V		
11	U	85	90	Z		
12	I	73	78	N		
13	C	67	72	H		
14	K	75	80	P		

Sheet1

9

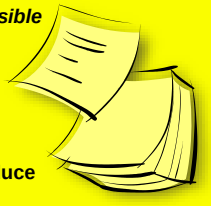
Polyalphabetic Substitution Ciphers

- Can use different offsets for *each position* in plaintext
  - E.g., *Vigenère* cipher is like 26 Caesar ciphers
  - Use key indicating *which offset* to use for *which position* in sequence of 26 letters
- See
  http://www.trincoll.edu/depts/cpsc/cryptography/vigenere.html
  Or
  http://en.wikipedia.org/wiki/Vigen%e8re_Cipher
  for detailed illustrations of how to perform the Vigenère cipher with a particular key on a specific plaintext

10

One-Time Pad

- Use a fixed and shared secret to determine offsets
- In theory, is only cipher *impossible* to break IFF
  - Pad kept secret
  - Key data truly random
  - Key data never re-used
- In practice, people use natural language (e.g., novels) and reduce strength of algorithm
- Major problem: how to distribute the pad securely?
  - *Explain the problem.*



IFF means "if and only if"

11

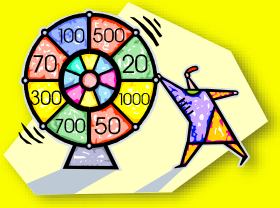
Secure Key Distribution

- The problem of distributing a key securely is completely general to all *secret-key* algorithms
  - Shared secret essential for both enciphering and deciphering data
- Therefore both sender and receiver must share the secret securely
- But if it were secure to transmit the key, you could transmit the plaintext message too
- So how do you get the secret from one to the other securely?
- Need an alternate communications channel with higher security

12

Example of Linear Congruential Pseudo-Random Number Generator

$$n_{i+1}=frac(\pi^{n_i})$$



13

Linear Congruential Pseudo-Random Number Generators

- Amateurs assume that RAND() function can be used as basis of one-time pad
  - But in fact such functions are NOT truly random
- For one thing, use floating-point data with specific number (e.g., 17) of significant figures
  - Thus must inevitably repeat (WHY?)
- Therefore ciphers with natural-language plaintext are subject to frequency analysis
  - WHY?
- Nonetheless, can be useful for simulations and demonstrations

14

Topics

- Basic Concepts & Terminology
- Types of Algorithm
- Cryptanalysis

15

Cryptanalysis

- Kerckhoffs' Principle
- Cryptanalytical Methods
- Types of Cryptanalytical Attacks

16

Kerckhoffs' Principle*

- The strength of an encryption algorithm does not reside in the secrecy of the algorithm

Corollary:

- The strength of an encryption algorithm is not measurable unless the algorithm is known

** Published in 1883. Not to be confused with Kirchoff's Laws (physics)*

17

Dangers of Proprietary Algorithms

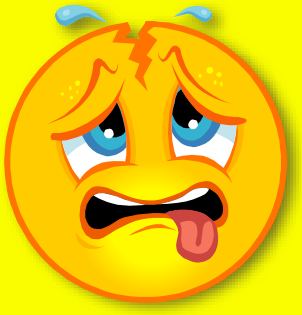
Therefore beware of secret, proprietary algorithms

- Many amateurs have failed utterly to defeat cryptanalysis
- Must demonstrate that even with *knowledge of the algorithm* and even *knowledge of a plaintext & ciphertext* sample, still too expensive to decrypt general ciphertext to make cryptanalysis worthwhile

18

Cryptanalytical Methods

- Frequency-Based Cryptanalysis
- Brute-Force Cracking
- Attacking Weak Algorithms



19

Frequency-Based Cryptanalysis

- Possible to use frequency of single letters and *digraphs* (pairs of letters) to analyze ciphertext
  - ❑But this technique works only for plaintext based on natural language
  - ❑Must know (or guess) which language is used*
  - ❑Need large amounts of data
- Does not help with cryptanalysis of purely numerical data unless there are regularities in the plaintext**

* e.g., frequency of single letters in plain English follows sequence *approx* like *ETAOINSHRDLU*

** But in detecting accounting fraud, *Benford's Law* can help. See for example *http://www.usfsp.edu/gkearns/Articles_Fraud/Benford%2520Analysis%2520Article.pdf*

20

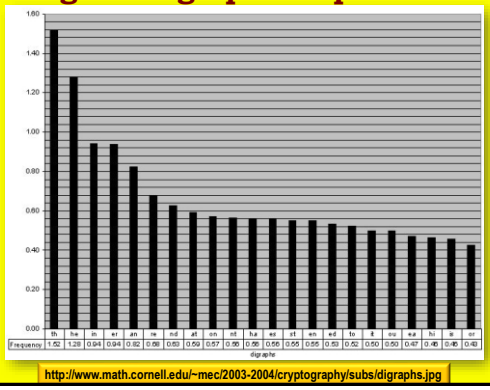
Frequency-Based Analysis: A Bit More Detail

- Ciphertext only: Study patterns in ciphertext
  - ❑Digraphs: pairs of symbols in sequence
  - ❑Trigraphs: sets of three symbols in a row
- Plot frequencies of digraphs, trigraphs etc.
- Tables exist of known frequencies of transition probabilities for letters in natural language
  - ❑E.g., in English *th* more common than *tx*
  - ❑AKA *Markoff Chain probabilities*
- Use transition probabilities to spot likely transformed ciphertext

See chart on next slide from Cornell University's "Math Explorer's Club" http://www.math.cornell.edu/~mec/ (Used witih permission)

21

English Digraph Frequencies



http://www.math.cornell.edu/~mec/2003-2004/cryptography/subs/digraphs.jpg

22

Brute-Force Cracking

- Try every possible key
  - ❑Facilitated by massively parallel computing
- *Dictionary attacks* narrow the range of keys
  - ❑Helpful when one suspects that the target user has chosen *bad key*
    - ✓Names of pets, friends, sports teams, hobbies, objects on desk
  - ❑*Password-cracking programs* use dictionaries
    - ✓Try every word and combination
    - ✓Can also introduce numbers and symbols

23

Password-Cracking Programs

- Most systems store *one-way encrypted* passwords in a file
  - ❑E.g., /password/etc. file in UNIX systems
  - ❑*One-way* because cannot deduce original password from knowing ciphertext
    - ✓Why not?
- Password encryption algorithm is known
  - ❑Encrypt all possible passwords
  - ❑Compare ciphertexts against stored values
  - ❑Hits indicate valid passwords or equivalents (WHY?)

Defending Against Password-Cracking Programs

- How can you choose passwords that are hard to crack?
- Don't use real words
  - Why?
- Introduce numbers and symbols into the password sequence
  - Why?
- Change your password periodically
  - Why?
- Don't use the same password on public Web sites as on important / secure production sites
  - Why?

Interfering with Brute-Force Cracking

- Need to know the algorithm used for encryption
  - Why?
- Must be able to *recognize* successful decryption
  - Why?
- *Superencryption* of plaintext makes brute-force cracking *more difficult* but not impossible
  - Suppose adversary uses two algorithms, E_1 and E_2 using keys k_1 and k_2 respectively
  - Thus must crack $E_{2k2}((E_{1k1}(P))$ which has a *keyspace* that is the *product* of k1 and k2
- Using different data encoding schemes can confuse cryptanalyst (e.g., use EBCDIC & ASCII)

26

Attacking Weak Algorithms

- Find methods of deducing key due to bad algorithms
  - But may be able to find key only one message at a time
  - May be able to demonstrate that algorithm is fundamentally flawed – may not successfully protect ciphertext against analysis (e.g., Knapsack algorithm)
- Fundamental principle: strength of encryption measured by *time* and *cost* of cryptanalysis *for specific application*

27

Types of Cryptanalytical Attacks

- *Ciphertext only:*
  - No idea of plaintext at all
- *Known plaintext:*
  - Examine plaintext + ciphertext
- *Chosen plaintext:*
  - Choose plaintext and examine ciphertext
- *Adaptive chosen plaintext* (aka *differential cryptanalysis*):
  - Repeatedly choose plaintext and examine results of encryption

28

Start here Monday

29

Stronger Encryption & the PKC

CSH6 Chapter 7 pp 7.16-7.43

- Stronger Encryption
  - Transposition Ciphers
  - Block Ciphers & Chaining
  - Product Ciphers
  - DES: Data Encryption Standard
  - Triple DES (3DES)
  - AES: Advanced Encryption Standard
- Public Key Cryptosystem (PKC)

30

Stronger Encryption

- Substitution ciphers are generally *weak* (i.e., cheap or quick to crack)
- Stronger ciphers include
  - Transposition ciphers
  - Block ciphers & chaining
  - Product ciphers

31

Transposition Ciphers

- Change order of plaintext
  - Use specific algorithm (rule)
- Example: matrix rotation
  - Matrix dimensions can serve as key; e.g., 6 x 8 then read as 8 x 6
  - Read text in opposite direction of matrix
  - See next slide for illustration
- Interferes with expected frequencies of digraphs, trigraphs etc.

32

Transposition Ciphers: Example

The quick brown fox jumped over the lazy dogs.
Tioxerlohcw d ageeknj tzs uohy.qbfmve urope d

T	h	e		q	u
l	c	k		b	r
o	w	n		f	o
x		j	u	m	p
e	d		o	v	e
r		t	h	e	
l	a	z	y		d
o	g	s	.		

T	l	o	x	e	r	l	o
h	c	w		d		a	g
e	k	n	j		t	z	s
			u	o	h	y	.
q	b	f	m	v	e		
u	r	o	p	e		d	

33

Cryptanalytical Attacks on Transposition Ciphers

- Susceptible to combination of brute-force and frequency-based analysis
  - Try different offsets looking for familiar / frequent digraphs
  - This helps to determine the original matrix and its rotation
- Nonetheless, transposition is an important part of more complex encryption schemes

Copy of the Rosetta Stone (Create Commons License)

34

Block Ciphers & Chaining

- Introduce additional complexity:
  - Break plaintext into blocks
  - Apply transposition and substitution ciphers to each block
- Chaining
  - Introduce an element from previous block into next block
  - Thus specific ciphertext becomes context dependent
  - Risk: transmission error may render ciphertext unrecoverable

35

Product Ciphers

- Use all methods at once
  - Blocks
  - Chaining
  - Transpositions
- Problem: no mathematical way of proving that a product cipher is actually strong
  - Therefore try to look at degree of randomness
  - Measure frequencies of symbols
  - Also 1st, 2nd, 3rd . . . nth-order correlations

A B C D E F G H I J K L M N

36

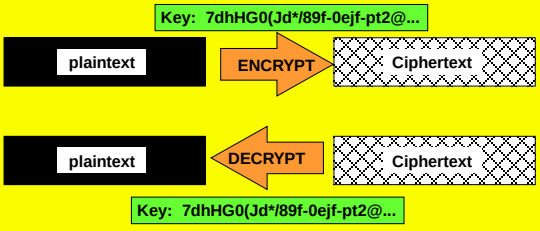
DES: Data Encryption Standard

- 1971: IBM created "Lucifer"
  - Used in financial transactions
  - Single-key symmetric algorithm using 56-bit keys
- NIST selected Lucifer to be the DES
  - 1977: Federal Information Processing Standard (FIPS) 46
  - US government standard for unclassified use
- Widely adopted in commercial banking
- Originally defined in hardware
  - Increased general computer speed led to approval for software implementations

37

Encryption: DES

- Data Encryption Standard
  - example of *symmetric* encryption algorithm



38

Triple DES (3DES)

$$C = E_{k1}[D_{k2}[E_{k1}(P)]]$$

Where

- $E_{k1}(P)$ means "encrypt plaintext using key 1"
- C means "ciphertext"
- Keylength 110 bits
- Keyspace $2^{110} \approx 10^{36}$
- Much used for key management (see next lecture)

39

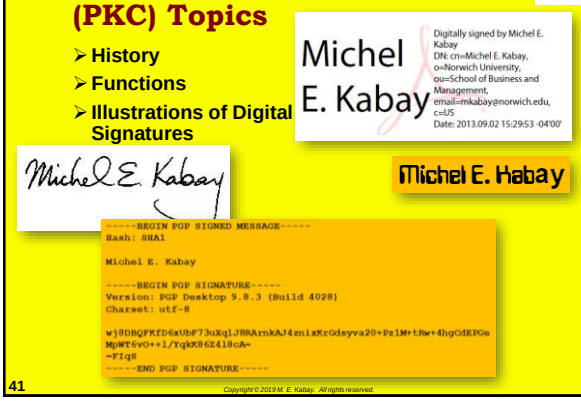
AES: Advanced Encryption Standard

- 1997: NIST requested new encryption algorithm
  - Protect sensitive unclassified US government information
- Competition among candidate algorithms
  - Winner: Rijndael ("Rhine doll")
  - Drs Joan Daeman & Vincent Rijmen from Belgium
- Block cipher w/ variable block length & variable key length (easily extendible)
  - Easy to implement in hardware (e.g., smart cards) as well as software

40

Public Key Cryptosystem (PKC) Topics

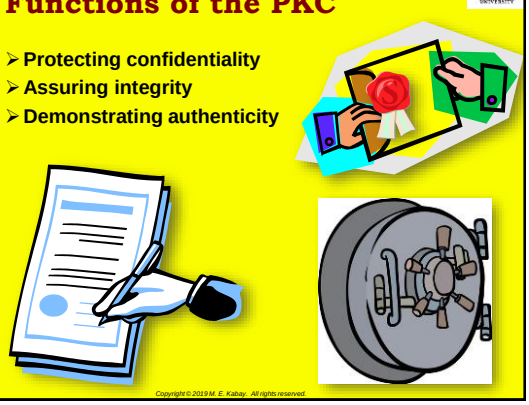
- History
- Functions
- Illustrations of Digital Signatures



41

Functions of the PKC

- Protecting confidentiality
- Assuring integrity
- Demonstrating authenticity



42

PKC: Public Key Cryptosystem (1)

- Discoverers / Inventors – 1974-1975
  - Stanford University
    - ✓Whitfield Diffie
    - ✓Martin Hellman
  - UC Berkeley
    - ✓Ralph Merkle
- Radically new concept at the time
  - Create 2 complementary keys
  - Make one of them public
  - Dispense with problems of secure key distribution for decryption



Diffie, Hellman, Merkle

43

PKC History (2)

- Idea developed into the Public Key Cryptosystem (PKC) by three scientists
  - Ron Rivest
  - Adi Shamir
  - Len Adleman
- Founded RSA Data Security Inc. (RSADSI)
  - Now one of best-known security firms
  - Sponsor highly-regarded annual conference
  - Web site has much useful information

http://www.rsa.com

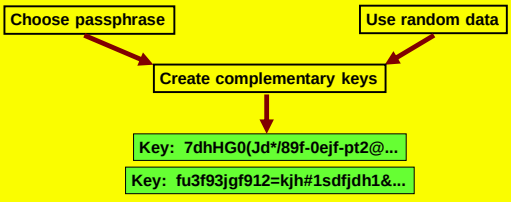


Rivest, Shamir, Adleman

44

Encryption Using PKC (1)

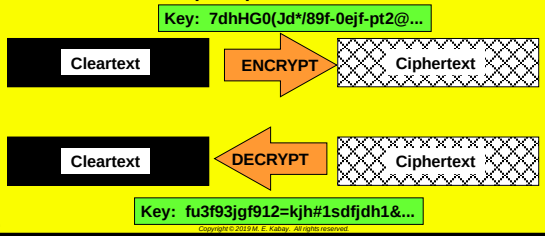
- Key generation produces 2 keys
  - Each can decrypt the ciphertext produced by the other
  - One is defined as *public*
  - Other is kept as *private*



45

Encryption Using PKC (2)

- Key generation produces 2 keys
  - Each can decrypt the ciphertext produced by the other
  - One is defined as *public*
  - Other is kept as *private*



46

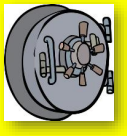
Using the PKC for Encryption (1)

- How can we send a message so only the desired *recipient* can read it? Select the correct answers in the following description.
- First, encrypt the plaintext using the

☐ Sender's ☑ Public Key

or or

☑ Recipient's ☐ Private



47

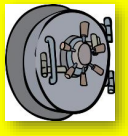
Using the PKC for Encryption (2)

- Send the ciphertext to the recipient
- Next, decrypt the ciphertext using the

☐ Sender's ☐ Public Key

or or

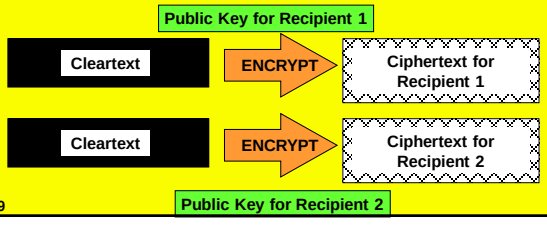
☑ Recipient's ☑ Private



48

Sending a Ciphertext to Multiple Recipients

- What if you have to send a message securely to many people?
  - Obvious way is to encrypt the message separately for each recipient
  - Thus generate as many ciphertexts as recipients



49

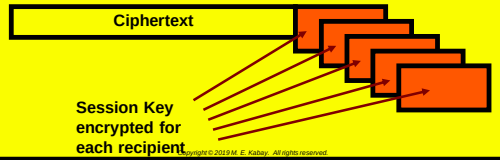
Multiple Recipients (2)

- However, e-mail normally makes it easy to send one message to multiple recipients
  - Don't want to send a different ciphertext to each recipient
- PKC algorithms are computationally demanding
  - Can take significant time to encrypt messages
  - Encrypting same message *n* times could take a long time

50

Multiple Recipients (3)

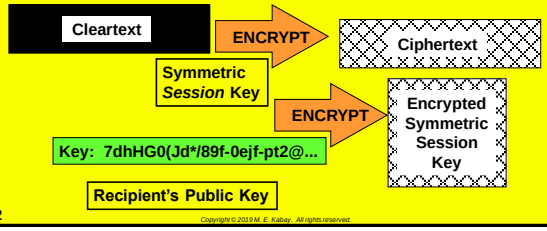
- Use a one-time symmetric key to create ciphertext -- the *session key*
- Prepare as many copies of this symmetric key as necessary to reach all the recipients
- Encrypt a copy of the symmetric key with the public key of a specific recipient
  - Do this step for each recipient



51

Multiple Recipients (4)

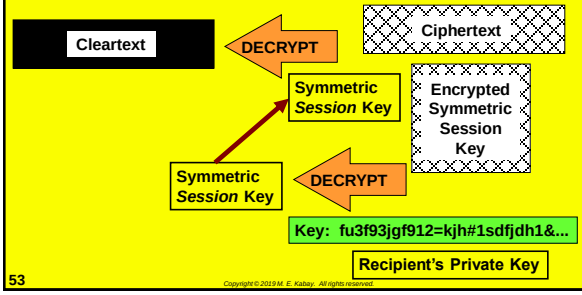
- Send both the ciphertext and the encrypted decryption keys to all the recipients



52

Multiple Recipients (5)

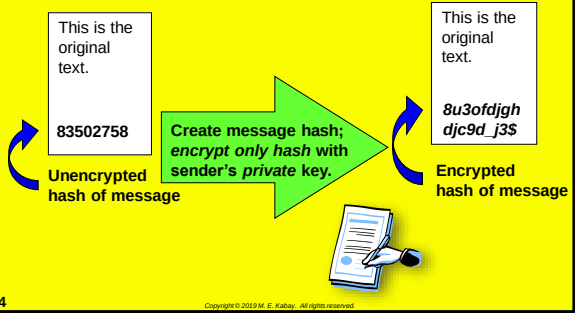
- Each recipient decrypts the asymmetric key using their own private key



53

Digital Signature Using PKC (1)

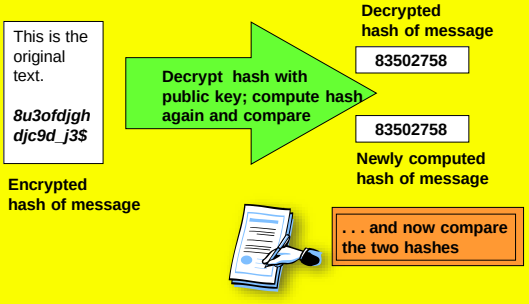
- Signing a document using PKC



54

Digital Signature Using PKC (2)

- Verifying the signature using PKC



55

Digital Signature Using PKC (3)

- IF (decrypted hash = newly computed hash)
- THEN
  - The message has not been modified in transit
  - The message was signed by the owner of the private key corresponding to the public key*

** Or by someone who has compromised that key!*

56

PGP & GPG

- Screenshots illustrating use of popular PKC encryption and digital signature tools
- Creating a new key
- Signing a document
- Effects of corrupting a document
- Encrypting a document
- Decrypting a document
- Establishing the Web of Trust

57

PGP

- Tool for applying PKC
- Supports
  - File encryption / decryption
  - Message encryption / decryption
  - Disk encryption / decryption
  - Authentication through digital signatures
- Provides facilities for establishing *web of trust* (see lecture on PKI) among users
  - Key distribution
  - Key signing
  - Key revocation & expiration

58

PGP: Pretty Good Privacy

- Phil Zimmermann
  - Computer programmer
  - Civil libertarian
- Released Pretty Good Privacy*
  - June 1991 – worldwide distribution
  - Became most widely-used encryption program in world
- PZ worked with Viacrypt to create PGP 3 (renamed PGP 5) in 1997
- Developed OpenPGP (RFC 4880)
- Free Software Foundation developed GNU Privacy Guard (GPG) in compliance with OpenPGP

* Reference to Garrison Keillor's *Prairie Home Companion* radio show, where a mythical sponsor was "Ralph's Pretty Good Grocery."

59

PGP (cont'd)

- Zimmermann investigated by grand jury for *supposedly* violating ITAR (Intl Traffic in Arms Regulations)
  - Much protest (cypherpunks & also Prof Kabay writing vehemently in *Network World*!)
  - PZ responded by publishing entire code in a book – not subject to ITAR
  - Prosecution abandoned after several years
- Bought by Network Associates (NAI) in 1997
- New PGP company created 2002 & bought code from NAI
- Now PGP Corporation; see FAQ for more info http://www.pgp.com/company/faqs.html

60

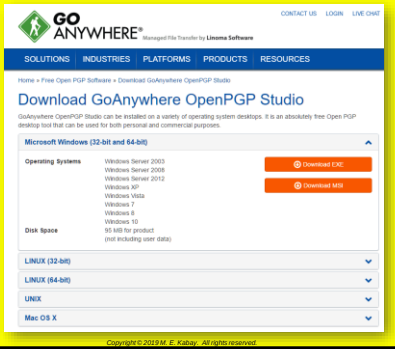
PGP Desktop*

** In Aug 2016 the current version was PGP Desktop v 10.1.1*

61

Getting PGP Free

- https://www.goanywhere.com/openpgp-studio/download



62

Encryption: PGP Demo

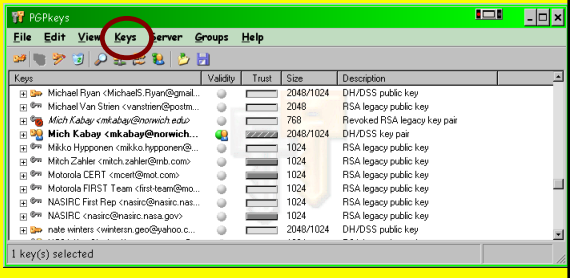
Screenshots of old commercial version 8.1:

- Creating a private key / public key pair
- Signing a document with a private key
- Validating a signature with a public key
- Effect of a single-byte change on validity of a digital signature
- Encrypting a document using a public key
- Decrypting a document using a private key
- Effect of a single-byte change on decryption
- Signing someone's public key
- GPG

**Using screen captures*

63

PGP: Creating a Private Key / Public Key Pair

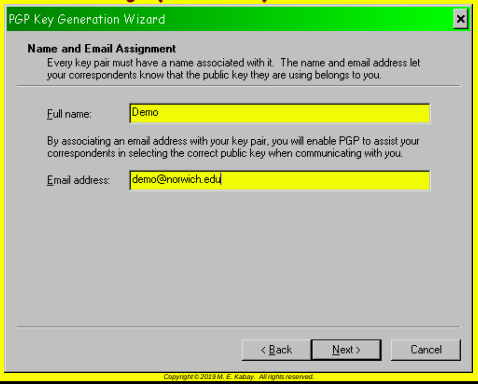


64

New Key (cont'd)

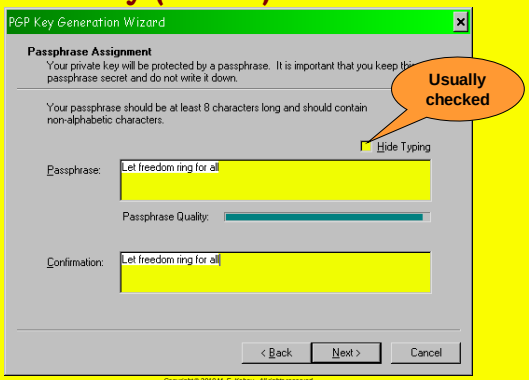
65

New Key (cont'd)



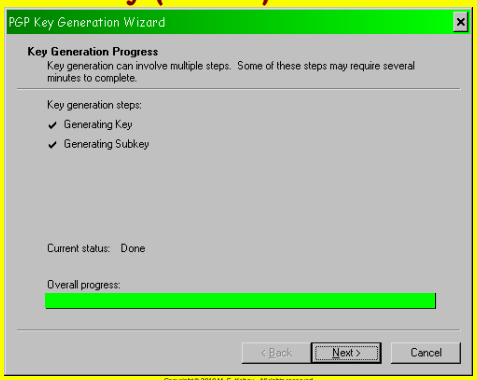
66

New Key (cont'd)



67

New Key (cont'd)

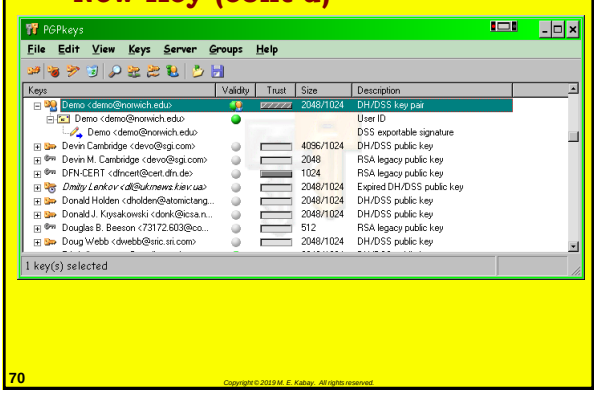


68

New Key (cont'd)

69

New Key (cont'd)



70

PGP: Signing a Document With a Private Key

```
-----BEGIN PGP SIGNED MESSAGE-----

This is the original text.
-----BEGIN PGP SIGNATURE-----
Version: PGP Personal Privacy 6.5.8
Comment: Digital signatures increase security for everyone.

iQB1AwUBO+dfOzPd6/an40lzAQF7rQL+MPLG+V/J8H0QhDLE3JUDo
4tYqPOJ682r
zv457NmbycGz05hlq6TtNuLCWRbashcQZ7eWiwHybgLwhgbYY8G4
HnAYFHh6vKyb
3csmoTm29uHD+/4av98awA23Bf1aEW+t
=WhgQ
-----END PGP SIGNATURE-----
```

71

PGP: Validating a Signature With a Public Key

72

PGP: Single-byte Change Alters Digital Signature

-----BEGIN PGP SIGNED MESSAGE-----

This is the original text!

-----BEGIN PGP SIGNATURE-----
Version: PGP Personal Privacy 6.5.8
Comment: Digital signatures increase security for everyone.

iQB1AwUBO+dfOzPd6/an40lzAQF7rQL+MPLG+V/J8H0QhDLE3JUDo
4tYqPOJ682r
zv457NmbycGz05hlq6TtNuLCWRbashcQZ7eWiwHybgLwhgbYY8G4
HnAYFHh6vKyb
3csmoTm29uHD+/4av98awA23Bf1aEW+t
=WhgQ
-----END PGP SIGNATURE-----

73

Single-byte Change Alters Digital Signature (cont'd)

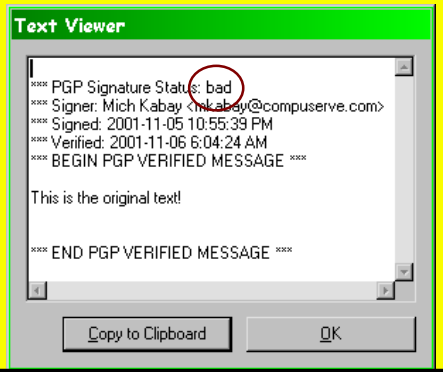
Everything from here on is different

iQB1AwUBO+dfOzPd6/an40lzAQ F7rQL+MPLG+V/J8H0QhDLE3JU Do4tYqPOJ682r zv457NmbycGz05hlq6TtNuLCW RbashcQZ7eWiwHybgLwhgbYY8 G4HnAYFHh6vKyb 3csmoTm29uHD+/4av98awA23Bf 1aEW+t =WhgQ	iQB1AwUBO+fETTPd6/an40lzAQ FagQL/Thfw3DAJA/KRgoH+kSFc oRL39eJp4s5h v3zeHUesOkgQk2zSUF+evbRhw 5cxZJkUA1Qid6cg58tEaP9jl+7J3 wILmJrFPF/K L42qO9yjxalNssnflUaSf7ry7xXV3 bIK =svYa
This is the original text.	This is the modified text!

Remember, exclamation mark is different from original

74

Single-byte Change (cont'd)

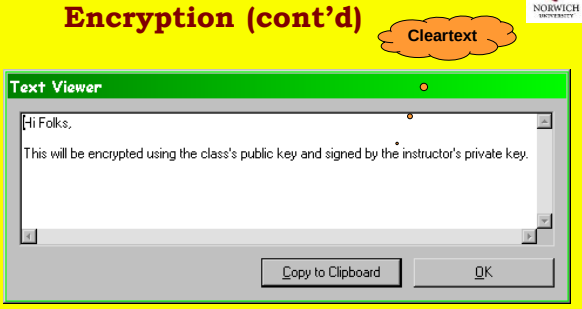


75

PGP: Encrypting a Document Using a Public Key

76

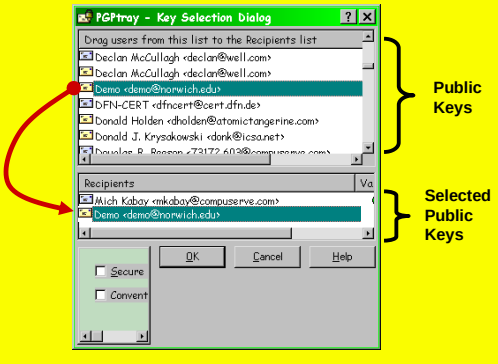
Encryption (cont'd)



Note that the sender should ALWAYS encrypt using the ***sender's public key*** as well as the ***recipient's public key*** to allow decryption by the sender.

77

Encryption (cont'd)



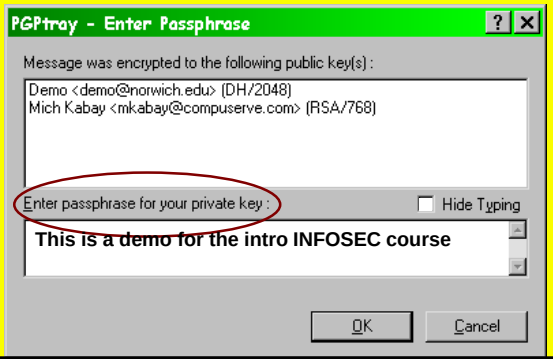
78

Encryption (cont'd)

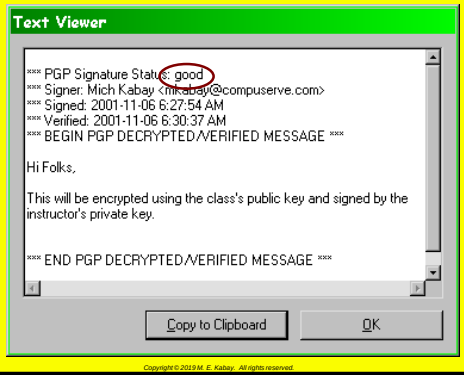
Ciphertext

```
-----BEGIN PGP MESSAGE-----
Version: PGP Personal Privacy 6.5.8
Comment: Digital signatures increase security for everyone.

hQIOA8BxvYx90+tvEAf/eXW8l9vY45PwN27gd4kzg0DyoCJiucXXA4eDtzHqBUJm
nWadsATpcspqQNaZwgp8d5FwxRhafOtG2WqSXl2SgsFUljraYgXdlRBm2qeU/dKh
8+NfNZv2ol5bPaDAg9oiYDRV7KcFAMm3RfliSTRv6f72mE+Y6VPAvQmL6zjLAe0D
i/9n7Sh0UyM+71YuNeY0V+EZ88rTBa7JP4xr/GfAVKwwSHlWEhOfiwe82LMxOnSg
t8vj6amULTYp4daULqDt+qewULR4XWLhQ73zVT9578BSaUmgSnlRXvelsHj4fj9n
2+Ry7QZZL7k2XDybljAw8QfLcShDgC3n5Wlt+mWLwwgA8zJSyzTdKYtVh0kuo3CP
9kkGpkgqyqitVFhua+J4bvQrrhLCLBzCWXBl6RSdFvChJ7JNhtgzajV16L5SepDS
tHMcahPvGoE1e+uG5P80xqngbc5Sl7BY0jfsUjGBlyfVRpt+oC37l4W2ZKZ6mgRU
4yY+H2NTi28gC0SfXjzwRwS4qARSicjluUkYVuuRqSDfeK4slaV8FVs/xg+Ra7U3
HuyD8VmBi0/uO3RSsyPldbh2FqA1+raqeL2yuoUXLe8DI1CToLQ595/4s0wiNIzT
Ys2W5ZbOT+P2gjoVfNaRQIWFzntkRXBcs/Kx9R8pu+NMTdpJsip0oqiGfH2jtqk
2IUAbAMz3ev2p+NJcwEC/i8eoq7FornXxjxzB5/hQ3le0Ww+Vxk2LZVHdT07eqiz
8eN9e2XSxt4cr216MFCPOf1Wj8j3suYvX7mnJa7hu0mvJvxaIHawEq3U+r4ZczcC
e+q9YAukL9iXyRHYCrWZWqT6dj1Bynmi6hzgLfApq+270u8QwaGud1d/aOHGlhD7
+9JcSC1AkXoVZsA3ltaOMPP7frIOOZfdyFzr874mhG+Lru9sBFUy1S7h3gNQfUwx
ZEe9uGnDfNUth33VrrtMqjlvUjh9gZN8BOXdkKOk0WGkVvJDy6D8bSphSaQR+vvP
V83K4BaD24kiAJ70NLbeQXPx2H5j0HYT+4bD0RTt4RQgbeRLhqgwZfKpVdldXC13
P06+MTFliqcs+p4OJo/Mj67H6x877KWU3G7SKG4pBpgmy6KwKeUW8j9EpcKTgh+
+8/tgDZNAzcm8vnCQ9HEAAsN6KM9V0qoCiyDDA==
=llr+
-----END PGP MESSAGE-----
```


PGP: Decrypting a Document Using a Private Key



Decryption (cont'd)



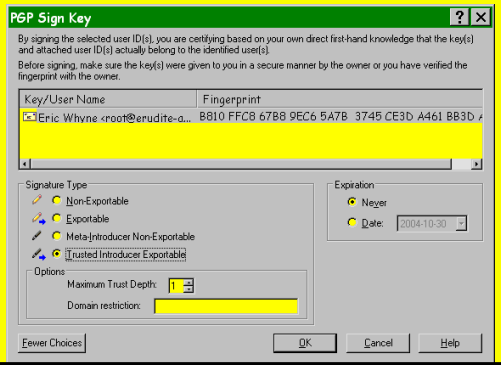
PGP: Effect of a Single-byte Change on Decryption

Changed a C to X in this position

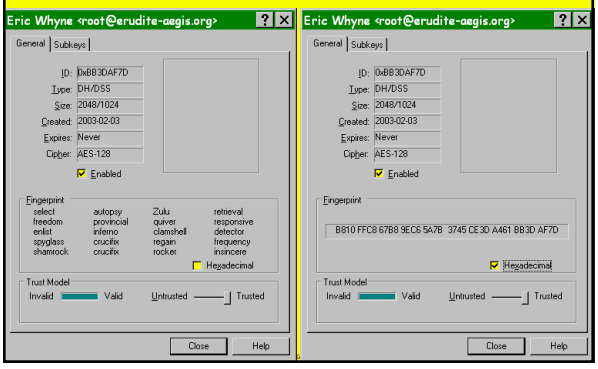
```
-----BEGIN PGP MESSAGE-----
Version: PGP Personal Privacy 6.5.8
Comment: Digital signatures increase security for everyone.

hQIOA8BxvYx90+tvEAf/eXW8l9vY45PwN27gd4kzg0DyoCJiucXXA4eDtzHqBUJm
nWadsATpcspqQNaZwgp8d5FwxRhafOtG2WqSXlXSgsFUljraYgXdlRBm2qeU/dKh
8+NfNZv2ol5bPaDAg9oiYDRV7KcFAMm3RfliSTRv6f72mE+Y6VPAvQmL6zjLAe0D
i/9n7Sh0UyM+71YuNeY0V+EZ88rTBa7JP4xr/GfAVKwwSHlWEhOfiwe82LMxOnSg
t8vj6amULTYp4daULqDt+qewULR4XWLhQ73zVT9578BSaUmgSnlRXvelsHj4fj9n
2+Ry7QZZL7k2XDybljAw8QfLcShDgC3n5Wlt+mWLwwgA8zJSyzTdKYtVh0kuo3CP
9kkGpkgqyqitVFhua+J4bvQrrhLXLBzCWXBl6RSdFvChJ7JNhtgzajV16L5SepDS
tHMcahPvGoE1e+uG5P80xqngbc5Sl7BY0jfsUjGBlyfVRpt+oC37l4W2ZKZ6mgRU
4yY+H2NTi28gC0SfXjzwRwS4qARSicjluUkYVuuRqSDfeK4slaV8FVs/xg+Ra7U3
HuyD8VmBi0/uO3RSsyPldbh2FqA1+raqeL2yuoUXLe8DI1CToLQ595/4s0wiNIzT
Ys2W5ZbOT+P2gjoVfNaRQIWFzntkRXBcs/Kx9R8pu+NMTdpJsip0oqiGfH2jtqk
2IUAbAMz3ev2p+NJcwEC/i8eoq7FornXxjxzB5/hQ3le0Ww+Vxk2LZVHdT07eqiz
8eN9e2XSxt4cr216MFCPOf1Wj8j3suYvX7mnJa7hu0mvJvxaIHawEq3U+r4ZczcC
e+q9YAukL9iXyRHYCrWZWqT6dj1Bynmi6hzgLfApq+270u8QwaGud1d/aOHGlhD7
+9JcSC1AkXoVZsA3ltaOMPP7frIOOZfdyFzr874mhG+Lru9sBFUy1S7h3gNQfUwx
ZEe9uGnDfNUth33VrrtMqjlvUjh9gZN8BOXdkKOk0WGkVvJDy6D8bSphSaQR+vvP
V83K4BaD24kiAJ70NLbeQXPx2H5j0HYT+4bD0RTt4RQgbeRLhqgwZfKpVdldXC13
P06+MTFliqcs+p4OJo/Mj67H6x877KWU3G7SKG4pBpgmy6KwKeUW8j9EpcKTgh+
+8/tgDZNAzcm8vnCQ9HEAAsN6KM9V0qoCiyDDA==
=llr+
-----END PGP MESSAGE-----
```


Single-byte Change & Decryption

Signing Someone's Public Key



Checking the Public Key Fingerprint



GPG: GNU Privacy Guard

From http://www.gnu.org/software/gnupg/gnupg.html :

- Implements OpenPGP Internet standard (RFC4880)
- Full interoperability with other modern encryption programs
- Meets all requirements of a standard Unix utility
- Uses DSA, ElGamal, 3DES and Twofish as encryption algorithms and more
- Easy implementation of new algorithms using extension modules
- Portuguese, French, German, Italian, Polish, Russian and Spanish language support
- Online help system
- Optional anonymous message receivers
- Integrated support for HKP keyservers
- Runs on most Unix platforms and support for other platforms is coming soon

86

Now go and study

87