CS 5602 Lec 02 Introduction to Cryptography (Cryptology) II

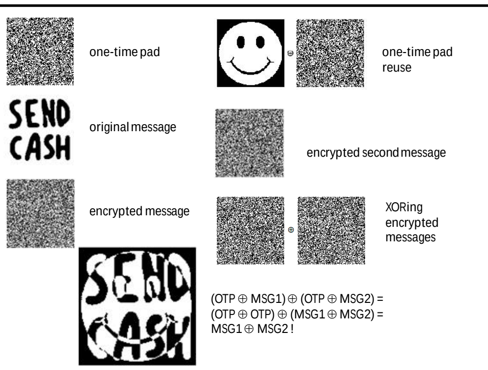
George Markowsky
Computer Science Department
Missouri University of Science and Technology

1

One-Time Pad Reuse

- **Don't Do It!**
- Why not?
- The following graphical example comes from
- https://cryptosmith.com/2008/05/31/stream-reuse/

2



3

Problems with Keys

- Distribution
- Updating
- Security
- Distribution
- Updating
- Security
- How can the public use cryptography?

4

Venona Project – Wikipedia

The Venona project (1943–80) was a counter-intelligence program initiated by the U.S. Army's Signal Intelligence Service (later the National Security Agency). The purpose of the Venona project was the decryption of messages transmitted by the intelligence agencies of the Soviet Union, e.g. the NKVD, the KGB (foreign intelligence) and the GRU (military intelligence). During the 37-year duration of the Venona project, the Signal Intelligence Service decrypted and translated approximately 3,000 messages from Russian to English; among the signals-intelligence yielded was discovery of the Cambridge Five espionage ring in Britain and Soviet espionage of the Manhattan Project in the U.S. The Venona project remained secret for more than fifteen years after it concluded, and some of the decoded Soviet messages were not declassified and published until 1995.

5

Venona Project – Wikipedia

This message traffic, which was encrypted with a one-time pad system, was stored and analyzed in relative secrecy by hundreds of cryptanalysts over a 40-year period starting in the early 1940s. Due to a serious blunder on the part of the Soviets, some of this traffic was vulnerable to cryptanalysis. The Soviet company that manufactured the one-time pads produced around 35,000 pages of duplicate key numbers, as a result of pressures brought about by the German advance on Moscow during World War II. The duplication—which undermines the security of a one-time system—was discovered and attempts to lessen its impact were made by sending the duplicates to widely separated users. Despite this, the reuse was detected by cryptologists in the US.

6

Venona Project – Wikipedia

The decrypted messages gave important insights into Soviet behavior in the period during which duplicate one-time pads were used. With the first break into the code, Venona revealed the existence of Soviet espionage at Los Alamos National Laboratories. Identities soon emerged of American, Canadian, Australian, and British spies in service to the Soviet government, including Klaus Fuchs, Alan Nunn May, and Donald Maclean. Others worked in Washington in the State Department, the Treasury, Office of Strategic Services, and even the White House.

The decrypts show the U.S. and other nations were targeted in major espionage campaigns by the Soviet Union as early as 1942. Among those identified are Julius and Ethel Rosenberg; Alger Hiss; Harry Dexter White, the second-highest official in the Treasury Department; Lauchlin Currie, a personal aide to Franklin Roosevelt; and Maurice Halperin, a section head in the Office of Strategic Services. CHECK ALSO NSA.GOV

7

Computers and Cryptography

- The impact of computers on cryptography has been huge
- Helped cryptanalysts
- Helped cryptographers
- Have replaced mechanical encoders!
- Everything naturally encoded in binary!

8

Contemporary Cryptography

- Up until about 1970, non-government cryptography was a mess
- We don't really know about government cryptography
- Just a collection of incompatible and not clearly understood methods
- The US government decided that it was time for some standardization

9

Block Cipher – Wikipedia

In cryptography, a block cipher is a deterministic algorithm operating on fixed-length groups of bits, called blocks, with an unvarying transformation that is specified by a symmetric key. Block ciphers operate as important elementary components in the design of many cryptographic protocols, and are widely used to implement encryption of bulk data.

The modern design of block ciphers is based on the concept of an iterated product cipher. In his seminal 1949 publication, *Communication Theory of Secrecy Systems*, Claude Shannon analyzed product ciphers and suggested them as a means of effectively improving security by combining simple operations such as substitutions and permutations. Iterated product ciphers carry out encryption in multiple rounds, each of which uses a different subkey derived from the original key. One widespread implementation of such ciphers, named a Feistel network after Horst Feistel, is notably implemented in the DES cipher. Many other realizations of block ciphers, such as the AES, are classified as substitution-permutation networks.

10

DES

- *Data Encryption Standard* -- came from IBM
- Certified by NBS (NIST) in 1976 for 10-15 years for cryptographic protection of sensitive, but unclassified computer data
- 56 bit key -- can be broken by NSA, but not by most private concerns
- Encoding and decoding is rapid

11

Contemporary Cryptography

- In 1973 NBS (now NIST) issued a call for a new standard with the following requirements
  - high level of security
  - completely specified and "easy" to understand
  - secrecy should depend on key
  - available to all
  - adaptable
  - efficient and economical
  - exportable
  - verifiable

12

DES

- Two rounds of request were put forward
- In 1975, IBM's algorithm based on Horst Feistel's *Lucifer was* selected to become *DES* (data encryption standard)
- Was certified only for 15 years
- Key length was limited to 56 bits by NBS (originally proposed to be 112 bits)

13

DES

- Has been *decertified*
- In the public domain
- Very widely used
- A secret key system
- Replaced by AES (Advanced Encryption Standard)

14

Public vs. Private Concerns

- The Federal Government does not want criminals to have strong encryption, so it wants to withhold it from the general public
- Forbids export of cryptographic software and systems
- Believe that only people in the US can write programs!
- OK to publish research!

15

Public vs. Private Concerns

- The *Clipper Chip*
- Various sorts of key escrow systems
- *Salt II* treaties prohibited encryption to deny telemetry data
- PGP -- Pretty Good Privacy
- Public key cryptosystems have laid the foundations for modern e-commerce

16

Public Key Cryptography

- Asymmetric -- uses two keys
  - *Public key for encryption*
  - *Private key for decryption*
  - *The Public and Private keys are inverses of each other – Will return to this point a bit later*
- Public key and method for using it can be broadcast over ordinary channels or posted on a website
  - Must have trusted broadcast source
    - Trojan horse attack
  - Public key does not provide information about private key

17

Public Key Cryptography

- Provides means for *digital signatures*!
  - The public and private keys can change roles
    - Use the private key for encryption and the public key for decryption
    - Correct decryption shows that the owner of the private key encrypted the message -- hence the digital signature
- Proper functioning of cryptosystems is based on trust (belief in security)

18

Digital Signatures

- Will use Alice and Bob, and, when necessary, Eve

ALICE	BOB
1. Writes message	1. Decodes result with his private key
2. Encrypts copy with her private key	2. Separates message into parts
3. Encrypts combo with Bob's public key	3. Decodes attachment with Alice's public key
4. Sends result to Bob	4. If decode matches message, accepts it

19

Diffie-Hellman-Merkle

- Invented public key cryptosystems
- Did not provide the first really commercially useful system
- Anticipated by secret and unpublished work at British Intelligence
  - Anticipated by NSA?
- Based on *one-way* functions -- easy to compute, but hard to invert

20

Merkle-Hellman Public Key System

- We believe that *NP-complete* problems are intrinsically hard
- Merkle-Hellman tried to hide an easy problem in the guise of a hard problem
- Based on the *subset sum* problem:
  - *Given a sequence of positive integers $T_1,\ldots, T_k$ and a positive integer W, find a subsequence whose sum is W*

21

RSA Public Key System

- Ronald Rivest, Adi Shamir and Leonard Adleman
- Based on the problem of factoring
  - No one has found a "good" approach to solving this problem in thousands of years
  - If someone finds such a method and keeps it secret, they could really make a killing

22

G.H. Hardy

- In *The Mathematician's Apology*
  - *Real mathematics has no effects on war. No one has yet discovered any warlike purpose to be served by the theory of numbers.*
- Wars of the future will be information wars in part

23

RSA Public Key System

- Pick very large primes P and Q
- Let R = P*Q
- Let F = (P-1)*(Q-1)
- Find Y with GCD(F,Y) = 1
- Find A and B with A*Y+B*F = 1
- Publish R and Y
- To encode M (message viewed as a large number) compute
  - $E = M^Y \bmod R$
- To decode, compute
  - $E^A \bmod R$

24

Why Does RSA Work?

$E^A \bmod R =$
$M^{AY} \bmod R =$
$M^{1-BF} \bmod R =$
$M*(M^F)^{-B} \bmod R =$
$M*1^{-B} \bmod R =$
$M \bmod R$

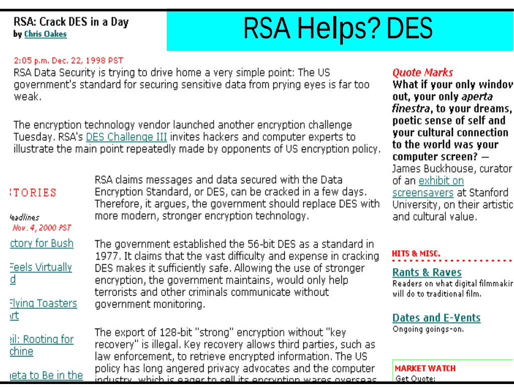
If anyone figures out how to factor efficiently, we will have a BIG problem on our hands!

25

Company sold for $200 million

26

RSA Helps? DES



27

PGP

- Created by Phil Zimmermann
- Should be usable on PCs
- RSA too resource intensive
- Uses RSA to send a secret key, and then the bulk of the message uses IDEA (like DES) with the secret key supplied by RSA
- Long series of legal actions
- Used by resistance groups around the world
- Now OpenPGP

28

Cryptography is HARD!

- Don't build your own crypto system without a LOT of study
- Get help
- Even dangerous to use "low-level packages" unless you understand what the defaults are
- Too easy to make a mistake

29

High Level View of Crytography and Other Transformations

A cryptographic method is really a *bijection (injection & surjection)*

f: PT → *CT*

(PT = PlainText, CT = CipherText)

There is always a f^{-1} *such that* $f(f^{-1}(x)) = x$ and $f^{-1}(f(y)) = y$

Typically, PT and CT are fixed blocks of the same length

30

Decrypting

- Given a cryptographic method (function) *f*, decrypting is finding f^{-1}. Once you find it, then security is completely gone

31

Data Compression

A *data compression* method is really a *bijection (injection & surjection)*

f: PT → CT

(PT = PlainText, CT = CompressedText)

There is <u>always</u> a f^{-1} *such that* $f(f^{-1}(x)) = x$ and $f^{-1}(f(y)) = y$

Typically, the length of CT is significantly less than the length of PT

32

Decompression

- Given a compression method (function) *f*, decompressing is finding f^{-1}. Once you find it, you can get the original text back

33

Encryption vs. Compression

- Encryption and compression seem to be the same thing
- What are the differences?
- Fixed length blocks vs. variable length blocks
- Most common compression algorithms are open and all the algorithms are published
- Some encryption algorithms are hidden, although some are open
- Compression is based just on the data being compressed
- Encryption is based on the data being encrypted along with some other information referred to as a key, which might or might not be public

34

A Compression Gotcha

- How can compression possibly work?
- Say you have a compression method, *f*, that takes all files with p bits into files that have k bits where p > k
- There are 2^p files with p bits and 2^k files with k bits
- Since $2^p > 2^k$, by the *Pigeonhole Principle* (what's that?) there must be at least two different files F1 and F2 such that *f*(F1) = *f*(F2)
- It is impossible to find any function *g* such that *g(f(F1)) = F1* and *g(f(F2)) = F2* because *g(f(F1)) = g(f(F2))*
- So what's the true story?
- Any compression method cannot always reduce the size of files

35

Compression Methods

- The bottom line is that any compression method must in some cases actually increase the size of the file rather than decrease it
- Why would you use such a method?
- The files that people actually produce are just a tiny, tiny, tiny fraction of all possible files of that size
- Compression methods are useful as long as they shrink the sizes of "useful" files

36

Entropy of Text

- Most things produced by humans have structure of some kind
- For example if you consider most human produced English text you will find that the letters do not occur with equal frequency
- This permits frequency analysis of simple cryptographic methods

37

Entropy of Text

- Claude Shannon, the father of information theory, introduced the *entropy function* which is very useful
- If you have a sequence of symbols a_1, a_2, a_3,..., a_n drawn from a k letter alphabet, then the entropy of the sequence is given by the formula below where p_i is the probability that letter i occurred in the string

$$-\sum_{i=1}^{k} p_i \log p_i$$

Can use logs in any base since results differ by a multiplicative constant

38

Hashing Methods

A *hashing method* is a function

f:AT → FLT

(AT = All Text, FLT = Fixed Length Text)

Why is this useful?

In general, there would be many (infinitely many in fact) files F_1, F_2, ... such that $f(F_1) = f(F_2) = \ldots$

A hashing method is useful if there is no easy way to find files that collide

39

Hash Functions

- Hash methods are generally called hash functions and are used in a variety of ways
- Of most interest to us, is their use for signatures
- How do you know that the file you downloaded is the correct file and has not been tampered with or corrupted in some way?

40

Hash Functions

- You compute the *hash* of what you downloaded and compare it to a reliable source of what it should be
- If they are the same you have some confidence that you have the correct file

41

MD5 hashes

The 128-bit (16-byte) MD5 hashes (also termed *message digests*) are typically represented as following demonstrates a 43-byte ASCII input and the corresponding MD5 hash:

```
MD5("The quick brown fox jumps over the lazy dog")
 = 9e107d9d372bb6826bd81d3542a419d6
```

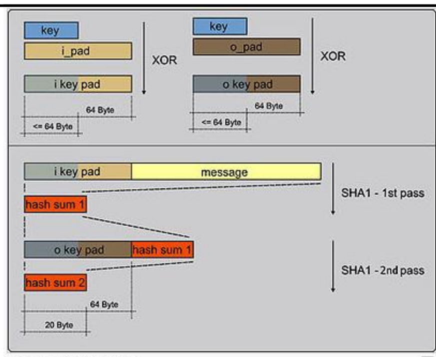
Even a small change in the message will (with overwhelming probability) result in a completely example, adding a period to the end of the sentence:

```
MD5("The quick brown fox jumps over the lazy dog.")
 = e4d909c290d0fb1ca068ffaddf22cbd0
```

The hash of the zero-length string is:

```
MD5("")
 = d41d8cd98f00b204e9800998ecf8427e
```

42



SHA-1 HMAC Generation.

43

```
Note 1: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating
Note 2: All constants in this pseudo code are in big endian.
        Within each word, the most significant byte is stored in the leftmost byte position

Initialize variables:
h0 = 0x67452301
h1 = 0xEFCDAB89
h2 = 0x98BADCFE
h3 = 0x10325476
h4 = 0xC3D2E1F0

Pre-processing:
append the bit '1' to the message
append 0 ≤ k < 512 bits '0', so that the resulting message length (in bits)
   is congruent to 448 ≡ −64 (mod 512)
append length of message (before pre-processing), in bits, as 64-bit big-endian integer

Process the message in successive 512-bit chunks:
break message into 512-bit chunks
for each chunk
    break chunk into sixteen 32-bit big-endian words w[i], 0 ≤ i ≤ 15

    Extend the sixteen 32-bit words into eighty 32-bit words:
    for i from 16 to 79
        w[i] = (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16]) leftrotate 1

    Initialize hash value for this chunk:
    a = h0
    b = h1
    c = h2
    d = h3
    e = h4
```

44

```
    Main loop:
    for i from 0 to 79
        if 0 ≤ i ≤ 19 then
            f = (b and c) or ((not b) and d)
            k = 0x5A827999
        else if 20 ≤ i ≤ 39
            f = b xor c xor d
            k = 0x6ED9EBA1
        else if 40 ≤ i ≤ 59
            f = (b and c) or (b and d) or (c and d)
            k = 0x8F1BBCDC
        else if 60 ≤ i ≤ 79
            f = b xor c xor d
            k = 0xCA62C1D6

        temp = (a leftrotate 5) + f + e + k + w[i]
        e = d
        d = c
        c = b leftrotate 30
        b = a
        a = temp

    Add this chunk's hash to result so far:
    h0 = h0 + a
    h1 = h1 + b
    h2 = h2 + c
    h3 = h3 + d
    h4 = h4 + e

Produce the final hash value (big-endian):
digest = hash = h0 append h1 append h2 append h3 append h4
```

45

```
(0  ≤ i ≤ 19): f = d xor (b and (c xor d))                (alternative 1)
(0  ≤ i ≤ 19): f = (b and c) xor ((not b) and d)          (alternative 2)
(0  ≤ i ≤ 19): f = (b and c) + ((not b) and d)            (alternative 3)
(0  ≤ i ≤ 19): f = vec_sel(d, c, b)                       (alternative 4)

(40 ≤ i ≤ 59): f = (b and c) or (d and (b or c))          (alternative 1)
(40 ≤ i ≤ 59): f = (b and c) or (d and (b xor c))         (alternative 2)
(40 ≤ i ≤ 59): f = (b and c) + (d and (b xor c))          (alternative 3)
(40 ≤ i ≤ 59): f = (b and c) xor (b and d) xor (c and d)  (alternative 4)
```

46

Example hashes

Main article: Examples of SHA digests

The following is an example of SHA-1 digests. ASCII encoding is assumed for all messages.

```
SHA1("The quick brown fox jumps over the lazy dog")
= 2fd4e1c6 7a2d28fc ed849ee1 bb76e739 1b93eb12
```

Even a small change in the message will, with overwhelming probability, result in a completely c
example, changing dog to cog produces a hash with different values for 81 of the 160 bits:

```
SHA1("The quick brown fox jumps over the lazy cog")
= de9f2c7f d25e1b3a fad3e85a 0bd17d9b 100db4b3
```

47

http://www.h-online.com/security/news/item/NIST-certified-USB-Flash-drives-with-hardware-encryption-cracked-895308.html

4 January 2010, 18:34

NIST-certified USB Flash drives with hardware encryption cracked

Encrypting USB Flash memory from Kingston, SanDisk and Verbatim Kingston, SanDisk and Verbatim all sell quite similar USB Flash drives with AES 256-bit hardware encryption that supposedly meet the highest security standards. This is emphasised by the FIPS 140-2 Level 2 certificate issued by the US National Institute of Standards and Technology (NIST), which validates the USB drives for use with sensitive government data. Security firm SySS, however, has found that despite this it is relatively easy to access the unencrypted data, even without the required password.

48

Details of DES

- Lots of splitting of blocks, XORing, permuting, etc
- All steps must be reversible
- The following diagrams give some insight into its operation
- Clearly a step above previous commercial ciphers

49

FIPS PUB 46-3

FEDERAL INFORMATION
PROCESSING STANDARDS PUBLICATION

Reaffirmed
1999 October 25

U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology

50

Enciphering

A sketch of the enciphering computation is given in **Figure 1**.

The 64 bits of the input block to be enciphered are first subjected to the following permutation, called the initial permutation ***IP***:

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

51

That is the permuted input has bit 58 of the input as its first bit, bit 50 as its second bit, and so on with bit 7 as its last bit. The permuted input block is then the input to a complex key-dependent computation described below. The output of that computation, called the preoutput, is then subjected to the following permutation which is the inverse of the initial permutation:

$\underline{IP^{-1}}$

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

That is, the output of the algorithm has bit 40 of the preoutput block as its first bit, bit 8 as its second bit, and so on, until bit 25 of the preoutput block is the last bit of the output.

52

Let ***E*** denote a function which takes a block of 32 bits as input and yields a block of 48 bits as output. Let ***E*** be such that the 48 bits of its output, written as 8 blocks of 6 bits each, are obtained by selecting the bits in its inputs in order according to the following table:

E BIT-SELECTION TABLE

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

53

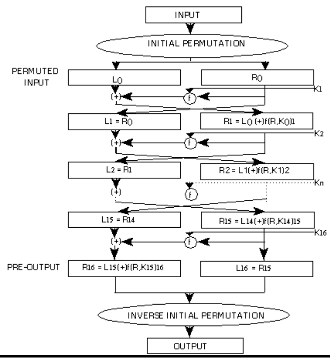
Each of the unique selection functions $S_1, S_2, \ldots, S_8$, takes a 6-bit block as input and yields a 4-bit block as output and is illustrated by using a table containing the recommended S_1:

$\underline{S_1}$

Column Number

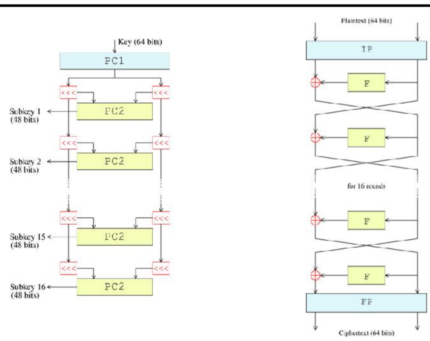
Row No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

54

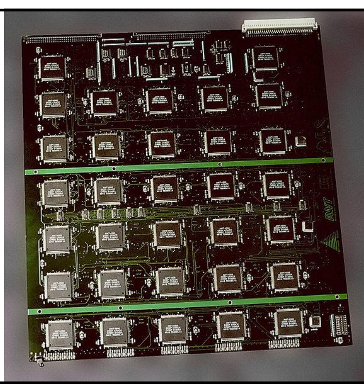


16 Rounds

55



56



The EFF's US$250,000 DES cracking machine contained 1,856 custom chips and could brute force a DES key in a matter of days — the photo shows a DES Cracker circuit board fitted with several Deep Crack chips.

57

Bruce Schneier

The Blowfish Encryption Algorithm -- One Year Later

B. Schneier

Dr. Dobb's Journal, September 1995.

DES is the workhorse of cryptography algorithms, and it's long past time to replace the 19-year-old standard. The recent design of a $1M machine that could recover a DES key in 3.5 hours only confirmed what everybody knew: DES's key size is far too small for today.

The world only partly trusted DES because it survived the scrutiny of the NSA. Experts trusted DES because it was a published standard, and because it survived 20 years of intensive cryptanalysis by cryptographers around the world. Cryptography is like that: confidence in an algorithm grows as group after group tries to break it and fails.

Candidates for a replacement are emerging, but none has taken widespread hold. Triple-DES is the conservative approach; IDEA (used in PGP) is the most promising new algorithm. And there is a bevy of unpatented also-rans: RC4 (once a trade secret of RSA Data Security, Inc. but now publicly available on the Internet), SAFER, and my own Blowfish.

I first presented Blowfish at the Cambridge Algorithms Workshop ("Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish)," *Fast Software Encryption*, R. Anderson, ed., Lecture Notes in Computer Science #809, Springer-Verlag, 1994) and in *Dr. Dobb's Journal* (April 1994). From the start Blowfish was intended to be a completely free--unpatented, unlicensed, and uncopyrighted--alternative to DES. Since then it has been analyzed by some people and has started to see use in some systems, both public and private. This article presents new Blowfish code, as well as updates on the algorithm's security.

58

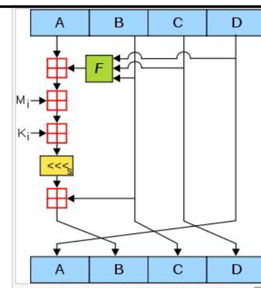


Figure 1. One MD5 operation. MD5 consists of 64 of these operations, grouped in four rounds of 16 operations. F is a nonlinear function; one function is used in each round. M_i denotes a 32-bit block of the message input, and K_i denotes a 32-bit constant, different for each operation. $\lll_s$ denotes a left bit rotation by s places; s varies for each operation. $\boxplus$ denotes addition modulo 2^{32}.

59

```
//Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating
var int[64] r, k

//r specifies the per-round shift amounts
r[ 0..15] := {7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22}
r[16..31] := {5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20}
r[32..47] := {4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23}
r[48..63] := {6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21}

//Use binary integer part of the sines of integers (Radians) as constants:
for i from 0 to 63
    k[i] := floor(abs(sin(i + 1)) × (2 pow 32))

//Initialize variables:
var int h0 := 0x67452301
var int h1 := 0xEFCDAB89
var int h2 := 0x98BADCFE
var int h3 := 0x10325476

//Pre-processing:
append "1" bit to message
append "0" bits until message length in bits ≡ 448 (mod 512)
append bit /* bit, not byte */ length of unpadded message as 64-bit little-endian integer to message
```

60

```
//Process the message in successive 512-bit chunks:
for each 512-bit chunk of message
    break chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15

    //Initialize hash value for this chunk:
    var int a := h0
    var int b := h1
    var int c := h2
    var int d := h3

    //Main loop:
    for i from 0 to 63
        if 0 ≤ i ≤ 15 then
            f := (b and c) or ((not b) and d)
            g := i
        else if 16 ≤ i ≤ 31
            f := (d and b) or ((not d) and c)
            g := (5×i + 1) mod 16
        else if 32 ≤ i ≤ 47
            f := b xor c xor d
            g := (3×i + 5) mod 16
        else if 48 ≤ i ≤ 63
            f := c xor (b or (not d))
            g := (7×i) mod 16

        temp := d
        d := c
        c := b
        b := b + leftrotate((a + f + k[i] + w[g]) , r[i])
        a := temp

    //Add this chunk's hash to result so far:
    h0 := h0 + a
    h1 := h1 + b
    h2 := h2 + c
    h3 := h3 + d

var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)
```

61

```
  //leftrotate function definition
  leftrotate (x, c)
      return (x << c) or (x >> (32-c));
```

Note: Instead of the formulation from the original RFC 1321 shown, the following may be used for improved efficiency (useful if assembly language is being used - otherwise, the compiler will generally optimize the above code. Since each computation is dependent on another in these formulations, this is often slower than the above method where the nand/and can be parallelised):

```
(0  ≤ i ≤ 15): f := d xor (b and (c xor d))
(16 ≤ i ≤ 31): f := c xor (d and (b xor c))
```

62

Bruce Schneier

Cryptanalysis of MD5 and SHA: Time for a New Standard

By Bruce Schneier
Computerworld
August 19, 2004

At the Crypto 2004 conference in Santa Barbara, Calif., this week, researchers announced several weaknesses in common hash functions. These results, while mathematically significant, aren't cause for alarm. But even so, it's probably time for the cryptography community to get together and create a new hash standard.

One-way hash functions are a cryptographic construct used in many applications. They are used with public-key algorithms for both encryption and digital signatures. They are used in integrity checking. They are used in authentication. They have all sorts of applications in a great many different protocols. Much more than encryption algorithms, one-way hash functions are the workhorses of modern cryptography.

In 1990, Ron Rivest invented the hash function MD4. In 1992, he improved on MD4 and developed another hash function: MD5. In 1993, the National Security Agency published a hash function very similar to MD5, called the Secure Hash Algorithm (SHA). Then in 1995, citing a newly discovered weakness that it refused to elaborate on, the NSA made a change to SHA. The new algorithm was called SHA-1. Today, the most popular hash function is SHA-1, with MD5 still being used in older applications.

One-way hash functions are supposed to have two properties. One, they're one-way. This means that it's easy to take a message and compute the hash value, but it's impossible to take a hash value and re-create the original message. (By "impossible," I mean "can't be done in any reasonable amount of time.") Two, they're collision-free. This means that it's impossible to find two messages that hash to the same hash value. The cryptographic reasoning behind these two properties is subtle, and I invite curious readers to learn more in my book Applied Cryptography.

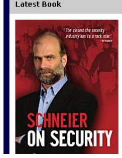
Breaking a hash function means showing that either -- or both -- of those properties aren't true. Cryptanalysis of the MD4 family of hash functions has proceeded in fits and starts over the past decade or so, with results against simplified versions of the algorithms and partial results against the whole algorithms.

This year, Eli Biham and Rafi Chen, and separately Antoine Joux, announced some pretty impressive cryptographic results against MD5 and SHA. Collisions have been demonstrated in SHA. And there are rumors, unconfirmed at this writing, of results against SHA-1.

Search
blog only
essays and op eds only
whole site
Search

Crypto-Gram Newsletter
A free monthly e-mail newsletter on security and security technology.
read more

Latest Book



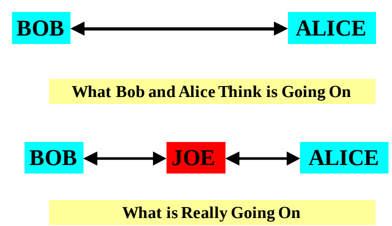
63

MITM Attack—Wikipedia

- In cryptography, a man-in-the-middle attack (MITM) is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised.
- The attacker must be able to observe and intercept messages going between the two victims.

64

The Man-in-the-Middle Attack



65

MITM Example with Public Key Cryptography System

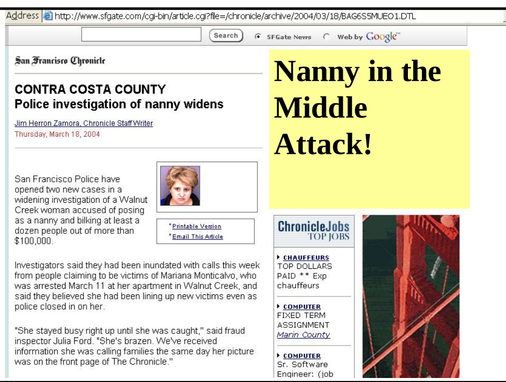
- Alice wants to communicate with Bob, but Joe wants to eavesdrop
- Alice must ask Bob for his public key, which Bob sends, BUT which Joe intercepts
- Joe substitutes his public key and sends it to Alice
- Alice encrypts the message using Joe's public key and Joe intercepts it

66

MITM Example with Public Key Cryptography System

- Joe decrypts the message using his private key and stores it
- He then encrypts the message using Bob's public key and sends it to Bob along with his public key so that Bob thinks that he has Alice's public key
- Joe can sit in the middle and read all messages, while Bob and Alice think they are secure!

67


Address http://www.sfgate.com/cgi-bin/article.cgi?file=/chronicle/archive/2004/03/18/BAG6S5MUEO1.DTL

Search SFGate News Web by Google

San Francisco Chronicle

CONTRA COSTA COUNTY
Police investigation of nanny widens

Jim Herron Zamora, Chronicle Staff Writer
Thursday, March 18, 2004

San Francisco Police have opened two new cases in a widening investigation of a Walnut Creek woman accused of posing as a nanny and bilking at least a dozen people out of more than $100,000.

Printable Version
Email This Article

Investigators said they had been inundated with calls this week from people claiming to be victims of Mariana Monticalvo, who was arrested March 11 at her apartment in Walnut Creek, and said they believed she had been lining up new victims even as police closed in on her.

"She stayed busy right up until she was caught," said fraud inspector Julia Ford. "She's brazen. We've received information she was calling families the same day her picture was on the front page of The Chronicle."

ChronicleJobs TOP JOBS
CHAUFFEURS TOP DOLLARS PAID ** Exp chauffeurs
COMPUTER FIXED TERM ASSIGNMENT Marin County
COMPUTER Sr. Software Engineer: (job

Nanny in the Middle Attack!

68

San Francisco Police have opened two new cases in a widening investigation of a Walnut Creek woman accused of posing as a nanny and bilking at least a dozen people out of more than $100,000.

Investigators said they had been inundated with calls this week from people claiming to be victims of Mariana Monticalvo, who was arrested March 11 at her apartment in Walnut Creek, and said they believed she had been lining up new victims even as police closed in on her.


Walnut Creek Police Department

69

"She stayed busy right up until she was caught," said fraud inspector Julia Ford. "She's brazen. We've received information she was calling families the same day her picture was on the front page of The Chronicle."

Monticalvo, who was living under the name Michelle Carrick when arrested, is scheduled to appear in Contra Costa County Superior Court today to enter pleas to nine counts of burglary, grand theft and check fraud. The charges, filed last week, stem from the alleged theft in 2000 of $36,000 in traveler's checks from an Orinda family and the passing of several bad checks in Danville in December and January.

Prosecutors filed additional charges this week accusing Monticalvo of resisting arrest and assaulting a police officer with a deadly weapon. The charges stem from a March 3 incident outside a Kinko's in Lafayette, where Monticalvo allegedly ignored a police officer's order to stop and grazed his leg with her car as she fled.

70

In the week before her arrest, San Francisco police said, Monticalvo is believed to have stolen the identity of a legitimate nanny and then used it to bilk $2,500 from one victim. She also is believed to have been targeting new marks using advertisements expectant mothers posted on the online bulletin board Craigslist.

Ford, the fraud inspector handling the latest case, said Monticalvo had used the Internet to steal and assume the identity of Angela Ginette Jordan, a nanny who lives in Daly City. Investigators have said **Monticalvo often used the online bulletin board Craigslist to pose as an expectant mother needing a nanny, then used the resumes and references of applicants to assume their identity.**

71

Jordan said she was flabbergasted to learn Friday from police that a woman using her name had allegedly fleeced a pregnant San Francisco woman and arranged to meet several others.

"I assumed it was just another parent looking for a nanny, so I sent her my resume and driver's license number," Jordan said. "She used my references to back up her rip-off scam. I've been a nanny for six years. Now, everyone is going to think I am a thief. If I my record isn't spotless, no one will hire me. This is a nightmare."

72

Investigators suspect Monticalvo may have also used the name Michelle Carrick in recent weeks to search for potential victims in Contra Costa and Solano counties. Investigations also are underway in Hillsborough, San Mateo, Vallejo and San Anselmo, and Ford said people in Palo Alto, Walnut Creek, San Jose and Hayward had told her they thought Monticalvo might have swindled them.

"We've had a flood of calls," Ford said. "I wish I could help all of these people, but I told them they need to make a report to their local police department.

"Hopefully, she will get punished for every person she swindled. But it's going to take a while to unravel her web."

73

The Need For An Additional Transfer Over A Secure Channel

- With the exception of the Interlock Protocol, all cryptographic systems that are secure against MITM attacks require an additional exchange or transmission of information over some kind of secure channel.
- Many key agreement methods with different security requirements for the secure channel have been developed.

74

Cryptography in the Quantum Age!

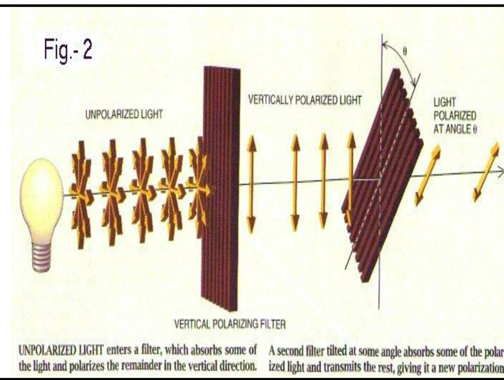
- Quantum mechanics appears odd to most people (everybody?)
- Nearly a century old, but still relatively unknown
- Will summarize some key ideas from QM that are of interest in computer science and cryptography
- Could have vast consequences for our field

75

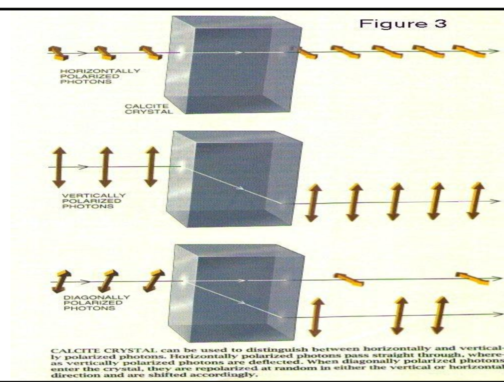
Brief Survey of QM

- The *Uncertainty Principle* states that it is impossible to have perfect knowledge of any particle or system
- The act of measurement, disrupts the system
- Will keep knowledge required to a minimum
- Have 4 types of photons

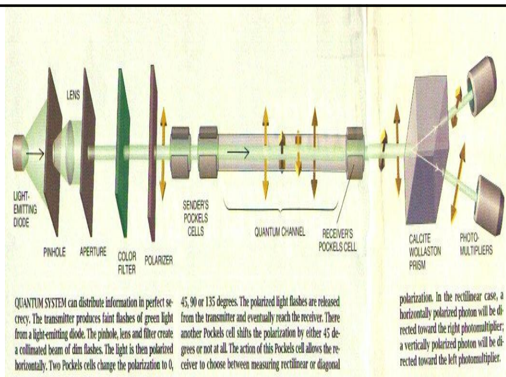
76



77



78

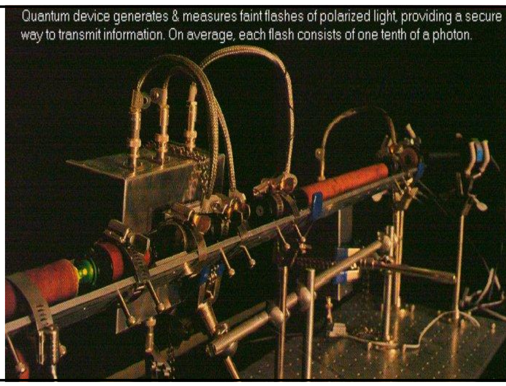


79

Some Properties of Photons

- If photons go through an incorrect filter all knowledge of their former state is forgotten!
- We can generate the types of photons that we want
- Can use random sequences of these photons to transmit a one time pad and to also detect eavesdropping
- Believe by laws of QM, to be unbreakable

82



80

Quantum Cryptography Without Eavesdropping

- Alice send Bob a random sequence of the 4 types of photons
- Bob picks one of two filters randomly and makes a measurement
- Alice then calls Bob and tells him for each photon what the correct filter should have been
- Alice and Bob can talk over an unsecured channel -- trust important
- Bob discards all missing photons and all photons measured with the wrong filter
- Gets a binary sequence using 0° = 0, 90° = 1, 45° = 0, and 135° = 1

83

Some Properties of Photons

- 0° and 90° photons are called *rectilinear photons*
- 45° and 135° photons are called *diagonal photons*
- Have *rectilinear* and *diagonal* filters
- The "correct" filter applied to a photon will almost always give the correct answer (it might fail to detect it)
- The "incorrect" filter will randomly convert to one of the other types of photons

81

Quantum Cryptography with Eavesdropping

- To detect eavesdropping, publicly publish what a fair number of bits should have been, say 100 bits
- If someone was trying to read the transmission, a bunch of these should be wrong since eavesdropper would have wrong guesses
- If all 100 bits are correct, use the remaining bits

84

Problems of Quantum Cryptography

- Need single photons so adversary can't pick them off
- Weak signals have limited distances
  - Some limited tests have been done with fiber and in air
- Amplification difficult
- Relatively slow
- Easily suppressed
- Just the beginnings, so stick around
- Might be limitations of physical laws at some point

85

Quantum Eavesdropping?

- This student is working on a thesis in Norway on quantum eavesdropping strategies
- Not fully analyzed yet

86

Quantum Computing

- The possibility exists of using quantum mechanical states for computing
- Schrodinger's Cat can be both dead and alive
- Unmeasured particles can be in a variety of states simultaneously
- Potentially a huge number of operations possible simultaneously

87

Quantum Computing

- If possible, can be used to solve NP complete problems, factoring, decryption, etc.
- True natural, parallel computation
- A variety of projects underway
- Of great military significance
- The full picture is not available to us at this time
- Not built yet!
- Are there quantum processes in the brain?

88

QCL - A Programming Language for Quantum Computers

Current version: qcl-0.4.0

Despite many common concepts with classical computer science, quantum computing is still widely considered as a special discipline within the broad field of theoretical physics. One reason for the slow adoption of QC by the computer science community is the confusing variety of formalisms (Dirac notation, matrices, gates, operators, etc.), none of which has any similarity with classical programming languages, as well as the rather ``physical'' terminology in most of the available literature.

QCL (Quantum Computation Language) tries to fill this gap: QCL is a high level, architecture independent programming language for quantum computers, with a syntax derived from classical procedural languages like C or Pascal. This allows for the complete implementation and simulation of quantum algorithms (including classical components) in one consistent formalism.

Download

The current version of QCL is 0.4.0:

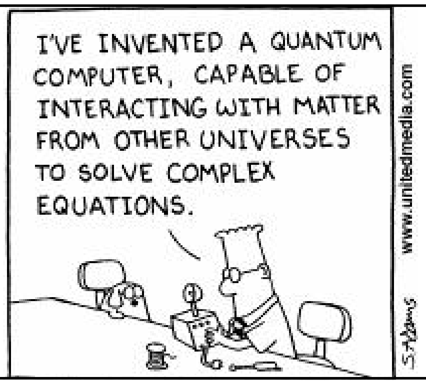
- Source Distriution: qcl-0.4.0
- Binary Distriution: qcl-0.4.0-bin *(ELF, i386, Linux 2.2, glibc2.1)*

Except for local array variables, QCL is now completely ANSI C++ complient (at least gcc says so). It is source and save-file compatible with all

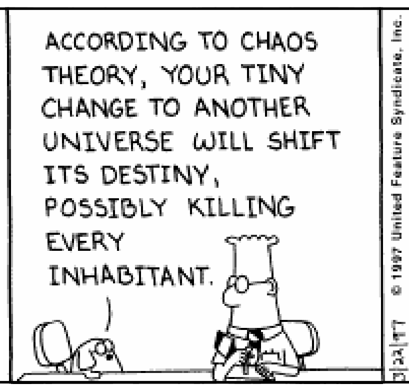
89

Dilbert Does Quantum Computing

90



91

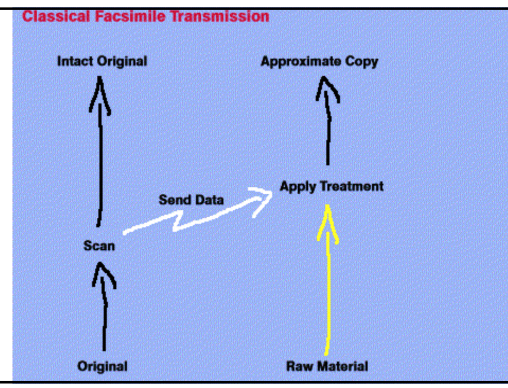


92

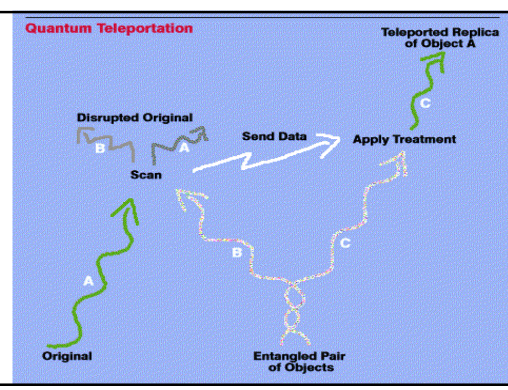
93

And Now
For
Something
Even More
Completely
DIFFERENT

94



95

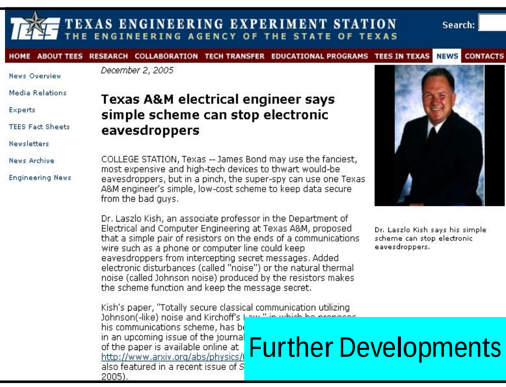


96

Some Observations

- This is an exciting time in computer science and cryptography
- This is truly the age of information and knowledge
- It is not clear that there is such a thing as "useless" information
- Learn as much as you can about computer science, mathematics and science -- it will be quite useful in the future

97



TEXAS ENGINEERING EXPERIMENT STATION
THE ENGINEERING AGENCY OF THE STATE OF TEXAS

Search:

HOME ABOUT TEES RESEARCH COLLABORATION TECH TRANSFER EDUCATIONAL PROGRAMS TEES IN TEXAS NEWS CONTACTS

News Overview
Media Relations
Experts
TEES Fact Sheets
Newsletters
News Archive
Engineering News

December 2, 2005

Texas A&M electrical engineer says simple scheme can stop electronic eavesdroppers

COLLEGE STATION, Texas -- James Bond may use the fanciest, most expensive and high-tech devices to thwart would-be eavesdroppers, but in a pinch, the super-spy can use one Texas A&M engineer's simple, low-cost scheme to keep data secure from the bad guys.

Dr. Laszlo Kish, an associate professor in the Department of Electrical and Computer Engineering at Texas A&M, proposed that a simple pair of resistors on the ends of a communications wire such as a phone or computer line could keep eavesdroppers from intercepting secret messages. Added electronic disturbances (called "noise") or the natural thermal noise (called Johnson noise) produced by the resistors makes the scheme function and keep the message secret.

Kish's paper, "Totally secure classical communication utilizing Johnson(-like) noise and Kirchoff's ... his communications scheme, has be... in an upcoming issue of the journal... of the paper is available online at http://www.arxiv.org/abs/physics/... also featured in a recent issue of S... 2005).

Dr. Laszlo Kish says his simple scheme can stop electronic eavesdroppers.

Further Developments

98