A Very Brief History of Encryption
Part 2 Electronic Computer Age

Larry Wittig

Lexington Computer and Technology Group

March 2012

Overview of this presentation

This presentation mostly follows The Code Book by Simon Singh:

- Simple translation and substitution encryption
- Vigenere (polyalphabetic) encryption to Enigma (WWII)

The older stuff

- The advent of electronic computers and the Arpanet/Internet
- Lucifer, DES and AES
- Diffie-Hellman Key Exchange
- RSA (public key – private key stuff)
- Pretty Good Privacy (PGP)
- SSL & TLS (the **s** in http**s**)

The newer stuff – since the the advent of electronic computers

Besides The Code Book I also consulted Wikipedia (quite often) and other Web pages. Generally I give links to my source material.

Discussed in Singh's book but not covered here in any detail:

- Mary Queen of Scotts vs. QE1
- Beale treasure papers
- Deciphering lost languages and ancient scripts
- Navajo code talkers during WW II
- Quantum computers

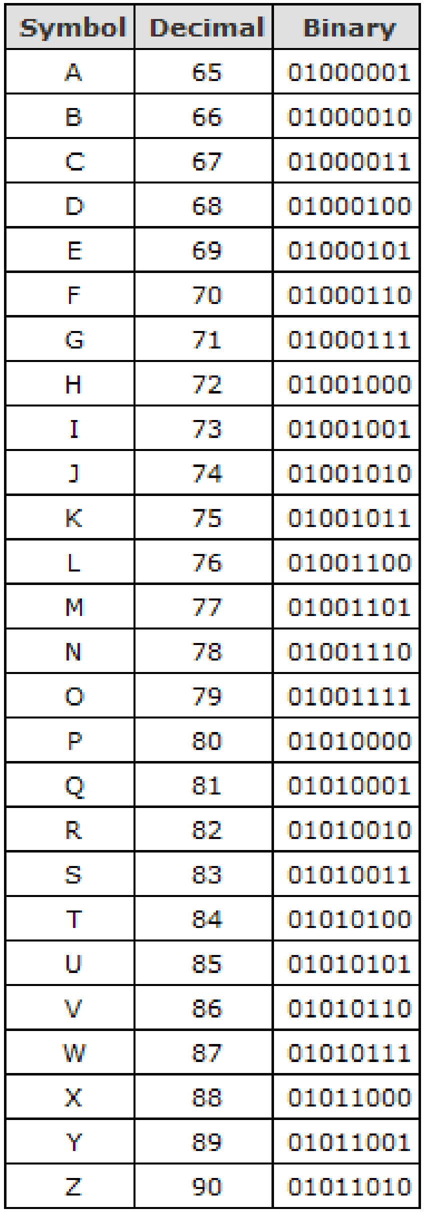
History of encryption post WWII

Electronic computers drastically changed what could be done:

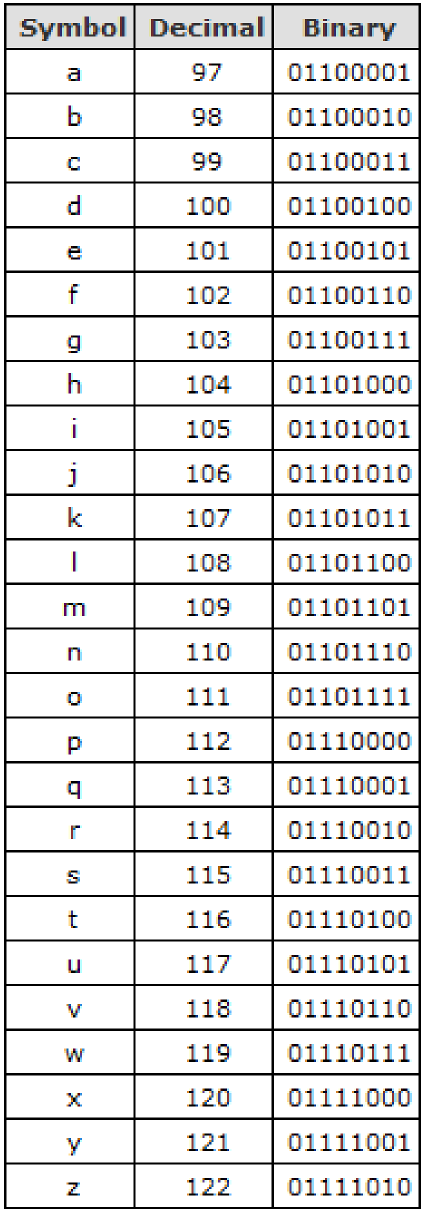
- IBM realized that to automate private business it needed some form of encryption and developed **Lucifer.** Under NBS Lucifer became **DES** (Data Encryption Standard) in 1976 and this was superseded by **AES** (Advanced Encryption Standard) under NIST in 2001.
- With the start of the Arpanet/Internet it became clear that private encryption was important if not necessary.
- In the early 70's this has lead to **Diffie-Hellman Key Exchanges (DHE)** and subsequently what is often referred to **public key** or **RSA** encryption. Versions of these plus AES are at the heart of today's secure communication on the Internet.
- There has been an ongoing battle between private encryption concerns and **NSA** (National Security Agency, a.k.a. No Such Agency) NSA wants to limit the strength of private encryption software.

Enter Electronic Computers

- What could be done mechanically both in terms of the number of operations and the quickness for both encrypting and decrypting increased by orders of magnitude
- The first change was to use a binary number system (See: http://www.kerryr.net/pioneers/ascii2.htm This is an Australian site, written in Australian English. Any re-semblance to American or Euro English is purely coincidental.)
- When encrypting in ACSI, the code representing one letter could even be cut in half.



Symbol	Decimal	Binary
A	65	01000001
B	66	01000010
C	67	01000011
D	68	01000100
E	69	01000101
F	70	01000110
G	71	01000111
H	72	01001000
I	73	01001001
J	74	01001010
K	75	01001011
L	76	01001100
M	77	01001101
N	78	01001110
O	79	01001111
P	80	01010000
Q	81	01010001
R	82	01010010
S	83	01010011
T	84	01010100
U	85	01010101
V	86	01010110
W	87	01010111
X	88	01011000
Y	89	01011001
Z	90	01011010



Symbol	Decimal	Binary
a	97	01100001
b	98	01100010
c	99	01100011
d	100	01100100
e	101	01100101
f	102	01100110
g	103	01100111
h	104	01101000
i	105	01101001
j	106	01101010
k	107	01101011
l	108	01101100
m	109	01101101
n	110	01101110
o	111	01101111
p	112	01110000
q	113	01110001
r	114	01110010
s	115	01110011
t	116	01110100
u	117	01110101
v	118	01110110
w	119	01110111
x	120	01111000
y	121	01111001
z	122	01111010

ASCII conversion

Be sure to drink your Ovaltine.

in 8-bit ASCII becomes

01000010 01100101 00100000 01110011 01110101 01110010 01100101 00100000
01110100 01101111 00100000 01100100 01110010 01101001 01101110 01101011
00100000 01111001 01101111 01110101 01110010 00100000 01001111 01110110
01100001 01101100 01110100 01101001 01101110 01100101 0101110

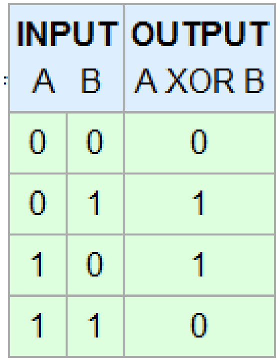
- This is a short sentence, imagine the number generated by a few paragraphs of plaintext.
- I used the following web site to do this conversion http://www.roubaixinteractive.com/PlayGround/Binary_Conversion/Binary_To_Text.asp

Electronic computer encryption
The turning point

- In the early 70's Horst Feistel et.al at IBM developed **Lucifer** which the NBS turned into **DES** (Data Encryption Standard) in 1975. It evolved into Triple DES. In **2001** was superseded by the **AES** (Advanced Encryption Standard) by NTIS (formerly NBS).
- There is a sorted history of **NSA** trying to keep public encryption standards from becoming too effective – they wanted to keep them at the point where only they could break them. Mostly this was a fight over keyword lengths.
- The Lucifer, DES and AES algorithms are block ciphers, that is they **work on blocks of bits of data by cutting, shifting, and substitution**.
- **Simplified example** of a DES-type encryption: http://www.usafa.edu/df/dfe/dfer/centers/accr/tools/DESCipherApplet.html
- With DES/AES it is still necessary to exchange a secret Keyword.

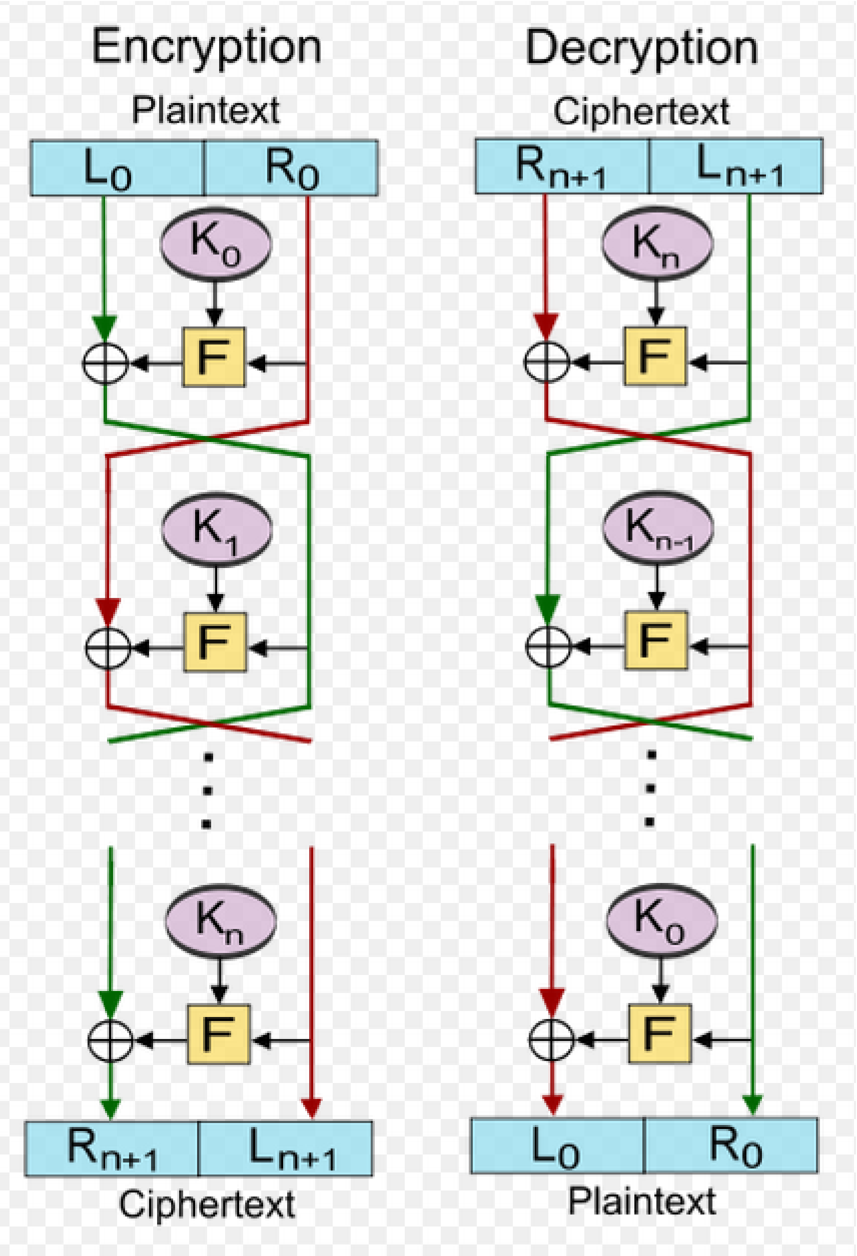
How DES Works

- DES is what is known as a block cypher – it breaks the plaintext into blocks of 64 bits.
- Divides each block in half and runs it through the algorithm depicted here.
- The full key is used to generate 16 sub-keys.
- At $\boxed{\text{F}}$ half of the data is replaced with values from a lookup table, and then combined with sub-key values using the XOR function.

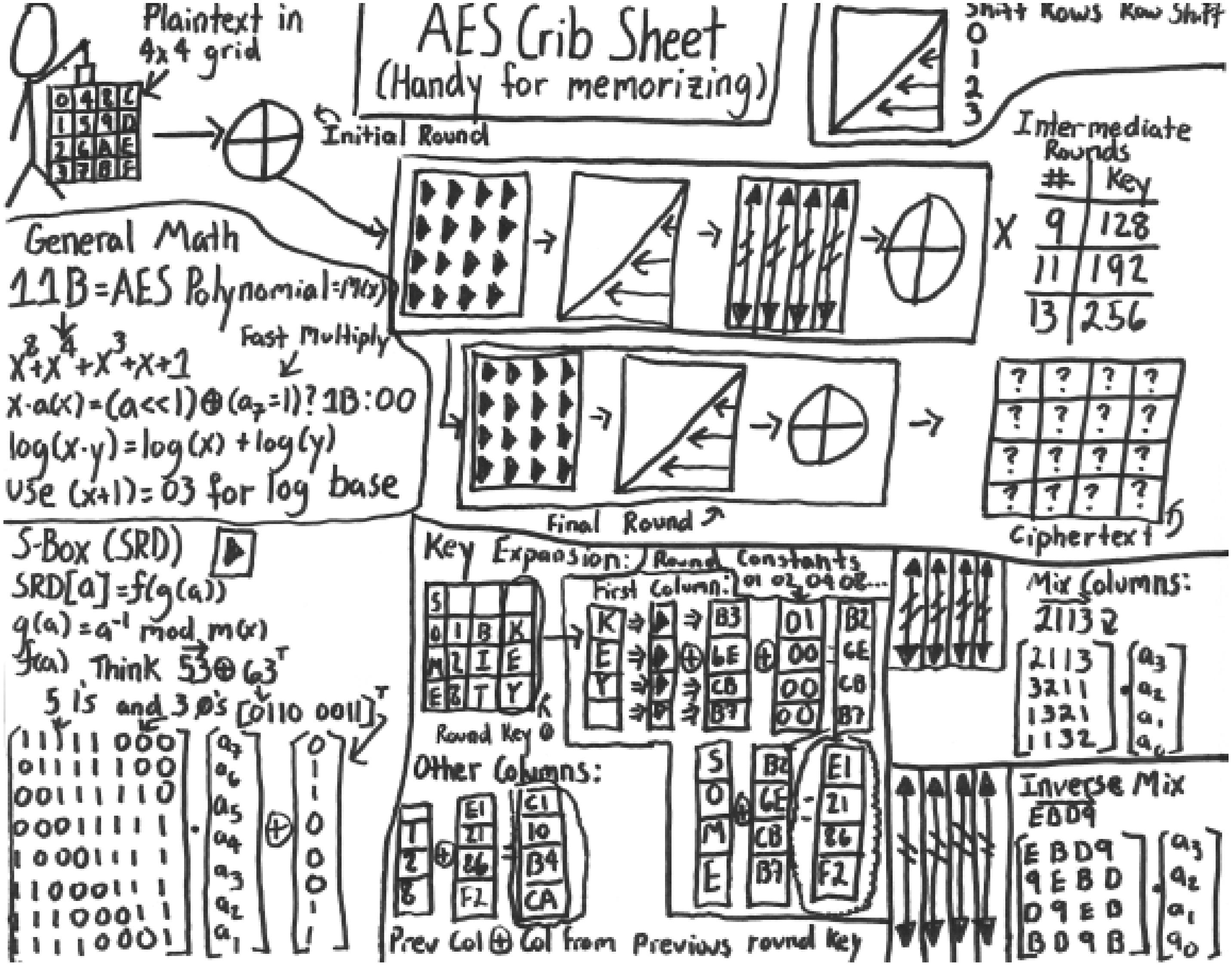


INPUT		OUTPUT
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

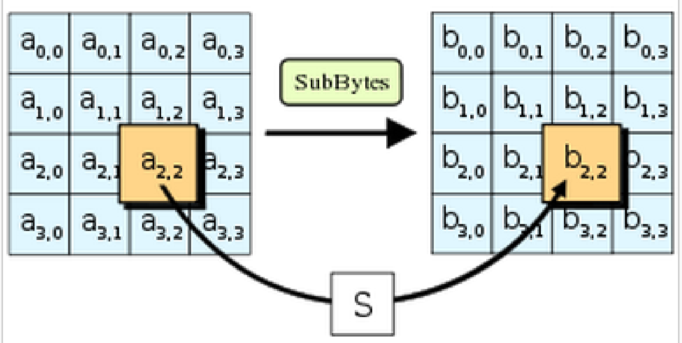
Figures from Wikipedia



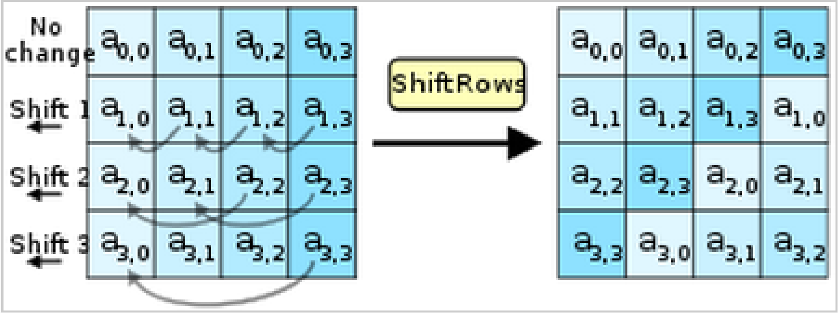
How AES Works



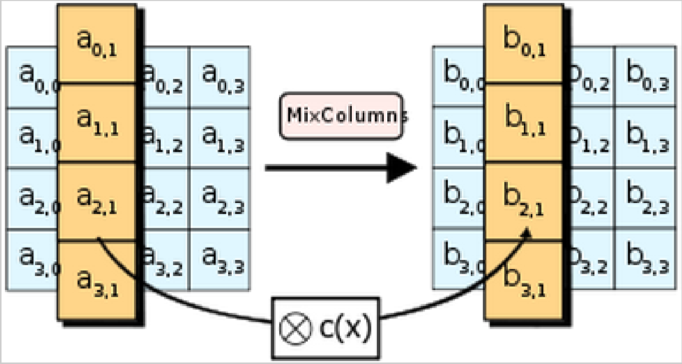
AES (from Wikipedia)



In the `SubBytes` step, each byte in the state is replaced with its entry in a fixed 8-bit lookup table, *S*; $b_{ij} = S(a_{ij})$.

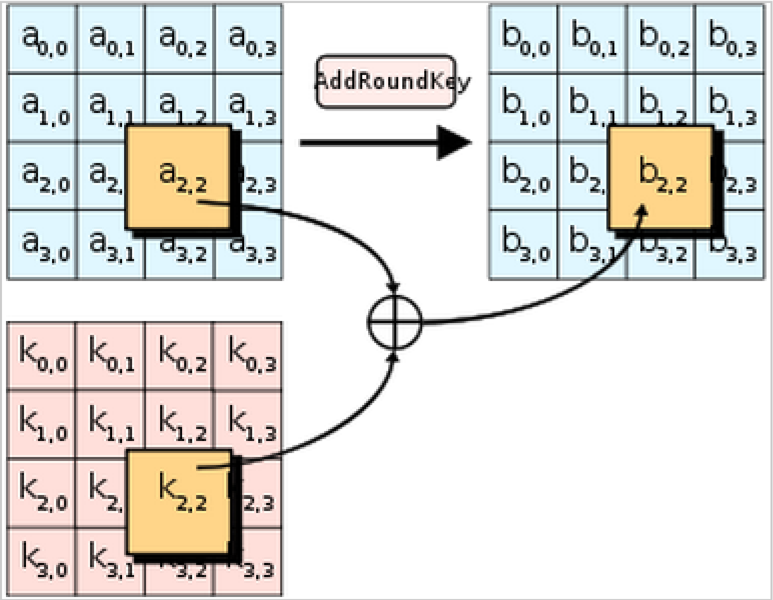


In the `ShiftRows` step, bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs for each row.



In the `MixColumns` step, each column of the state is multiplied with a fixed polynomial *c(x)*.

Originally was called Rijndael. This is just one of up to 14 rounds. The block size has a maximum of 256 bits, but the key size has no theoretical maximum.



In the `AddRoundKey` step, each byte of the state is combined with a byte of the round subkey using the XOR operation (⊕).

Windows Bit Locker

- Some versions of Vista and all versions of Windows 7 allow you to encrypt your files using a version of AES
  - Vista: http://technet.microsoft.com/en-us/library/cc766200(v=ws.10).aspx
  - W7: http://technet.microsoft.com/en-us/library/dd835565(v=ws.10).aspx
- Requires a one time partitioning of your HHD
  - I'm not sure its for the faint of heart – when I was looking into this the article said that this may stop your computer from booting
- People who travel with laptops that store sensitive information should encrypt it
- Don't lose the key

Diffie-Hellman Key Exchange

- Whitfield Diffie (MIT ’65 and MITRE) became interested in security and foresaw the need for it on the emerging ARPANet and what was to come
- He moved to Sanford to work on encryption with Martin Hellman and subsequently Ralph Merkle.
- Up to this time encryption was mostly with two-way **(symmetric*)** algorithms and D-H-M focused on one-way **(asymmetric^)** algorithms.
- To do this they used **modular arithmetic**
- This procedure allows Alice and Bob the establish a common secret key even when Eve sees the messages they exchange – DHE is susceptible to “man-in-middle attacks”.

* A symmetric encryption can be decrypted by reversing the steps

^ An asymmetric encryption is considerably harder to decrypt (“computationally infeasible”) and can not be decrypted by reversing the encryption steps

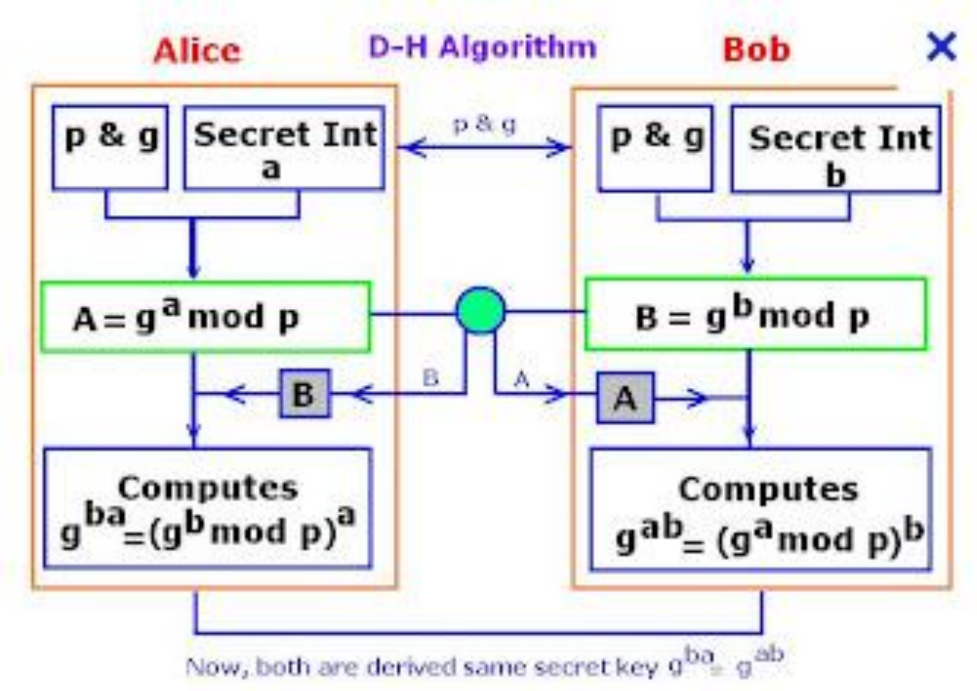
Modular Arithmetic

- Asymmetric functions are ones where it is very easy to go one way, but vey hard to reverse (not impossible but not in say the life of the universe)
- **a(mod b)** means you divide **a** by **b** and the answer is the remainder
- For example 2130(mod 1200) is 930 as for a 24 hr. clock, or 7(mod5)=2
- Modular arithmetic was studied by many great mathematicians including Euler and Fermat who developed theorems that help make computations easier.
- Modular arithmetic :
  - Gives a very erratic answers for small changes in the input – this makes it extremely hard to go reverse the process
  - Makes it possible to handle very large numbers – w/o modular math the numbers used in present day encryptions would be too big to even be stored by any computer

x	1	2	3	4	5	6	7	8	9	10	11	12
3^X	3	9	27	81	243	729	2187	6561	19683	59049	177147	531441
3^X (mod 17)	3	9	10	13	5	15	11	16	14	8	7	4

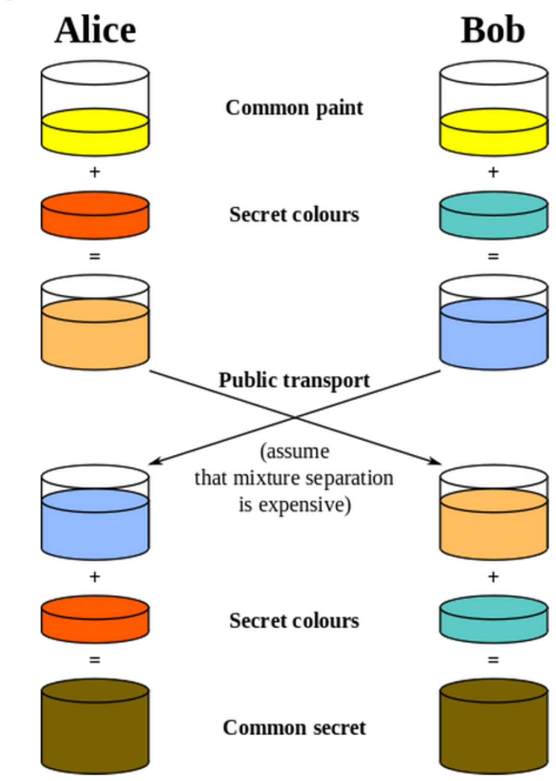
Diffie-Hellman-Merkle key agreement protocol

Modular Math Exchange


Alice
D-H Algorithm
Bob
×
p & g
Secret Int a
p & g
p & g
Secret Int b
$A = g^a \bmod p$
$B = g^b \bmod p$
B
B
A
A
Computes $g^{ba} = (g^b \bmod p)^a$
Computes $g^{ab} = (g^a \bmod p)^b$
Now, both are derived same secret key $g^{ba} = g^{ab}$

Asymmetric math is what allows Alice & Bob to freely exchange values even if Eve is listening.

Paint analogy



See Diffie Hellman computation

Private Key Encryption (PKE)

The Diffie, Hellman and Merkle key agreement protocol had/has some drawbacks:

- Its open to man-in-the-middle attacks – Eve could pretend to be both Alice and Bob during the initial contact
- Inefficient – have to go back & forth – its not as if Alice could simply deciding to send an encrypted message to Bob on the spur of the minute

Therefore, in addition to developing D-E key exchange, Diffie and Hellman developed the framework for an asymmetric cypher in 1975 (but not the final algorithm). They presented the concept in a paper, but it was **RS&A** who devised with the actual implementation.

DHE is still used extensively, but in many cases RSA is better.

RSA Public Key – Private Key Encryption

- Meanwhile back at MIT three young number theorists decide to take a crack at D-H suggestion.
- **RSA** – **R**ivest, **S**hamir and **A**dleman – RSA could refer to the people who developed the algorithm, the algorithm itself, or the company they founded (purchased by EMC for $2.1 B in 2006)
- The concept is shown on the next slide per a paper by the **PGP** folks (ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf).

First conventional symmetric encryption:

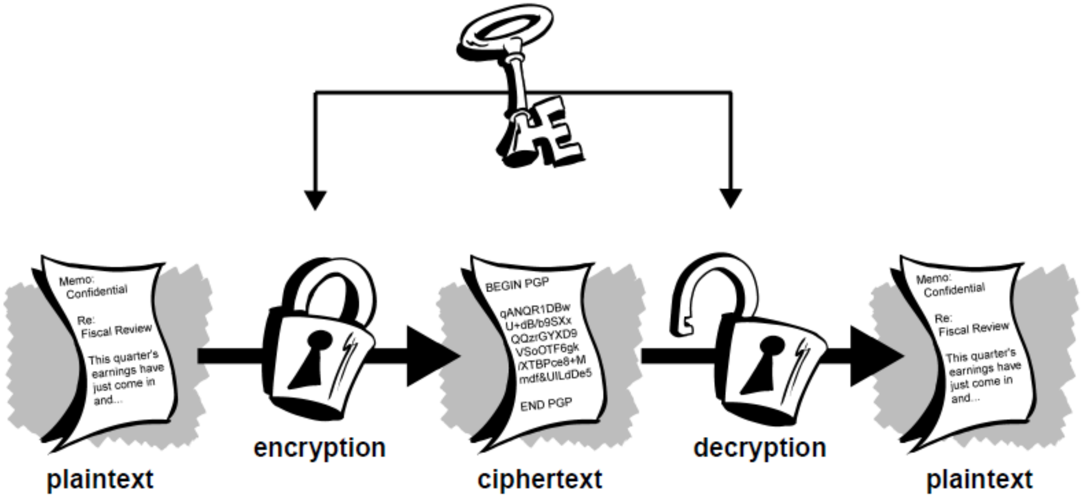


Figure 1-2. Conventional encryption

Public Key Encryption Concept

Bob makes his public key(s) available, generally through a trusted third party, Alice would get one and use it to encrypt a message and send it to Bob. Only Bob could open it using his private key.

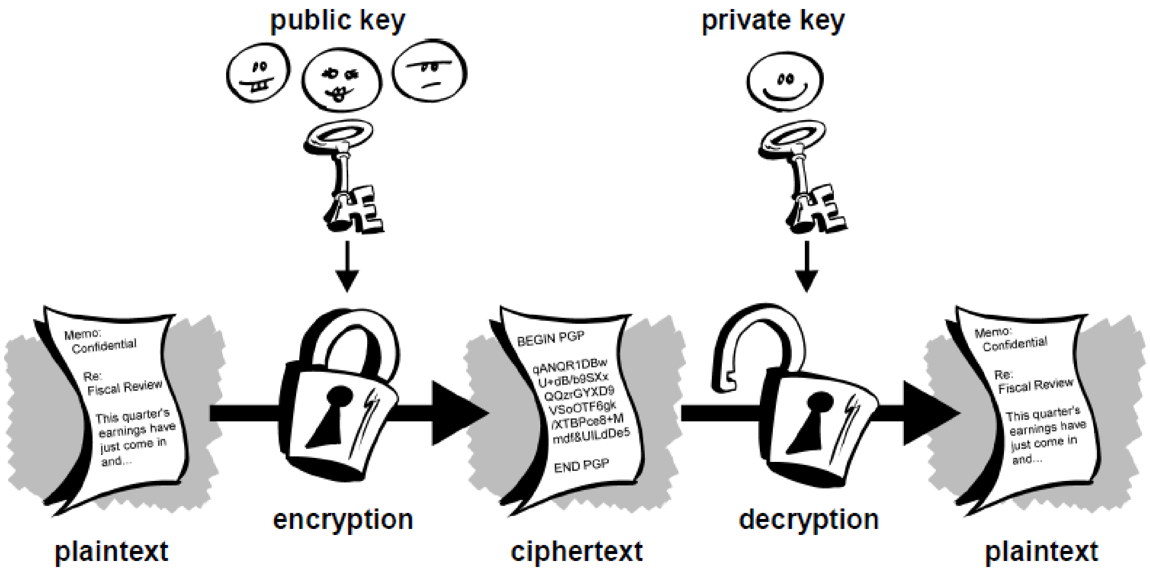


Figure 1-3. Public key encryption

It's a bit like Bob distributing a bunch of open padlocks for others to use to lock up information but that only he could open.

Digital Signature

- Alice send something out encrypted with her private key (something only she can do).
- Anyone with a authentic Alice public key can open it – so if it opens they know without a doubt that its from her.

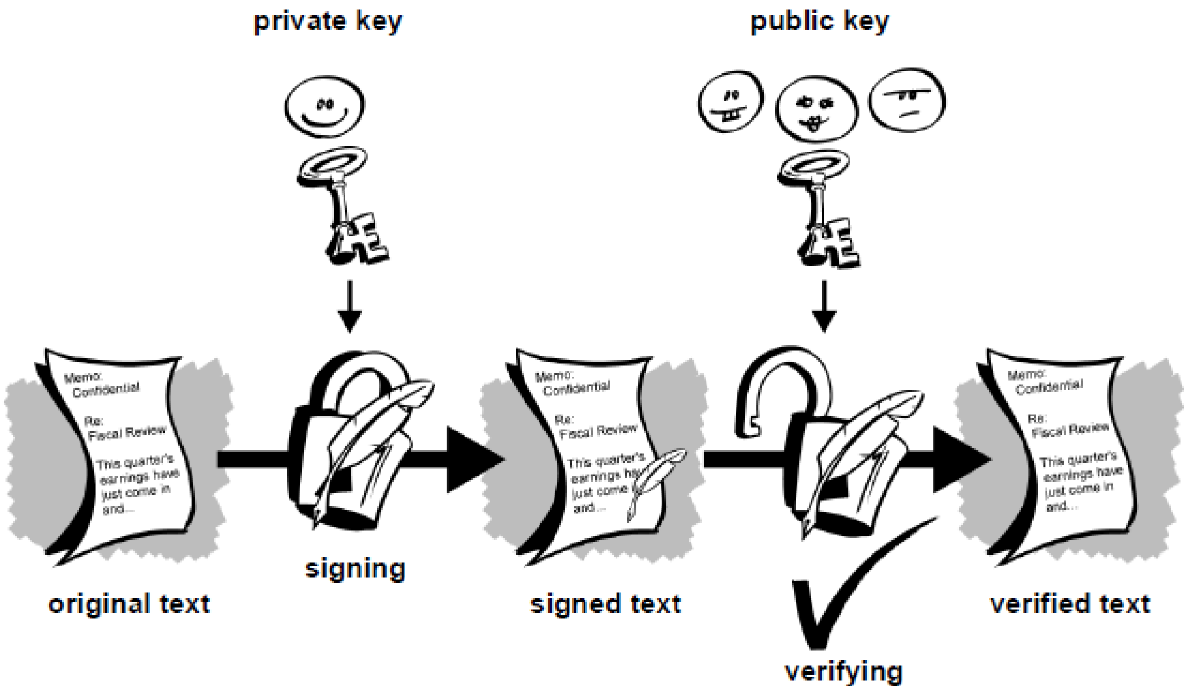


Figure 1-6. Simple digital signatures

Public & Private Keys based on Prime Numbers

- A prime number (a.k.a. a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself.
- Two is the only even prime.
- Numbers that are not primes (composite numbers) can be factored into primes starting with small primes and working your way up.

  For example: 315 = 3*3*5*7 = 3^2*5*7
- Primes: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, …
- The are theorems that make it relative easy to determine if a number is prime. To generate a large prime the approach is to generate a large random number and then to check to see if it is prime.

Public & Private Keys using Prime Numbers

- **If two large primes are multiplied together to form a number N, it can be very difficult to determine the two primes.**
- The other things that PKE use are:
  - Raising the message, expressed as a number, to a prime number power, e.g., $398,056,519,855,850,535,137^{65537}$ although there are shortcut ways to do this with modular math
  - Modular math – this is almost a requirement – without MM the encrypted messages (expressed as a number) gets so big they would well exceed the number of atoms in the universe (less than a google)

Factoring Products of Large Prime Numbers

- Martin Gardner (Scientific American) published a co-prime value N with 129 digits (~430 bits) in 1977 and it took 17 years before the answer was found using a network of 1600 workstations.
- The most difficult integers to factor in practice(using existing algorithms) are those that are products of two large primes of similar size, and for this reason these are the integers used in cryptographic applications. The largest such semi prime yet factored was RSA-768**,** a 768-bit number with 232 decimal digits, on December 12, 2009. This factorization was a collaboration of several research institutions, spanning two years and taking the equivalent of almost **2000 years of computing** on a single core 2.2 GHz AMD Opteron.
- Now days RSA recommends a value of N with approximately 1000 bits. Larger RSA values require more computation time.

Source: http://www.rsa.com/rsalabs/node.asp?id=3723

Public Key Encryption -- the Math

1. Randomly choose two prime numbers: p and q.
   - $p = 127$, $q = 211$
2. Compute $N = pq$
   - $N = 127 * 211 = 26{,}797$
3. Compute $N' = (p – 1)(q – 1)$
   - $N' = (127 – 1) * (211 – 1) = 26{,}460$
4. Choose e another prime < N' (or some other less stringent restrictions)
   - $e = 13{,}379$ [Recommended values are generally are not too large.]
5. Compute d as the multiplicative inverse of e, mod N' *[ed(modN') = 1]*
   - $d = 11{,}099$ since 13,379*11,099(mod 26,460) = 1
   - This may be the most difficult step, but there's a procedure for doing it known as the *extended Euclidean algorithm* that dates back to c.300 BCE.

Public Key Encryption -- Example

6. Destroy **p, q** & **N'** , keep **e, N** & **d**

7. Use **e** and **N** to encrypt a message ⇐ Public key information

8. Use **d** and **N** to decrypt.

Encryption Example – Encrypt a message M which is expressed as a number

- encrypt (M) = M^e (mod N), where in this example M is 10,237
- encrypt(10,237) = $10{,}237^{13{,}379}$(mod 26,797) = 8422
- 8422 is the encrypted message (R)

Decryption Example

- decrypt(R) = R^d(mod N) R is the received encrypted message
- decrypt(8422) = $8422^{11{,}099}$(mod 26797) = 10,237

Public Key Encryption -- Example

6. Destroy **p, q, N'**

7. Use **e** and **N** to encrypt a message ⇐ Public key information

8. Use **d** to decrypt.

This is the product of two large primes which is difficult to factor.

Encryption Example

- encrypt (M) = M^e (mod N), where M is the message to be sent
- encrypt(10,237) = $\boxed{10{,}237^{13{,}379}}$ (mod 26797) = 8422*
- 8422 is the encrypted M

Note that this number would have ~54,000 digits, and this example uses small numbers.

Decryption Example

- decrypt(R) = R^d(mod N), where R is the received encrypted message
- decrypt(8422) = 8422^{11099}(mod 26797) = 10,237*

* In modular math there are shortcut ways to handle large powers – you can not do this on your pocket calculator

Alternative Histories of Public Key Encryption

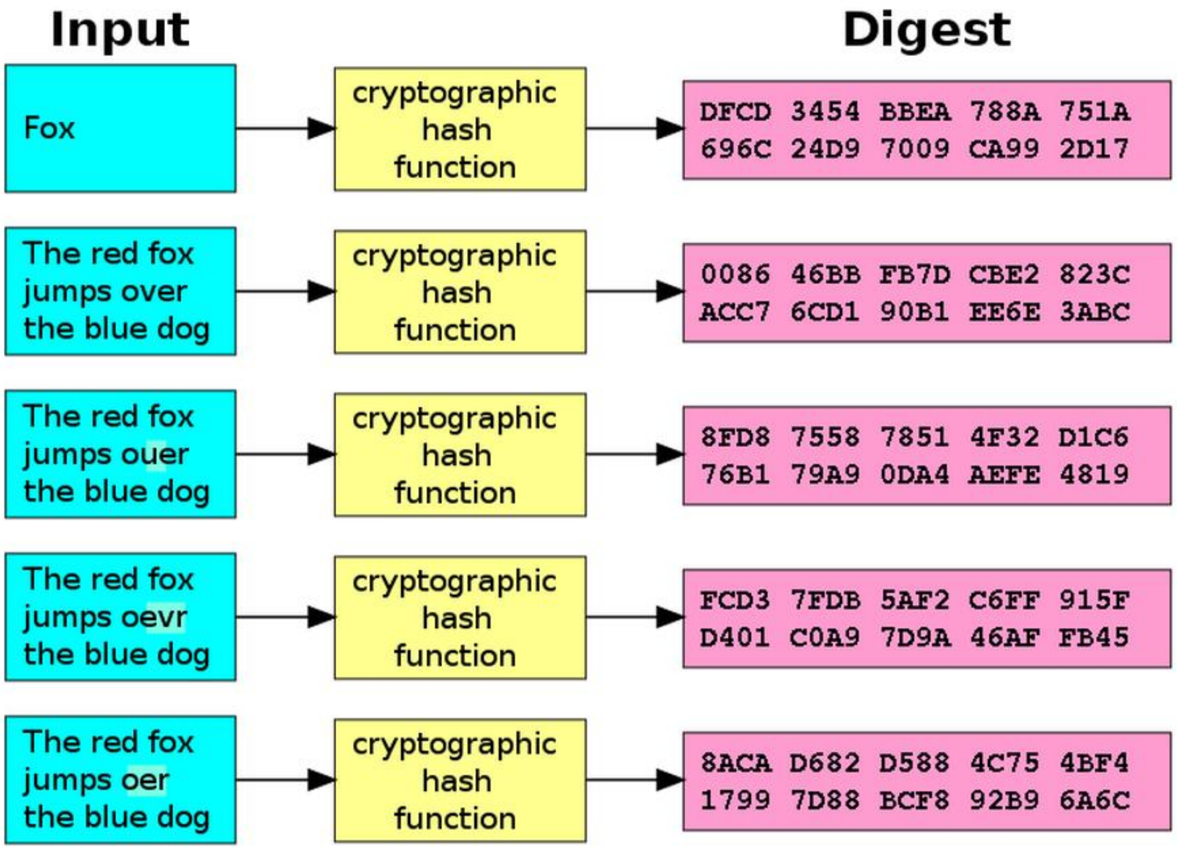
- **The Brits did it first** – about three years before RSA, but kept it secret until 1997
- The CESG (Communications Electronic Security Group), that is the Bletchley Park's wartime operation, by kept working after the war.
- Ellis, Cocks and Williamson discovered the fundamentals of RSA before Diffie, Hellman and Merkle discovered DHE.
- Only several years after RSA was well known was the CESG work acknowledged.
- What seems to be amazing is that not only did CESG come up with the asymmetric public key / private key concept, but from what is known, it was almost identical to RSA
- There is also some hint that **NSA did work on PKE even earlier** in the 60's. But what they did is still classified.

PGP (Pretty Good Privacy)

- RSA is somewhat expensive (proprietary and computationally intensive) to implement for everyday transactions.
- Phil Zimmerman in the mid-80’s came up with the concept of PGP – it sort of like good enough encryption for the masses.
- PGP is not as secure as RSA but its pretty good and runs a 1000 times faster.
- The basic concept is that DES is good but the key system is weak, so use something like DES to encrypt the plaintext but generate a strong keyword using RSA. Prime numbers were generated based on mouse movements.
- Zimmerman’s software package ran into a lot of opposition from US Customs officials and RSA (patent infringement). He was accused of being an arms dealer for selling PPG outside the country over the Internet. While Zimmerman was fighting the courts, PGP and RSA were being developed in Europe so the export question became mute. MIT published PPG (in the form of an OCR book) and PGP settled with RSA by getting a license.
- After changing hands a few times, PGP is now owned by Symantec.

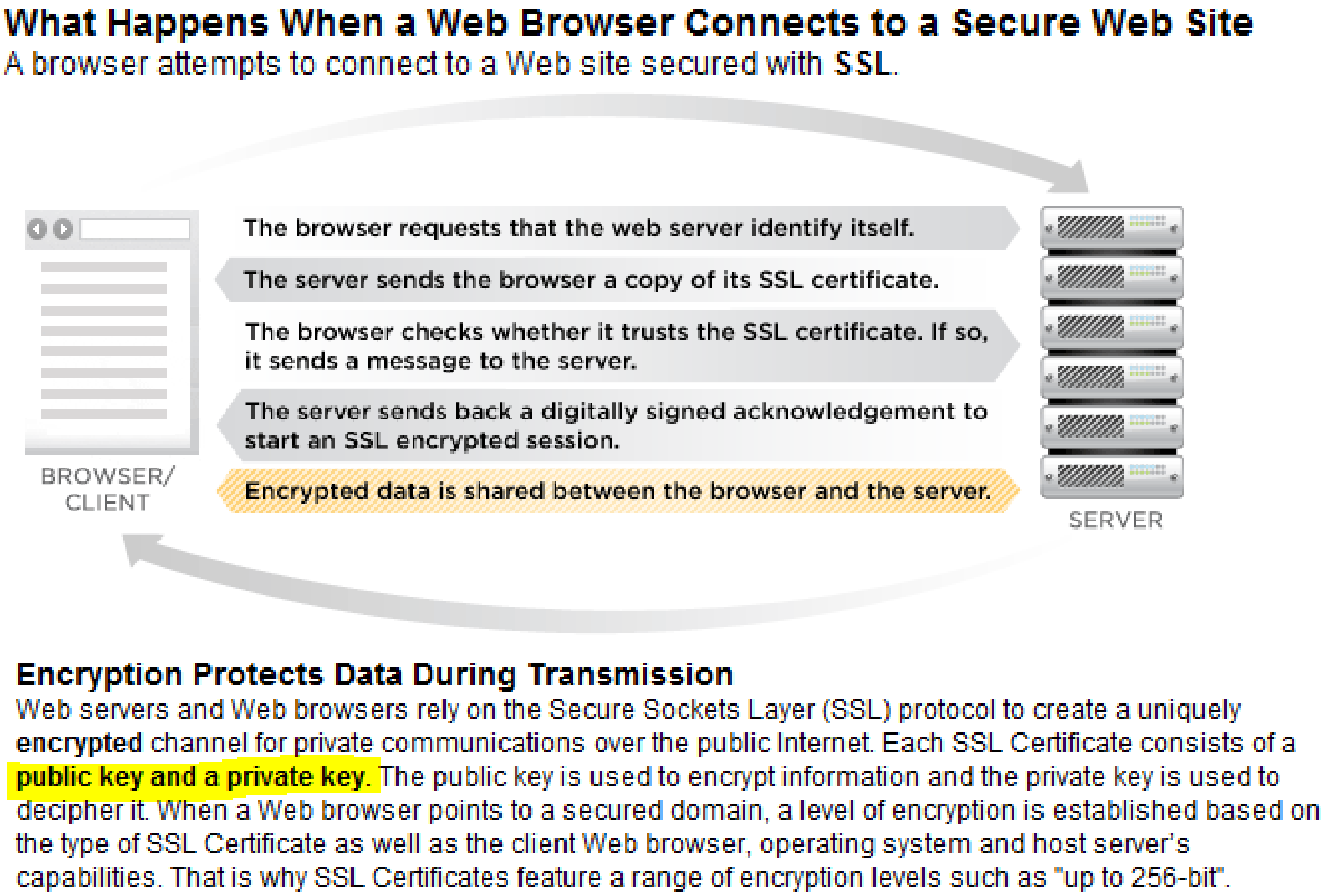
HASH Functions

From Wikipedia: A **cryptographic hash function** is a hash function that can be defined as a deterministic procedure that takes an arbitrary block of data and returns a fixed-size bit string, the (**cryptographic**) **hash value**, such that an accidental or intentional change to the data will change the hash value.


Input
Digest
Fox
cryptographic hash function
DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
The red fox jumps over the blue dog
cryptographic hash function
0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC
The red fox jumps ouer the blue dog
cryptographic hash function
8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819
The red fox jumps oevr the blue dog
cryptographic hash function
FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45
The red fox jumps oer the blue dog
cryptographic hash function
8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

Hash digests can not be decrypted. However, as one example, it could be a way to handle passwords. A site would have on record the digests of passwords, but not the actual password. When you connect it would hash your password and compare that to the digest. After hashing it would destroy the provided password.

SSL (secure socket layer) data transmission



What Happens When a Web Browser Connects to a Secure Web Site

A browser attempts to connect to a Web site secured with SSL.

The browser requests that the web server identify itself.

The server sends the browser a copy of its SSL certificate.

The browser checks whether it trusts the SSL certificate. If so, it sends a message to the server.

The server sends back a digitally signed acknowledgement to start an SSL encrypted session.

Encrypted data is shared between the browser and the server.

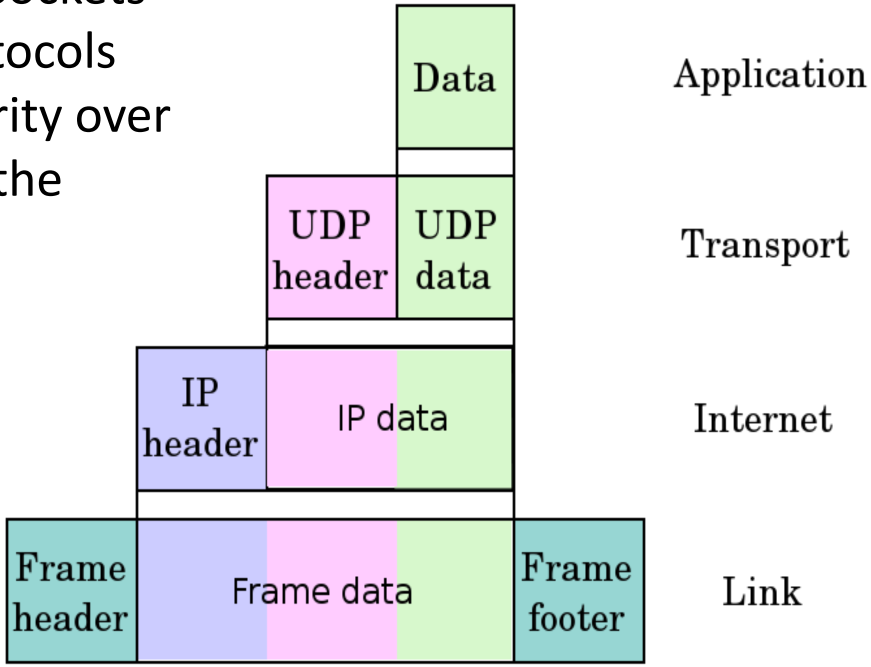
Encryption Protects Data During Transmission

Web servers and Web browsers rely on the Secure Sockets Layer (SSL) protocol to create a uniquely **encrypted** channel for private communications over the public Internet. Each SSL Certificate consists of a **public key and a private key**. The public key is used to encrypt information and the private key is used to decipher it. When a Web browser points to a secured domain, a level of encryption is established based on the type of SSL Certificate as well as the client Web browser, operating system and host server's capabilities. That is why SSL Certificates feature a range of encryption levels such as "up to 256-bit".

From: http://www.verisign.com/ssl/ssl-information-center/how-ssl-security-works/index.html

TLS (transport layer security)

From Wikipedia: Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols that provide communication security over the Internet. TLS and SSL encrypt the segments of network connections above the Transport Layer, **using asymmetric cryptography for key exchange, symmetric encryption for privacy, and message authentication codes for message integrity.** (Sounds a lot like PGP.)



Hypertext Transfer Protocol Secure (HTTPS) is a combination of Hypertext Transfer Protocol (HTTP) with SSL/TLS protocol.

SSL/TLS History

- **SSL** was developed by **Netscape**. The 1996 draft of SSL 3.0 was published by **IETF** (Internet Engineering Task Force). IETF released TLS 1.0 in January 1999. TLS requires a **certification authority** (CA) who holds public keys and issues temporary private keys.
- A good hunk of that business went to **VeriSign** (a **RSA** Security spin-off) who owns a lot of internet infrastructure and intellectual property. In 2010 Verisign sold its authentication business unit to **Symantec** (the Norton folks) for $1.28 billion. **GoDaddy** and **Comodo** are other key players.

Extra slides

Some common notation and terms

Alice, Bob and Eve – Common names used in encryption papers/articles: Alice (A) and Bob (B) want to exchange secret information and Eve wants to eavesdrop

Code, Cipher – A general method for hiding a message in text-like message (e.g., a Vigenerre square method)

Encrypt, Encode – to change from plain text to cipher text

Encryption – the process of disguising a message in such a way as to hide its substance.

DES/AES – Data/Advanced encryption standards – U.S. NBS/NIST standards

SSL – secure socket layer

TLS -- Transport Layer Security

RSA – **R**ivest, **S**hamir and **A**dleman – RSA could refer to the people, the cipher or the company (purchased by EMC for $2.1 B$ in 2006)

One-way hash -- a function of a variable string to create a fixed length value representing the original pre-image, also called a message digest, fingerprint, or message integrity check.

Some common notation and terms

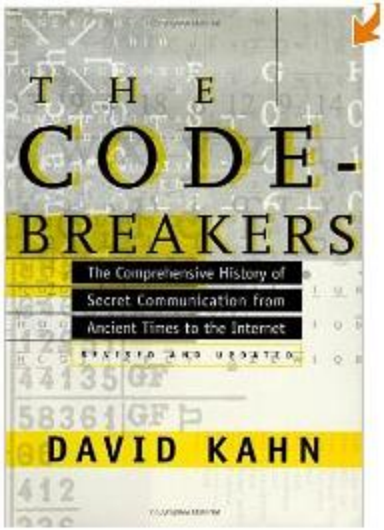
For more definitions see This document written by the PGP folks.

ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf

- Cryptography is the practice and study of techniques for secure communication in the presence of third parties (adversaries). In cryptography, encryption is the process of transforming information (referred to as plaintext) using an algorithm (called a cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key. The result of the process is encrypted information (in cryptography, referred to as ciphertext).
  - Cryptology: science of secret communication.
  - Cryptography: science of creating secret codes.
  - Cryptanalysis: science of code breaking.
- In cryptography, rubber-hose cryptanalysis is the extraction of cryptographic secrets (e.g. the password to an encrypted file) from a person by coercion or torture, in contrast to a mathematical or technical cryptanalytic attack.

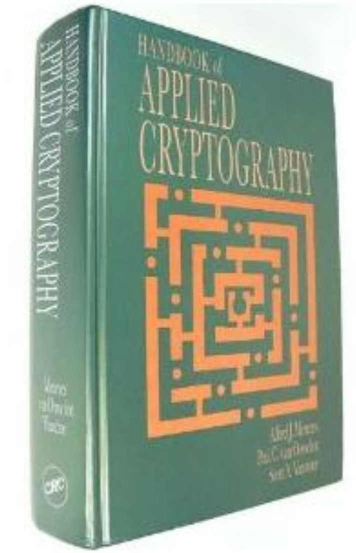
Some References

- *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet* by David Kahn is recognized the most through encryption book from ancient times to the 1960's. A shortened PDF version is available here: http://www.iraqiforum.org/books/2/Cryptography/4.pdf


THE
CODE-
BREAKERS
The Comprehensive History of
Secret Communication from
Ancient Times to the Internet
REVISED AND UPDATED
DAVID KAHN

- A full copy (816 pp.) of the *Handbook of Cryptography* is available for free on the internet as PDF files:

  http://cacr.uwaterloo.ca/hac/



- Other extended internet (HTML) write-up on cryptography is provided at:

  An Overview of Cryptography
  http://www.garykessler.net/library/crypto.html

  and

  U. of Wash. course on elementary no. theory incl. encryption

Even more misc. info.

Hedy Lamarr realized that by transmitting radio signals along rapidly changing, or hopping frequencies, American radio-guided weapons would be far more resilient to detection and jamming. The sequence of frequencies would be known by both the transmitter and receiver ahead of time, but to the German detectors their message would seem like gibberish. "No jammer could detect it, no German code-breaker could decipher a completely random code.“ (see US Pat.# 2292387)

God may not play dice with the universe, but something strange is going on with prime numbers -- Paul Erdos

The US had a machine similar to Enigma called SIGABA.

Free AES S/W: http://www.aescrypt.com/