Secret Writing 1

Steganography

Cryptography

Codes

Private Keys

Overview

Secret Writing (and especially Cryptography) is an important part of security but is by no means the only part

Necessary but not sufficient

Other parts of security include

Firewalls

Passwords

Scanning

Constraint of mobile code

And on and on…

Proxy servers

File access

Attack mechanisms

Virus, worms & trojans

Overview

This set of two lectures consists of the following

Terminology and Taxonomy

Some History, Examples, and Concepts

Private Key Encryption

Public Key Encryption

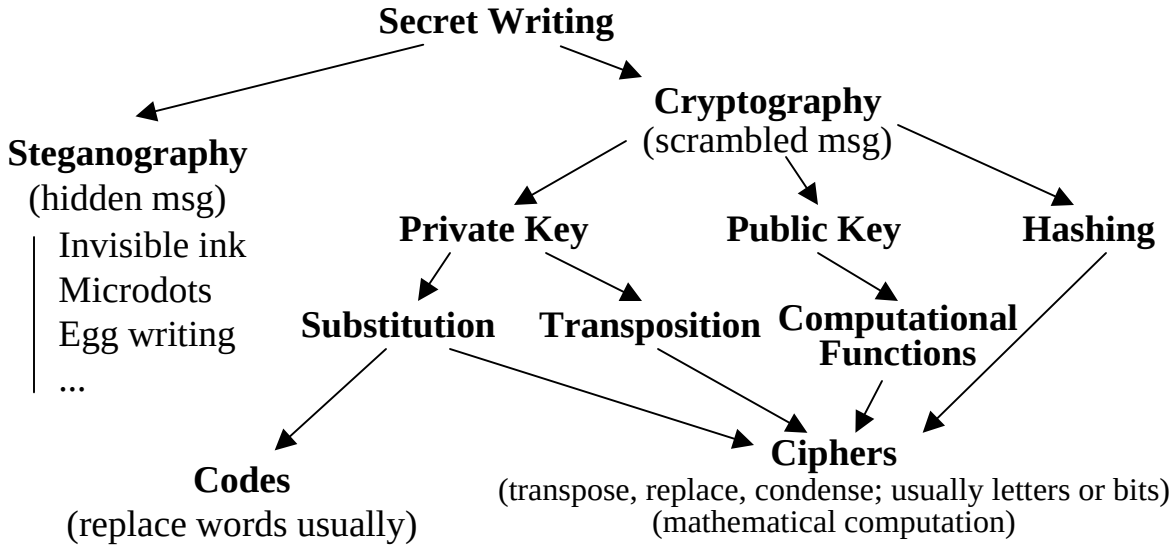
Hashing

Authentication

Non-repudiation

Putting it all together

A Taxonomy



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.** Private key ciphers usually use S and T. Public key systems use CF.

A Notable Code:
Navajo natural language

S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

Some Terminology

Secret Writing: The science of hiding ***messages*** or their ***meaning*** from outsiders.

Steganography: Science of hiding messages so that the ***existence*** of the message is ***unknown*** to outsiders.

Cryptography: Science of hiding the ***meaning*** of messages that are ***known*** to exist.

Writing and reading scrambled (often incorrectly referred to as "coded") messages.

Today ***cryptography*** subsumes ***coding***, ***transposing***, ***hashing***, and some forms of authentication.

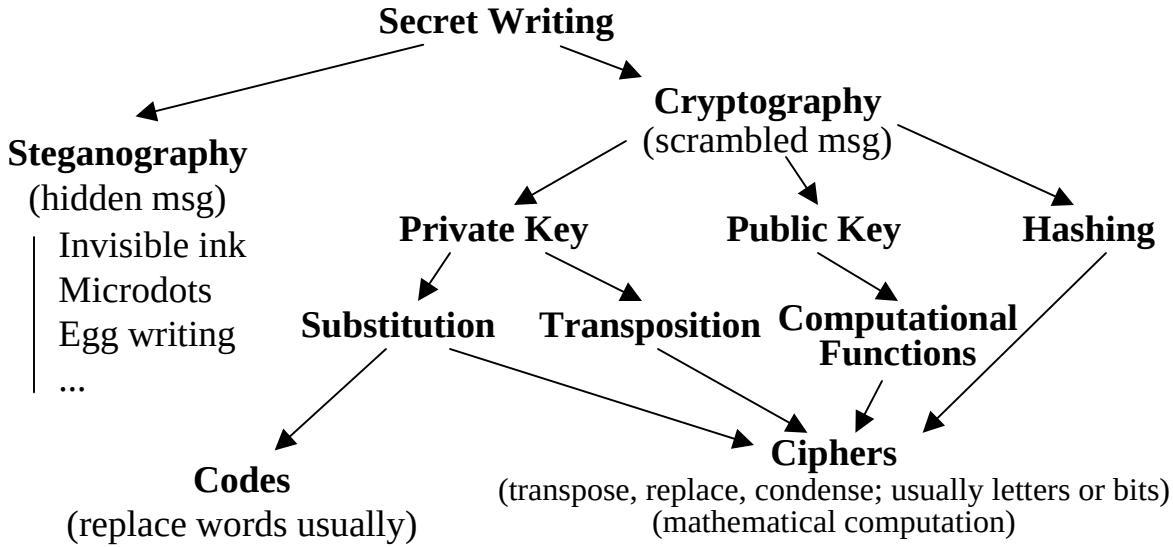
Some Terminology

Coding (Substituting): The process of substituting one part of a message with something that is external to the message so as to hide its meaning from outsiders. Parts can be letters, words, phrases, bits...

Transposing: The process of moving parts of a message from one place to another within the message so as to hide its meaning from outsiders. Parts can be letters, words, phrases, bits...

Hashing: Creating a fixed-size value called a “hash” using a message as the source to create the hash, and doing it in a way that distributes the possible messages evenly among the possible hash values.

A Taxonomy



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.** Private key ciphers usually use S and T. Public key systems use CF.

A Notable Code:
Navajo natural language

S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

Some More Terminology

Cryptanalyst: A person or persons that try to decipher encrypted messages. Often synonymous with "attacker" or "threat".

Encryption: The science of scrambling the contents of the message in such a way that hides its meaning from outsiders.

Plaintext or Cleartext: Unencrypted message (not necessarily text).

Ciphertext: Encrypted plaintext.

Some More Terminology

Confidentiality: Ensures that meaning of message is hidden for outsiders.

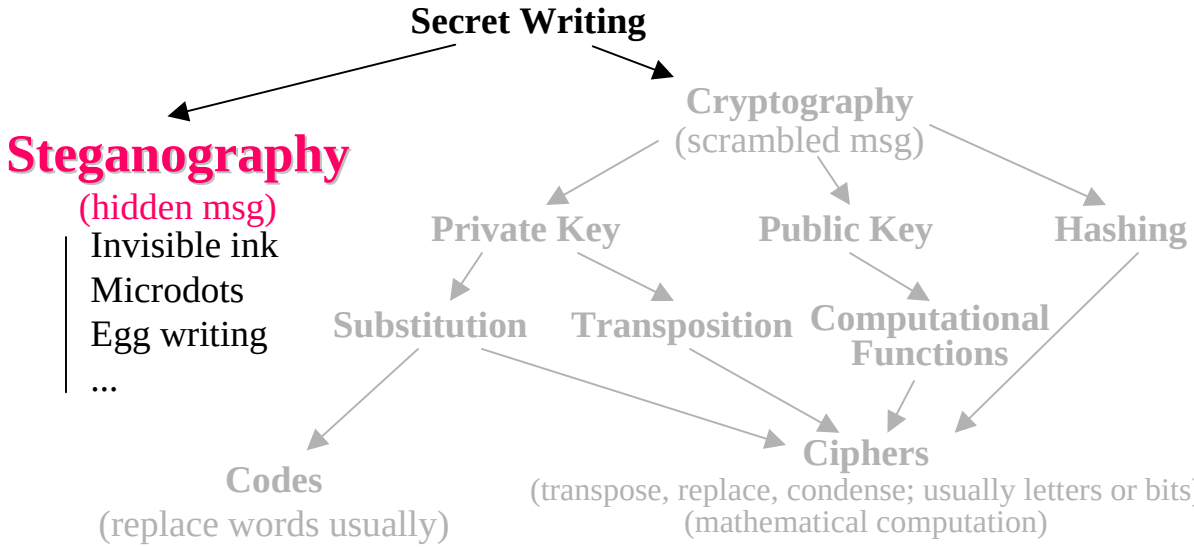
Authentication: Establishes the identity of the sender of the message.

Integrity: Ensures that the message and its authentication have not been changed.

Non repudiation: The sender cannot deny that she or he sent the message. Closely related to Authentication.

Steganography

A Taxonomy



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.** Private key ciphers usually use S and T. Public key systems use CF.

A Notable Code:
Navajo natural language

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

S = substitution
T = transposition
H = hashing
CF = computational functions

Steganography

Hides the fact that there is a message

Traced back to ancient Greece and China

Herodotus and Cicero chronicled its use

Some ingenious schemes

Invisible ink

Tattoos (shave head, tattoo, then let hair grow back -- Herodotus)

Inconspicuous mark on a tree or rock

Egg writing (ink made of alum and vinegar that penetrates the shell and leaves message on the hardened albumen after it is boiled)

Microdots (Germans used it in WW1 -- microfiche the size of a printed period put over the period in some innocent text)

Sequence of font usage in a newspaper, magazine…

Steganography

All sorts of schemes can be thought of

Used by allies during WW2 before Normandy invasion to alert the underground in France of the invasion & what to do

*The French resistance was alerted to carry out tasks by means of the **Messages Personnels**, transmitted by the BBC in French*

Hundred of mesages were regularly transmitted, masking the few of them that were really significant.

Messages transmitted at the end of May 1944

Quote from poem by Verlaine: "Les sanglots longs des violons de l'automne" (Long sobs of autumn violins)

Instructed the resistance to start disrupting rail traffic in the next few days

Transmitted on 5 June 1944

Les carottes sont cuites *("The carrots are cooked")*

Les dés sont jetés *("The dice have been thrown")*

Get ready!

Line from Verlaine poem: "Bercent mon coeur d'une langueur monotone" ("soothe my heart with a monotonous languor")

Attack now!

Steganography

Steganography bit rates and content are usually limited

But modern cyber stego allows large covert messages

The important point here is:

Enemy doesn't know that there's a message.

Steganography Example

Seascape without hidden message

Seascape with hidden message

The hidden message is:

"This is a test to see if I can hide and then detect this message in this image of a seascape."

Another Steganography Example That is Noticible

Steganography often limits the amount of information that can be effectively hidden

Steganography

I will not discuss steganography in this course

Steganography and steganalysis will be covered in

IT-S539 and ITMS539

Scheduled for spring 2015 (next semester)

Students doing stego projects will take ITMS539

Not ITMS549

Cryptography

What If Message is Known to Exist

What if

The contents is more extensive than steganography can handle? OR

The existence of a message is expected? OR

Stego is not useful or cumbersome

What's an example?

Then Cryptography is looked to for a solution

Cryptography

Cryptography consists of

Codes

Ciphers

Codes and Ciphers employ four broad technologies

Substitution

Transposition

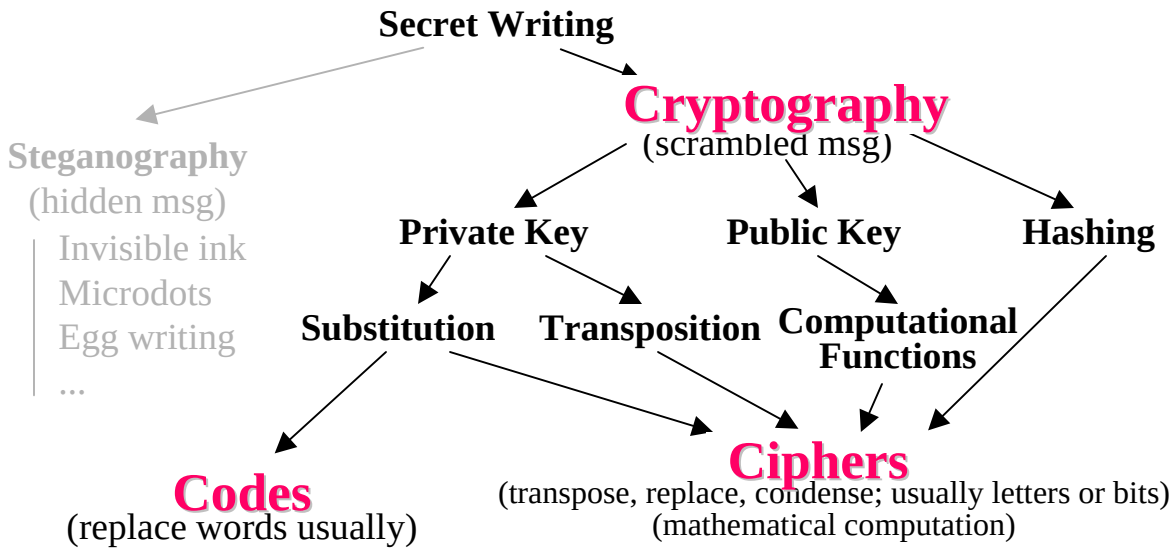
Computational functions

Hashing

Codes employ coarse grained *substitution*

Ciphers employ all four of the above

Taxonomy Again



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.**

Private key ciphers usually use S and T.

Public key systems use CF.

A Notable Code:
Navajo natural language

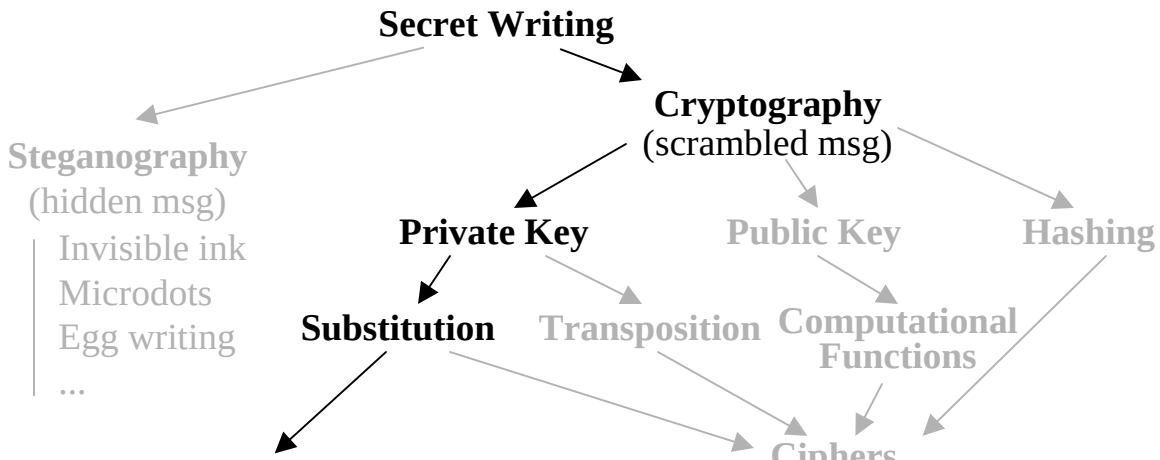
S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

Codes

Taxonomy Again



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.**

Private key ciphers usually use S and T.

Public key systems use CF.

A Notable Code:
Navajo natural language

S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

Codes

A code substitutes a part of a message with *something else* so as to hide its meaning from outsiders.

The "something else" is not another part of the message

This would be *transposition*

The "something else" is exogenous to the message

One word for another

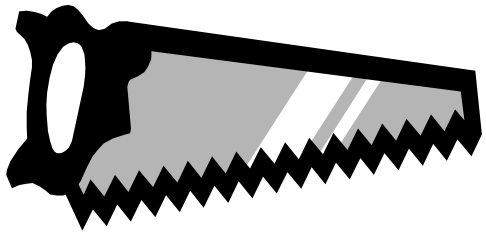
A word or phrase in one language for that of another language

A number for a letter or word

A symbol for part of a message

Also: ***Coarse grained***

Example of a Code*



A


What does this code tell you?

A WWII Code - Background

During WWII in the Pacific theater of war, the Japanese had a sophisticated cipher that the U.S. called ***Purple***

The U.S. broke the cipher allowing the U.S. Navy to

Know that there was a diversionary Japanese attack in the Aleutian Islands to mask the attack on Midway

Know when the Midway Island attack would occur

Know when and where Admiral Isoruko Yamamoto would be flying in a plane in the northern Solomon Islands

Nevertheless the U.S. became aware that the Japanese were very sophisticated regarding ciphers and presumably cipher cracking

WWII Code - Background

The U.S. had a cipher machine design called SIGABA

The U.S. used it with random keys that were changed often

SIGABA remained secure

But the SIGABA machines, although “luggable”, were heavy and encoding and decoding were tedious and time consuming

Field commanders needed something else

To carry about in the field, and

With which to communicate in the field with split second timing

WWII Code - Background

Walkie-Talkies and spoken English were used

Walkie-Talkies were portable short wave radios

Here are two examples of WWII walkie talkies

WWII Code - Background

To confuse the Japanese who were listening in

U.S. soldiers used as many profanities and obscenities as possible

This worked for awhile

Until the Japanese soldiers who had gone to college in the U.S. began to listen to the U.S. communications

They knew all the "good" words

What to do?

Navajo Code

U.S deployed Navajo American Indian soldiers who spoke the Navajo language to many regiments where fighting was going on

Technically Navajo was really a form of word substitution "coding" and even phrase substitution

This is an example of a Code -- not a Cipher

Navajo is one of the Na-Dene languages

Other Na-Dene languages: Athabaskan, Tinglit, Eyak

It has no known link to any Asian or European language

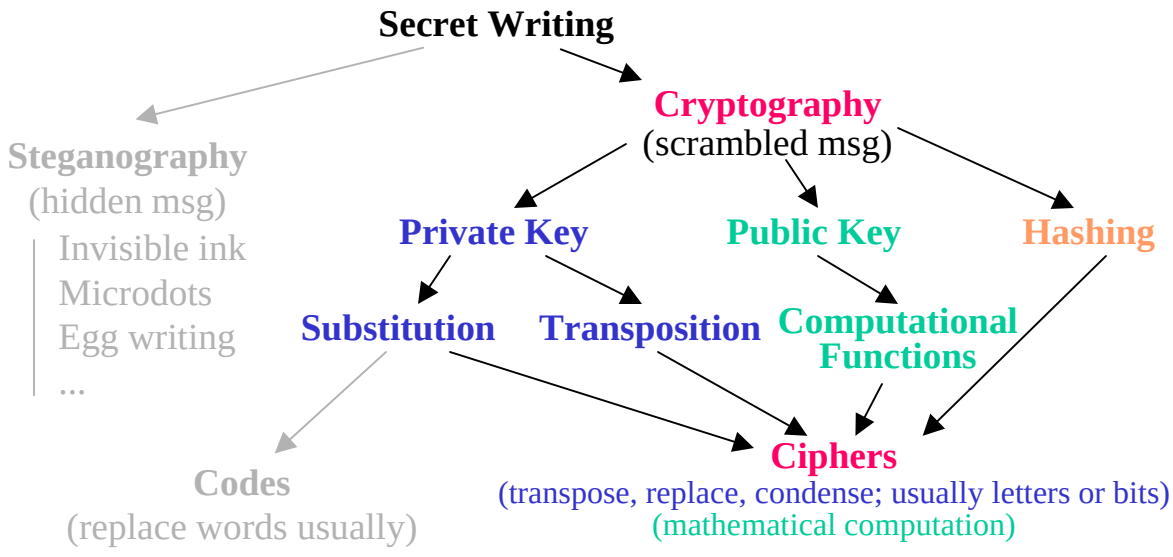
It was never broken by the Japanese

Interesting book

Doris Paul, "The Navajo Code Talkers", ISBN 0805945903, Dorrance Publishing 1998.

Cryptographic Ciphers

Taxonomy Yet Again



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.**

Private key ciphers usually use S and T.

Public key systems use CF.

A Notable Code:
Navajo natural language

S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)
Polyalphabetic (T&S)	Vigenère (S)
One-time pad (S)	Enigma (T&S)
DES (T&S)	RSA (T&S)
IDEA	MD5 (H)
RSA Pub (CF)	Diffie-Hellman (CF)
Elliptical (CF)	SHA-1 (H)
AES (T&S)	

Ciphers

Theoretically no cipher is undecipherable

Except maybe one (to be discussed later)

The idea is to make the work and time needed to break the cipher such that it is not worth it

Goals of Cryptographic Ciphers

Confidentiality

Ensures that the meaning of the message is hidden for outsiders.

Authentication

Establishes the identity of the sender of the message.

Integrity

Ensures that the message & its authentication have not been changed.

Non repudiation

The sender cannot deny that she or he sent the message.

Closely related to Authentication.

Non-repudiation is sometimes not listed as goal because it is achieved via authentication methods

Note: There is no attempt to hide the fact that a cipher exists.

How Goals Are Achieved

The three (or four) goals are achieved by using one or more of these technologies, often in combination

Private key encryption

Also called ***symmetric*** or *secret*

Public key encryption

Also called ***asymmetric***

Hashing

Also called ***message digest***

Ciphers

For all ciphers

$C = E_k[P]$

C = Ciphertext; P = Plaintext; E = Enciphering algorithm

k = symmetric secret key used with E

For all <u>really good</u> ciphers

Cryptanalyst can

Know E

Know C for chosen P

Have a much C as desired

But still cannot obtain P from C

What does the cryptanalyst not know?

Ciphers

For all ciphers

(1) $C = E_k[P]$

For all ciphers except hashing it must be such that

(2) $P = D_k,[C]$

D_k, = Deciphering algorithm used with key k'

Note: Hashing is one way

Substituting eqn (1)'s value for C into eqn (2)

(3) $P = D_k,[E_k[P]]$

Ciphers
Another Taxonomy: Stream & Block

Ciphers can also be divided into two types

Stream ciphers

Block ciphers

Stream ciphers process messages a bit or byte at a time when encrypting or decrypting

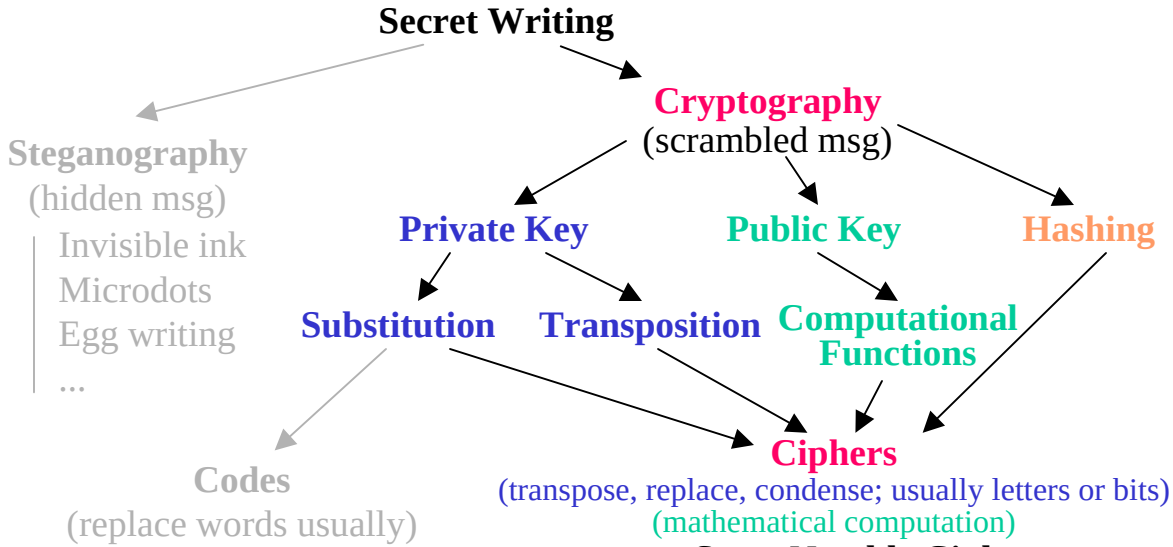
Block ciphers divide messages into blocks, each of which is then en/decrypted

Sort of like a substitution on very big characters

64-bits or more

Many current ciphers are block ciphers

Taxonomy Yet Again



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.**

Private key ciphers usually use S and T.

Public key systems use CF.

A Notable Code:
Navajo natural language

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)	**Stream**
Polyalphabetic (T&S)	One-time pad (S)	
Enigma (T&S)	Vigenère (S)	**Block**
DES (T&S)	RSA (T&S)	
IDEA	MD5 (H)	
RSA Pub (CF)	Diffie-Hellman (CF)	
Elliptical (CF)	SHA-1 (H)	
AES (T&S)		

S = substitution
T = transposition
H = hashing
CF = computational functions

Private Key Cryptography

Also called:

Symmetric Key Cryptography

Secret Key Cryptography

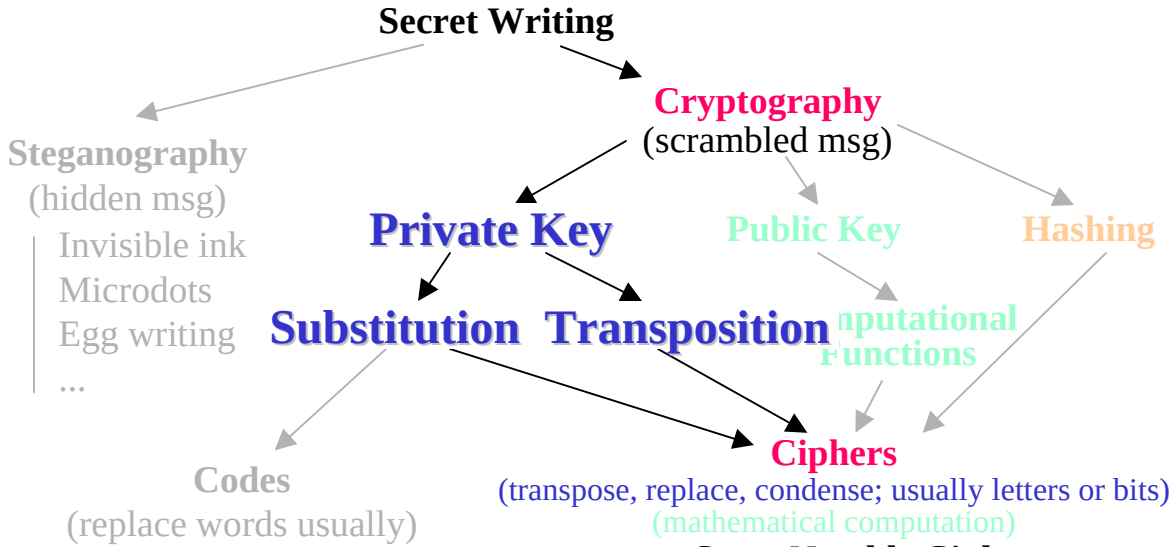
Overview
Private Key Cryptography

Private

Symmetric

Secret key

Taxonomy Yet Again



Cryptography and **Ciphers** usually refer to the same things, with Cryptography being the science and Ciphers being the encoded entities. The only exception is **Codes.**

Private key ciphers use S and T.

Public key systems use CF.

A Notable Code:
Navajo natural language

S = substitution
T = transposition
H = hashing
CF = computational functions

Some Notable Ciphers:

Caesar (T)	Monoalphabetic (S)	**Stream**
Polyalphabetic (T&S)	Vigenère (S)	
One-time pad (S)	Enigma (T&S)	
DES (T&S)	RSA (T&S)	**Block**
IDEA	MD5 (H)	
RSA Pub (CF)	Diffie-Hellman (CF)	
Elliptical (CF)	SHA-1 (H)	
AES (T&S)		

Private Ciphers

Until late 1970s all ciphers were private key

History of encryption has been a war between cryptography and cryptanalysis

First one has the upper hand; then the other!

Private Ciphers

All private (i.e., symmetric) key ciphers are combinations of *transposition* and *substitution*

Transposition means moving message parts from one place in the message to another place

Substitution means replacing message parts with other parts that are not explicitly part of the message

Message parts can be characters, bytes, bits…

Private Cipher Symmetry

Symmetry is achieved because the same or related algorithms and keys are used to encrypt and decrypt

e.g., $P = D_k[E_{k'}[P]]$

k *must be kept secret*

k' *is related to* ***k*** *or can be derived from* ***k***

In some notable encryption schemes $\boldsymbol{k' = k}$

Stream Ciphers

Stream ciphers process messages a bit or byte at a time when encrypting or decrypting

Example
Caesar Cipher (T)

Called the Caesar Cipher because Julius Caesar first document its use

Caesar Cipher is a stream cipher

Processes messages a character at a time

Example:

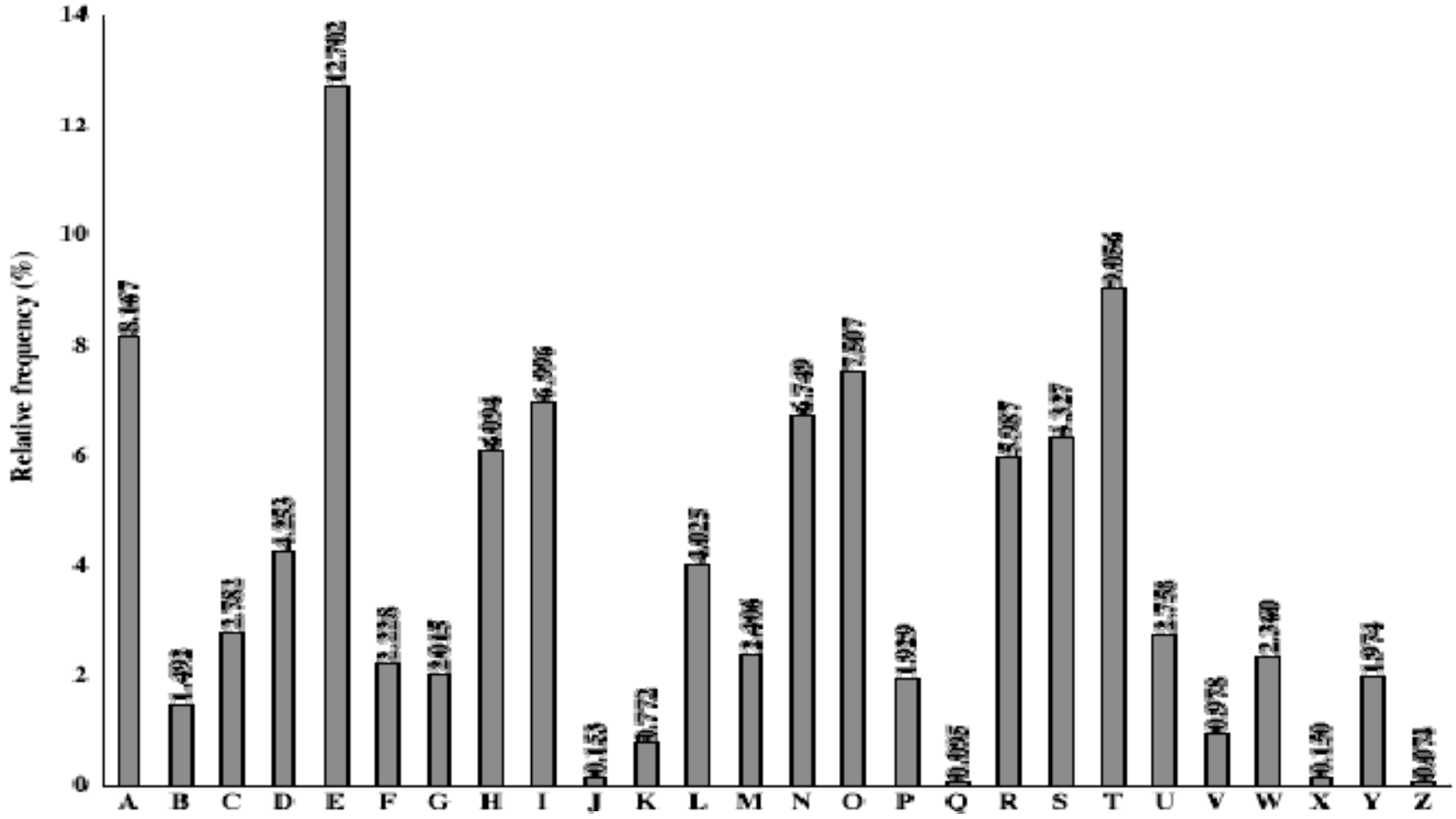
Key: Shift 3 places toward the front of the alphabet

Thus **STEALTHEHOTDOG**
Is encrypted as **pqbxiqebelqald**

Today easily deciphered by analyzing frequency of letters

Letter frequency is sometimes called ***Side Information (SI)***

Relative Frequency of Letters in English Text



Example
Monoalphabetic Substitution Cipher (S)

First recorded use is by Arabic historians

Monoalphabetic substitution stream cipher

For each plaintext character substitute a ciphertext character

e.g., A→#, B→q, C→, D→X, ...*

Here we have ***isomorphic*** substitution of one character for another

Today easily deciphered by analyzing frequency of letters

Example
Polyalphabetic Substitution Cipher (S)

Vigenère stream cipher is good example

The ***key*** is some word or string of characters or numbers – say the word "WHITE"

*Align **A** with **W**: Then B→X…**S**→**O** to encrypt 1st letter*

ABCDEFGHIJKLMNOPQR**S**TUVWXYZ
WXYZABCDEFGHIJKLMN**O**PQRSTUV

*Align **A** with **H**: Then B→I…**T**→**A** to encrypt 2nd letter*

ABCDEFGHIJKLMNOPQRS**T**UVWXYZ
HIJKLMNOPQRSTUVWXYZ**A**BCDEFG

…

Thus	**STEALTHEHOTDOG**
Is encrypted as	**oamtppomaspkwm**
Key:	**WHITEWHITEWHIT**

Here strong letter frequency is reduced. Why?

Analysis is more difficult but still quite possible

Enigma Cipher

Used by Nazi Germany in World War II

Automated by electro-mechanical machine

Used a form of Vigenère cipher followed by complex transposition

It was thought to be very secure

But the Germans were careless in picking keys

Key selection was too regular

As a result, Polish security broke it before Germany invaded Poland in 1939

Marian Rejewski (Polish intelligence)

Enigma Cipher

Poland gave the information to the British and French before the invasion of Poland in Sept. 1939

So for the early part of WWII, the allies could read all the German and Italian encrypted messages

But eventually the Germans started picking random keys

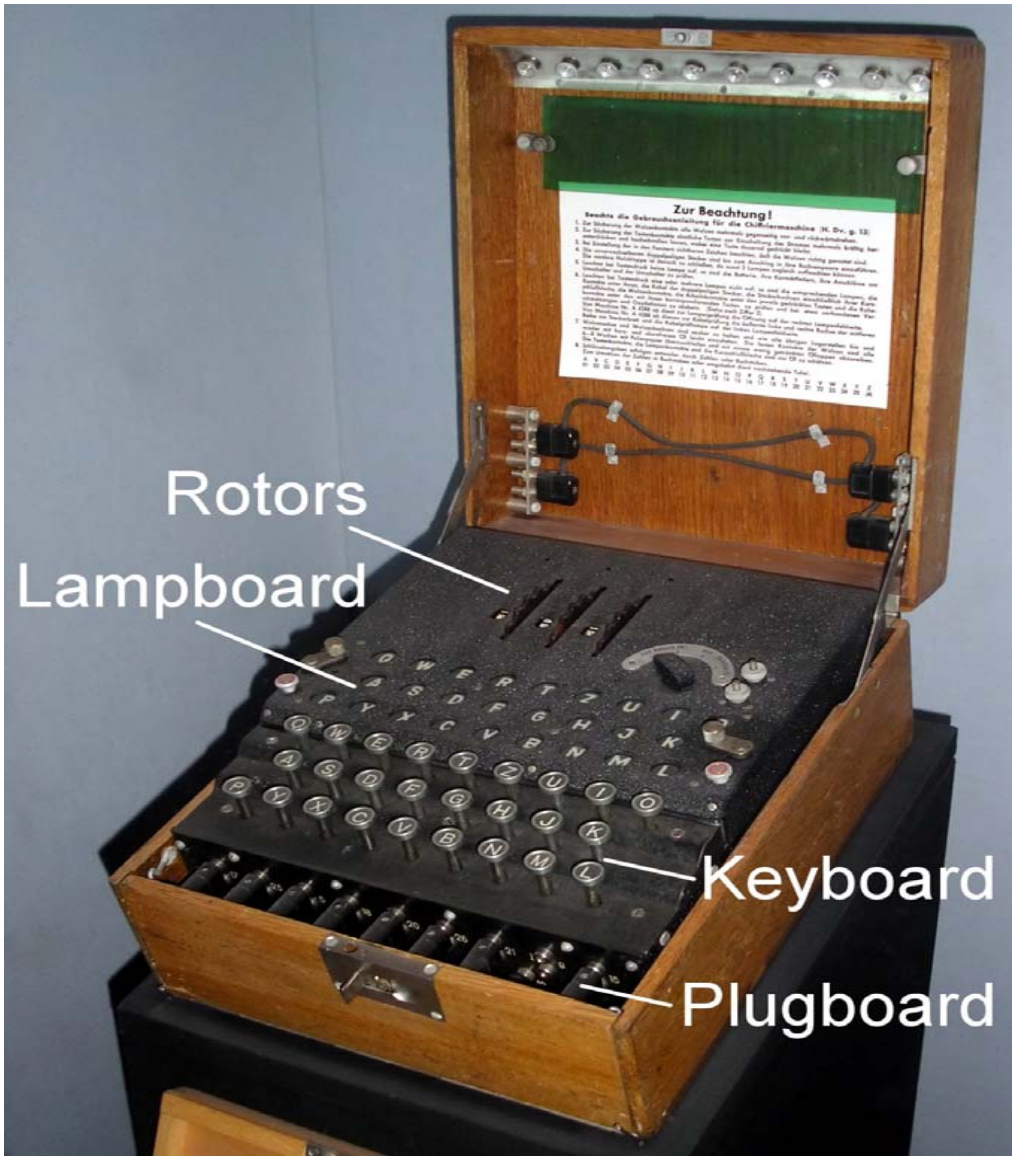
British (especially Alan Turing) developed ability to decrypt German encryption that used random keys

British even built their own Enigma machines

Bletchley Park

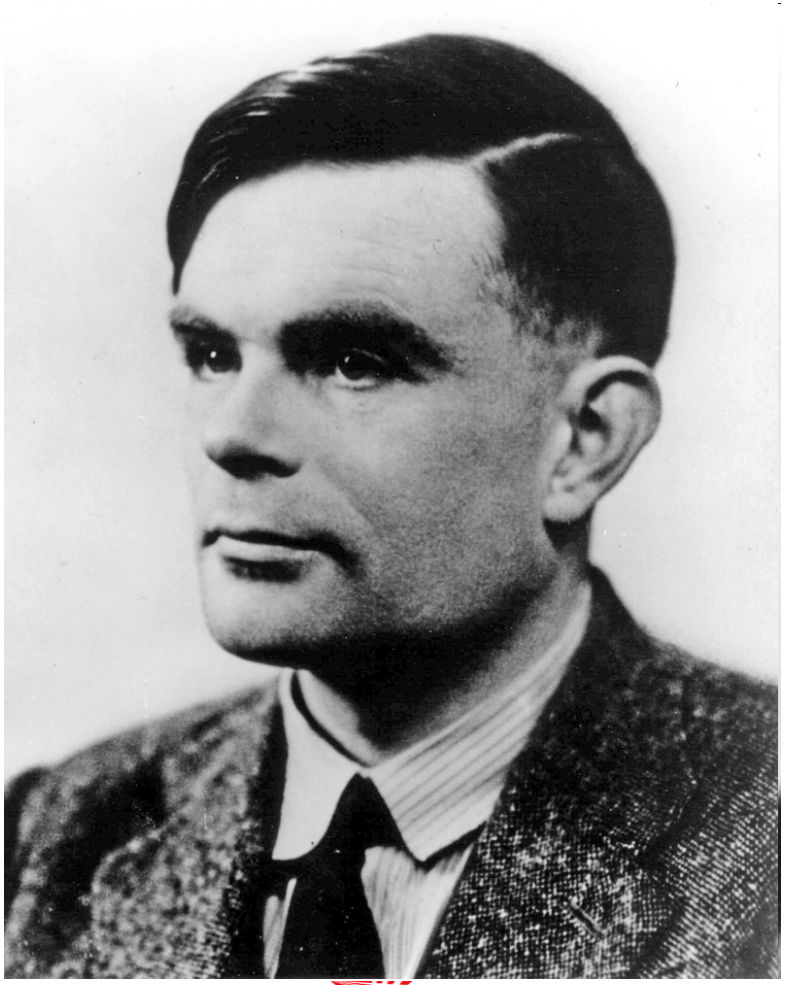
Breaking Enigma was a matter of very sophisticated frequency analysis (SI)

Enigma Machine



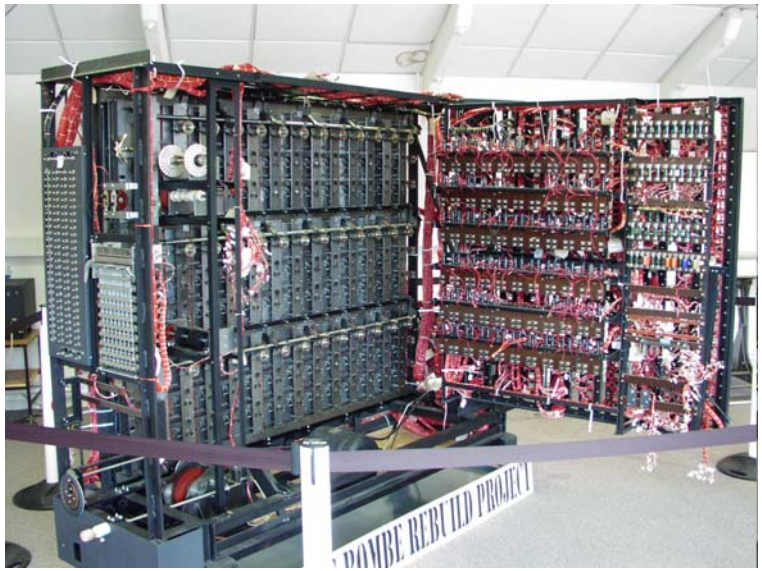
Bletchley Park - Today

Alan Turing



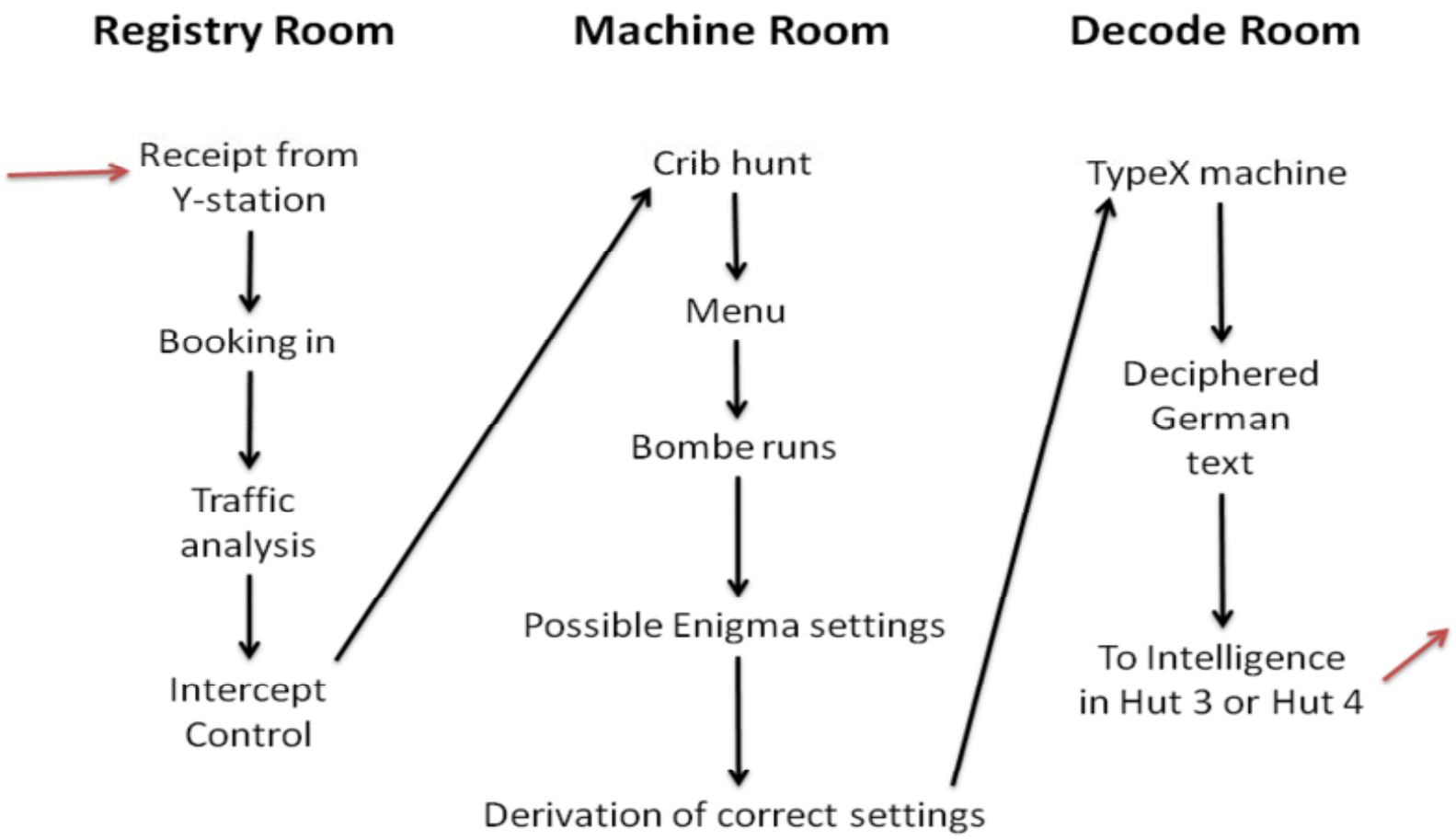
Bletchley Park & bombe

Two cottages in the stable yard at Bletchley Park, where Turing worked in 1939 and 1940.



Replica of a ***bombe*** machine used to decipher German encrypted messages

Code Breaking Process at Bletchley



Symmetric Block Ciphers

Block ciphers process messages into blocks, each of which is then en/decrypted

Sort of like a substitution on very big characters

64-bits or more

Block Cipher Principles

Most symmetric block ciphers are based on a **Feistel Cipher Structure** *(Horst Feistel)*

Feistel's work was based heavily upon the work of Claude Shannon

Block ciphers allow efficient decryption of ciphertext

Block ciphers look like an extremely large substitution

Substitute one block at a time

Note: There are 2^{64} possible encryptions for a 64-bit plaintext block

~ 18,000,000,000,000,000,000 or 18 billion billion

Claude Shannon and Substitution-Permutation Ciphers

In 1949 Claude Shannon (Bell Labs) introduced idea of substitution-permutation (S-P) networks

A "father" of information theory

Modern substitution-permutation product cipher

Communication Theory of Secrecy Systems, BSTJ 1949

These form the basis of modern block ciphers

S-P networks are based on the two primitive cryptographic operations we have seen before:

Substitution (S-box) *(substitution)*

Permutation (P-box) *(transposition)*

Provide ***confusion*** and ***diffusion*** of message

Confusion and Diffusion

Ciphers need to completely obscure the statistical properties of original message

(A one-time pad does this)

More practically, Shannon suggested combining elements to obtain:

Confusion

*Make relationship between the **ciphertext** and the **key** as complex as possible*

Diffusion

*Dissipate the statistical structure of **plaintext** over the entire **ciphertext***

Feistel Cipher Structure

Horst Feistel devised the ***feistel cipher***

Divide plaintext into a series of input blocks

Partition plaintext input block into two halves

Process each block through multiple rounds which

Perform a substitution on left data half of a block based on a function of right half & a subkey

Then have a permutation by swapping halves

This scheme is effective at implementing Shannon's substitution-permutation or confusion-diffusion concepts

Feistel Cipher Design Principles

Block size

Increasing size improves security, but slows cipher

Key size

Increasing size improves security, makes exhaustive key searching harder, but may slow cipher

Number of rounds

Increasing number improves security, but slows cipher

Subkey generation

Greater complexity can make analysis harder, but slows cipher

Round function

Greater complexity can make analysis harder, but slows cipher

Fast software en/decryption & ease of analysis

These are more recent concerns for practical use and testing

DES
Block Cipher

DES (Data Encryption Standard)

In 1970 American National Bureau of Standards (now NIST) generated a request for proposals (RFP) for a computer-based encryption standard

Goal: Secure from known algorithm and known plaintext attacks

IBM's Lucifer algorithm was chosen (Horst Feistel from IBM invented it.)

Modified to create DES

DES was adopted by NIST in 1977

DES (Data Encryption Standard)

Message is a binary bit string

Message is broken into 64-bit blocks

Each block encrypted using 56-bit secret key

Arranged as eight 7-bit pieces

Artificially expanded to 64 bits by adding one parity bit to each piece of the key

Compromise with NSA who was worried about not being able to decrypt messages

$C = E_k[P]; \quad P = D_k[C] = D_k[E_k[P]]$

E and D are closely related

K is the same for encryption and decryption

DES (Data Encryption Standard)

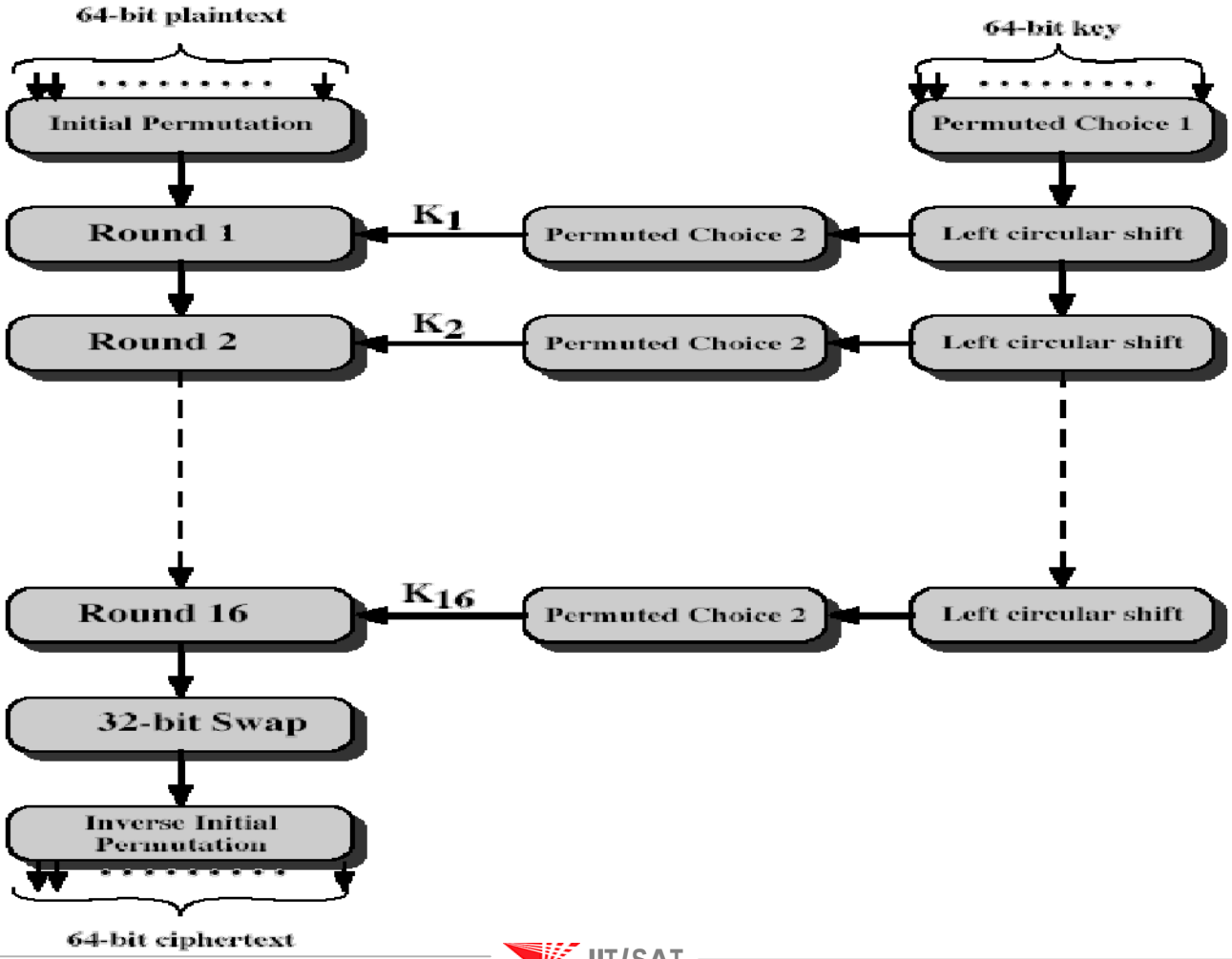
Complex algorithm on each block

- *16 rounds of encipherment/decipherment on each block*
- *56-bit key used to generate 16 sub keys of 48 bits each*
- *Each round uses one 48-bit sub key derived from the 56-bit key*

Process is symmetric: almost same process for encoding and decoding

- *In decipherment, subkeys are used in reverse order*

DES Encryption Process Flow



Initial Permutation

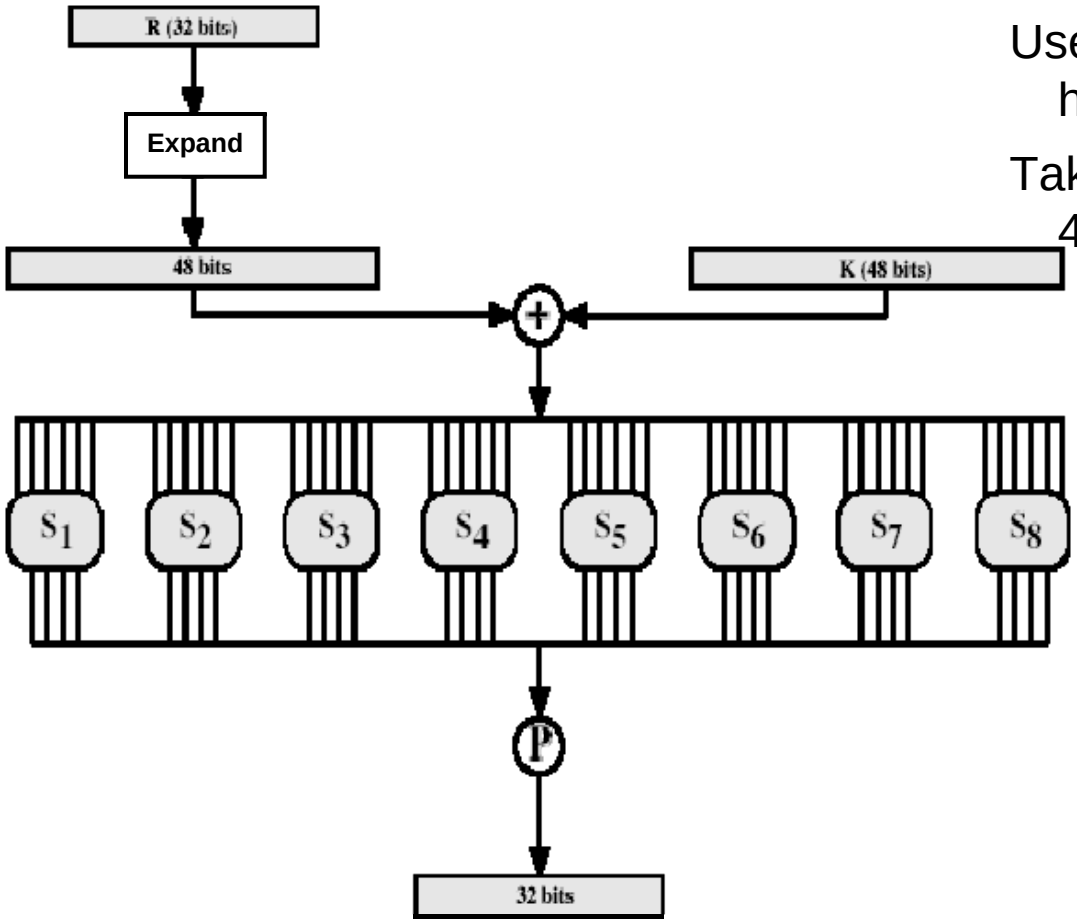
Reorders the input data bits

Even bits to LH half

Odd bits to RH half

Easy in hardware

DES Operation for Each Round



Uses two 32-bit L & R halves

Takes 32-bit right half and 48-bit subkey and:

- *Expands right half to 48-bits using perm E*
- *Adds it to the subkey*
- *Passes through 8 S-boxes to get 32-bit result*
- *Finally permutes this using 32-bit perm P*

Substitution Boxes S

Eight S-boxes which map 6 bits to 4 bits

Each S-box is actually 4 little 4 bit boxes

*Outer bits 1 & 6 (**row** bits) select one row*

*Inner bits 2-5 (**col** bits) are substituted*

Result is 8 groups of 4 bits, or 32 bits

Row selection depends on both data & key

Feature known as autoclaving (autokeying)

DES Key Schedule

The *Key Schedule* forms the subkeys used in each round

Consists of:

- *Initial permutation (PC1) of the key which selects 56-bits in two 28-bit halves*
- *16 stages consisting of:*
  - Selecting 24-bits from each half
  - Permuting them using permutation PC2 for use in a function ***f***,
  - Rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K

DES Decryption

Decrypt must unwind steps of encryption

With DES, we do encryption steps again using subkeys in reverse order (SK16 … SK1)

The initial permutation step of decryption undoes final permutation step of encryption

1st decrypt round with SK16 undoes 16th encrypt round

…

16th round with SK1 undoes 1st encrypt round

Then the final permutation undoes encryption's initial permutation

This recovers the original plaintext value

An Evaluation of DES

Some DES Observations
Avalanche Effect

Avalanche is a highly desirable property of encryption algorithms

A change of a single input bit (plaintext or key) results in changing many output (ciphertext) bits

Avalanche makes impossible the process of "homing-in"

Iteratively guessing keys and getting partial results

DES exhibits strong avalanche

*A change of **one** plaintext or key bit results in changing approx **half** of the ciphertext bits*

Some DES Observations
DES Weak Keys

We want $P = D_k[E_k[P]]$

where D_k is E_k with subkeys applied in reverse order

But for certain DES keys, $P = E_k[E_k[P]]$

K =`0101010101010101`

`1F1F1F1F1F1F1F1F`

`FEFEFEFEFEFEFEFE`

For these 3 keys

- Certain DES process intermediate results contain either all 0s or all 1s, and
- Shifting doesn't change the subkeys

DES Observations
DES Semi-weak Keys

$P = E_{k1}[E_{k2}[P]]$

K1 and K2 are such that octets are simply transposed

e.g., K1 = `01FE01FE01FE01FE`

K2 = `FE01FE01FE01FE01`

...

There are 6 semi-weak keys

For these keys, certain intermediate results contain either all 0s, all 1s, or alternate 0s and 1s

Shifting either doesn't change the subkeys or 1s complements the subkeys

DES Observations
DES 56-bit Key Too Short

Considering only exhaustive attacks

56 bit key yields 2^{56} or $7.2x10^{16}$ keys

In 1998 the Electronic Frontier Foundation's *DES Deep Crack* machine demonstrated $90x10^9$ DES keys per second

Was able to crack a DES encoded message in 9.2 days

In 1999 a machine was demonstrated to test $2.5x10^{11}$ keys per second (called Deep Crack Plus)

Was able to crack a DES encoded message in 3.3 days

Guess what NSA can do today!!

Strength of DES – Timing Attacks

Attacks the actual implementation of the cipher

Uses knowledge of consequences of implementation to derive knowledge of some/all subkey bits

Specifically uses the fact that calculations can take varying times depending on the value of inputs to it

Particularly effective on smartcards

Strength of DES – Analytic Attacks

Today several analytic attacks on DES exist

These utilize deep structures of the cipher

- *By gathering information about encryptions*
- *Can eventually recover some/all of the sub-key bits*
- *If necessary then exhaustively search for the rest*

Generally these are statistical attacks include

- *Differential cryptanalysis*
- *Linear cryptanalysis*
- *Related key attacks*

Differential Cryptanalysis

One of the most significant (public) advances in cryptanalysis

Murphy, Biham & Shamir made public in 1990

Known by NSA in 1970's

Powerful method to analyse block ciphers

Used to analyze most current block ciphers with varying degrees of success

DES reasonably resistant to it

Differential Cryptanalysis

Process

1. *Provide new plaintext that is different from the original plaintext where the differences are known*
2. *Observe the ciphertext output differences and their probabilities*
3. *Then find instances of some higher probability input/output difference pairs*

By iterating on above the cryptanalyst can infer the subkey that was used in the round

Then must iterate process over many rounds (with decreasing probabilities)

Linear Cryptanalysis

Another recent cryptanalysis approach is Linear Cryptanalysis

Also a statistical method

Must be iterated over rounds, with decreasing probabilities

Developed by Matsui et al in early 90's

Based on finding linear approximations

Attacks DES with up to 2^{47} known plaintexts

Still infeasible in practice

Block Chaining

Modes of Operation

Block ciphers encrypt fixed size blocks

eg. DES encrypts 64-bit blocks, with 56-bit key

If the plaintext consists of an arbitrary amount of information to encrypt, how do we use block ciphers?

Four mechanisms were defined for DES in ANSI standard ***ANSI X3.106-1983 Modes of Use***

Increased to 5 mechanisms for DES and AES

Modes of Operation

ECB: Electronic Code Book

CBC: Cipher Block Chaining

CFB: Cipher FeedBack

OFB: Output FeedBack

CTR: CounTeR

Electronic Code Book (ECB)

ECB (Electronic Code Book) [Doesn't really chain]

Each block of plaintext P is encoded independently

$C_i = E_{K1}(P_i)$ *where i is the* i_{th} *block of P*

A block can be thought of as a value that is substituted

Like a codebook value

A block of P will always be encrypted into the same C

In highly structured long messages, C may be vulnerable

Effective for encrypting single values

Comments About ECB

Repetitions in message may be detectable in ciphertext

Very much so:

- If data is aligned with message block
- With data such as graphics
- With messages that change very little, which become a code-book analysis problem

ECB's weakness are due to encrypted message blocks being independent

No avalanche effect between blocks

Main use is in sending a few blocks of data

Cipher Block Chaining (CBC)

Message is broken into blocks as with ECB

But these blocks are linked together in the encryption operation

Each previous cipher block is chained with current plaintext block

An Initial Vector (IV) is used to begin the process

$$C_i = E_{K1}(P_i \oplus C_{i-1}) \quad \text{and} \quad C_{-1} = IV$$

At end of message if the last block is short

Pad either with known non-data value (e.g nulls), or

Pad last block with count of pad size

Uses: bulk data encryption and authentication

Comments About CBC

Avalanche effect is significant across blocks

- *Each ciphertext block depends on **all** message blocks before it*
- *Identical blocks will encode differently as a function of their location in a message*
- *A change in the message affects all ciphertext blocks after the change as well as the original block*

Sender & receiver both need to know IV

- *However if IV is sent in the clear, an attacker can change bits of the first block, and change the IV to compensate*
- *Hence either IV must be a fixed known value or it must be sent encrypted in ECB mode before rest of message*

Cipher FeedBack (CFB)

This should have been named CFF (Cipher Feed Forward)

This is a combination of stream and block-chaining encryption

Plaintext is added to the output of the block cipher

Result is fed forward for encryption of the next block

Standard allows any number of bits (1,8 or 64 or whatever) to be fed

Denoted CFB-1, CFB-8, CFB-64 etc

Is most efficient is to use all 64 bits (CFB-64)

$$C_i = P_i \oplus E_{K1}(C_{i-1}) \qquad C_{-1} = IV$$

Uses: stream data encryption, authentication

Comments About CFB

Appropriate when data arrives in bits

Commonly used in stream mode encryption

Limitations:

- *Must temporarily pause while do block encryption after every n-bits*
- *Noisy communication channels*
  - Errors caused perhaps by noisy communication channels propagate for several blocks after the error
  - Solutions: Error-free channels such as TCP or OFB mode

Output FeedBack (OFB)

This is combination of stream & block-chain encryption

Plaintext is XORed to the output of the block cipher in order to get the ciphertext block

This output is then fed back

$$C_i = P_i \oplus O_i$$

Independent of P

$$O_i = E_{K1}(O_{i-1}) \qquad O_{-1} = IV \qquad O_0 = E_{K1}(IV)$$

Feedback (O_i) is independent of message

Can be computed in advance

Uses: stream encryption over noisy channels

Comments About OFB

Used when errors in feedback are a problem or where one needs to encrypt before message is available

Superficially similar to CFB

But feedback is from output of cipher and is independent of ciphertext message

Hence you must **never** reuse the same sequence (key+IV)

Sender and receiver must remain in sync, and some recovery method is needed to ensure this occurs

Comments About OFB

Originally specified with m-bit feedback in the standards

Research has shown that only **OFB-64** should be used

A variation of a Vernam cipher

Counter (CTR)

This is a “newer” mode *(though proposed early on)*

Similar to OFB but encrypts counter value rather than any feedback value

Must have a different key & counter value for every plaintext block (never reused)

$$C_i = P_i \oplus O_i$$

$$O_i = E_{K1}(i)$$

Uses: high-speed network encryptions

Comments About CTR

Efficiency

- *Can do parallel encryptions in advance of need*
- *Good for bursty high speed links*

Random access to encrypted data blocks

Provable security (good as other modes)

But must ensure never reuse key/counter values

- *Otherwise could break (cf OFB)*

Triple DES

Triple DES

3DES

Uses the same algorithm as DES

Applies the algorithm three times

Uses a different key each time

$C = E_{k1}[D_{k2}[E_{k3}[P]]]$

Effectively a key of 168 bits - maybe???

We'll discuss effective key size later

Pros & Cons

Uses same software as DES

Block size is smallish (64 bits), resulting in insecurities

Three times a slow as DES

AES

Advanced Encryption Standard

History

A replacement for DES was clearly needed

Theoretical attacks exist that can break it

Exhaustive key search attacks have been demonstrated

Can use Triple-DES – but slow and has small blocks

US NIST issued call for ciphers in 1997

15 candidates accepted in Jun 98

5 were short-listed in Aug-99

Serpent	*Ross Anderson, Eli Biham, Lars Knudsen*
Twofish	*Bruce Schneier et al*
Rijndael	*Vincent Rijmen, Joan Daemen*
MARS	*Don Coppersmith (IBM). Derived from Feistel*
RC6	*Ron Rivest (Rivest Cipher 6)*

History

	Rijndael	Serpent	Twofish	MARS	RC6
Ease of Implementation	Good	Good	OK	Low	Low
Hardware Performance	High	High	OK	Low	OK
Software Performance	High	Low	Low	Medium	Medium
Embedded Sys, CCs, Smart Cards, SIMS…	Good	Good	OK	Low	Low
Overall Design	OK	Poor	Good	OK	Poor
Overall Security	OK	Good	Good	Good	OK

Rijndael's *(RhineDahl)* proposal was selected as the AES in Oct-2000

Issued as FIPS PUB 197 standard in Nov-2001

AES
Advanced Encryption Standard

AES is Advanced Encryption Standard

128-bit block size

Keys can be 128, 192, or 256 bits

Approved by NIST in November 2001

Today AES is used all over

DES and 3DES not used too much any more

AES variant designations

AES-128, AES-192 and AES-256

AES
Advanced Encryption Standard

Encryption is accomplished as a series of "rounds"

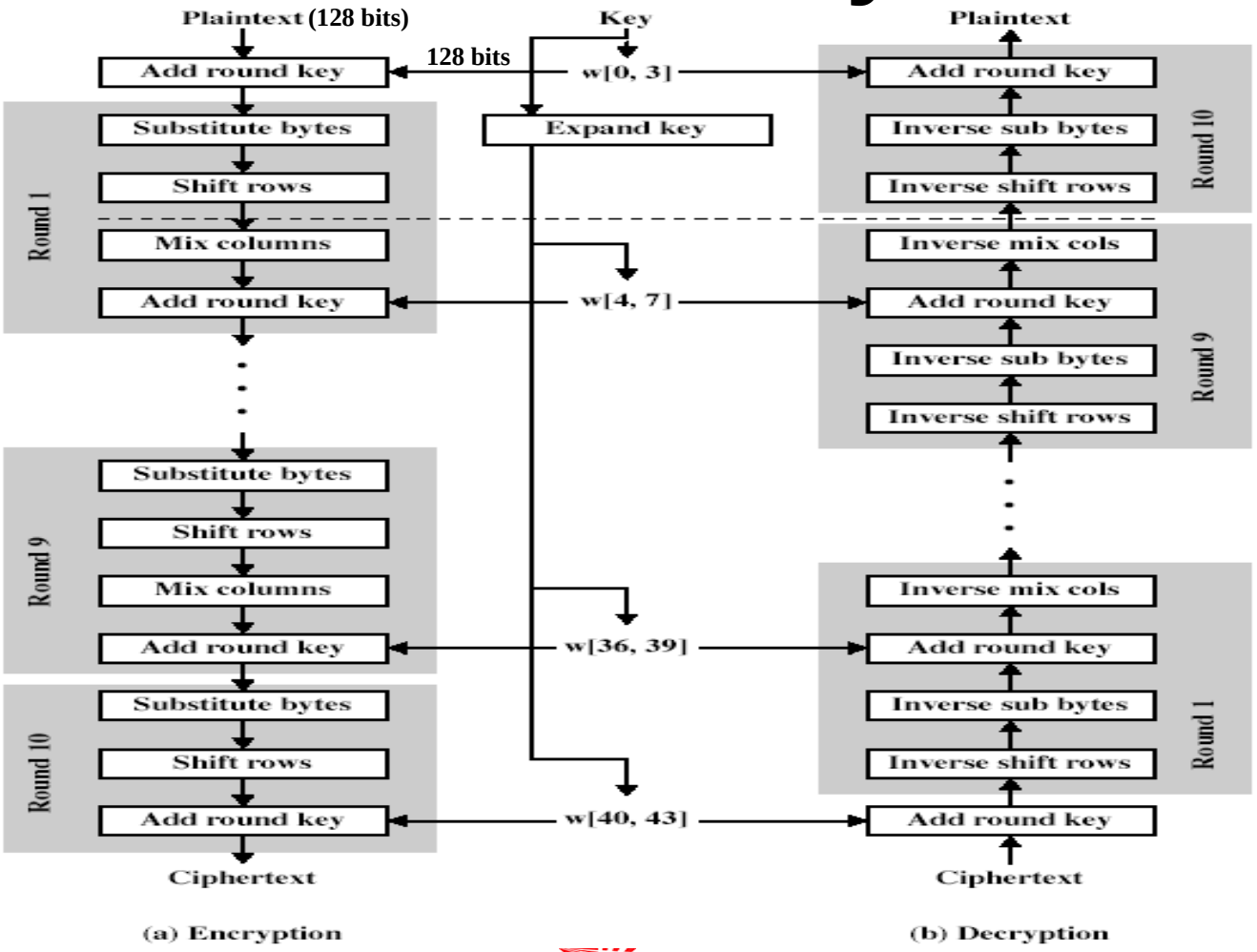
The number of rounds varies

- 10 rounds for 128-bit keys
- 12 rounds for 192-bit keys
- 14 rounds for 256-bit keys

The key is used to create a set of 44 32-bit words, **w[i]**

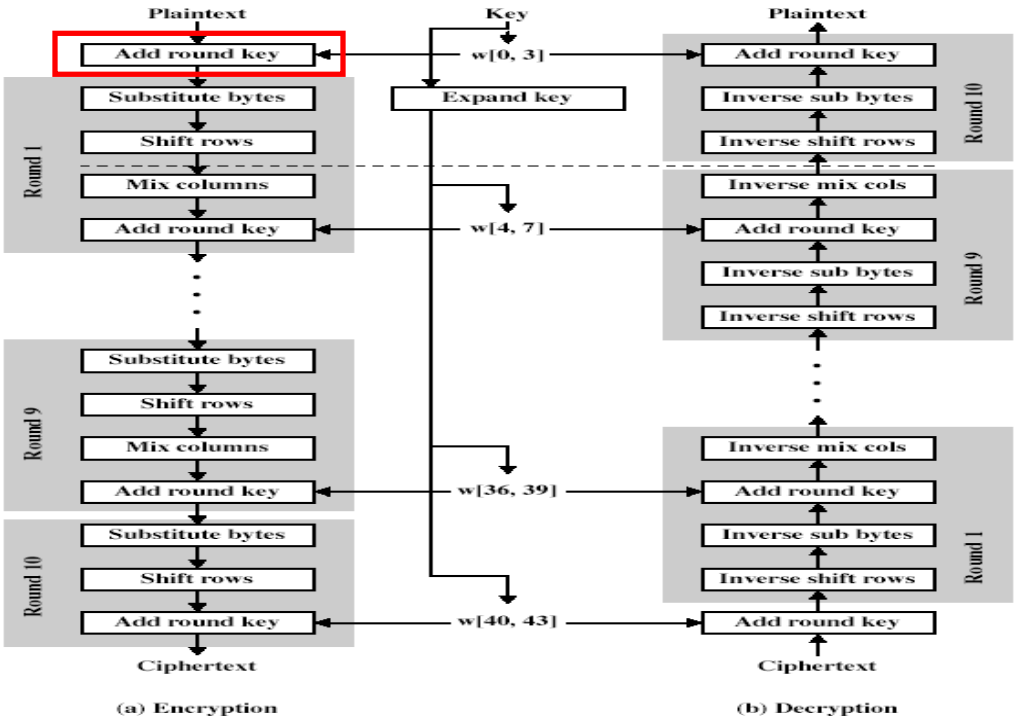
Four of these words (128 bits) are used as the key for each round of processing

AES Process Diagram for 128-bit Keys



Insert Rounds Key w[0-3]

XOR the plaintext with
the 1st set of four 32-bit
derived key words



Byte Substitution

A simple substitution of each byte using an S-Box

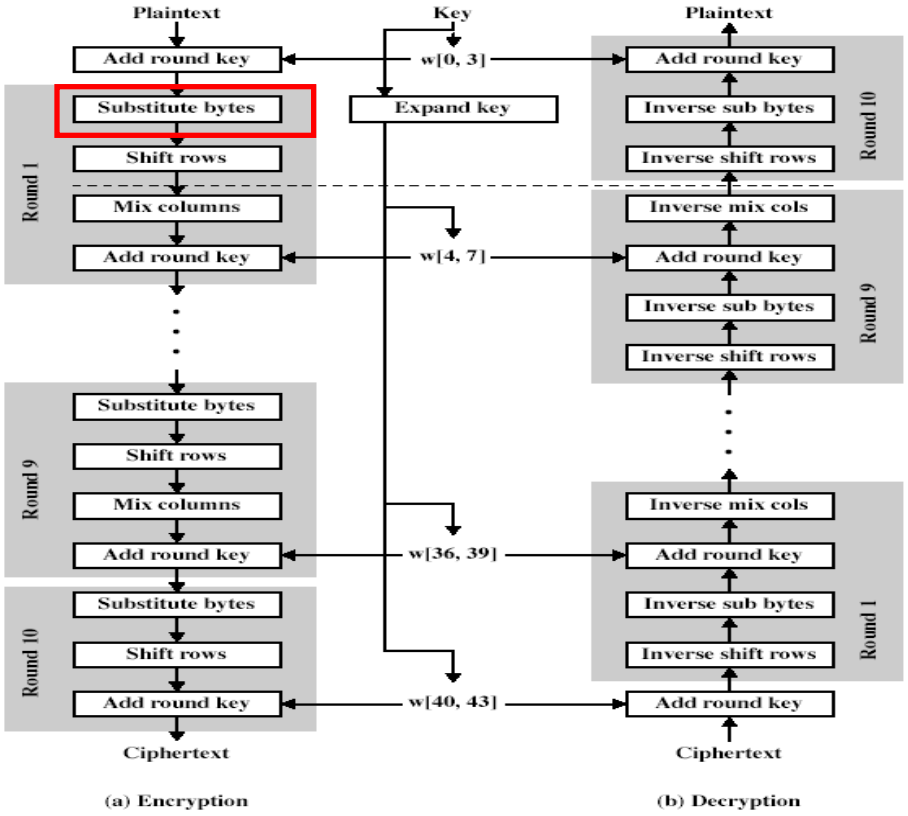
S-Box is a table of 16x16 bytes containing a permutation of all 256 8-bit values

Each byte is replaced by byte in row (left 4-bits) & column (right 4-bits)

*eg. A byte with the value **95** is replaced by row 9 col 5 byte which is the value **2A***

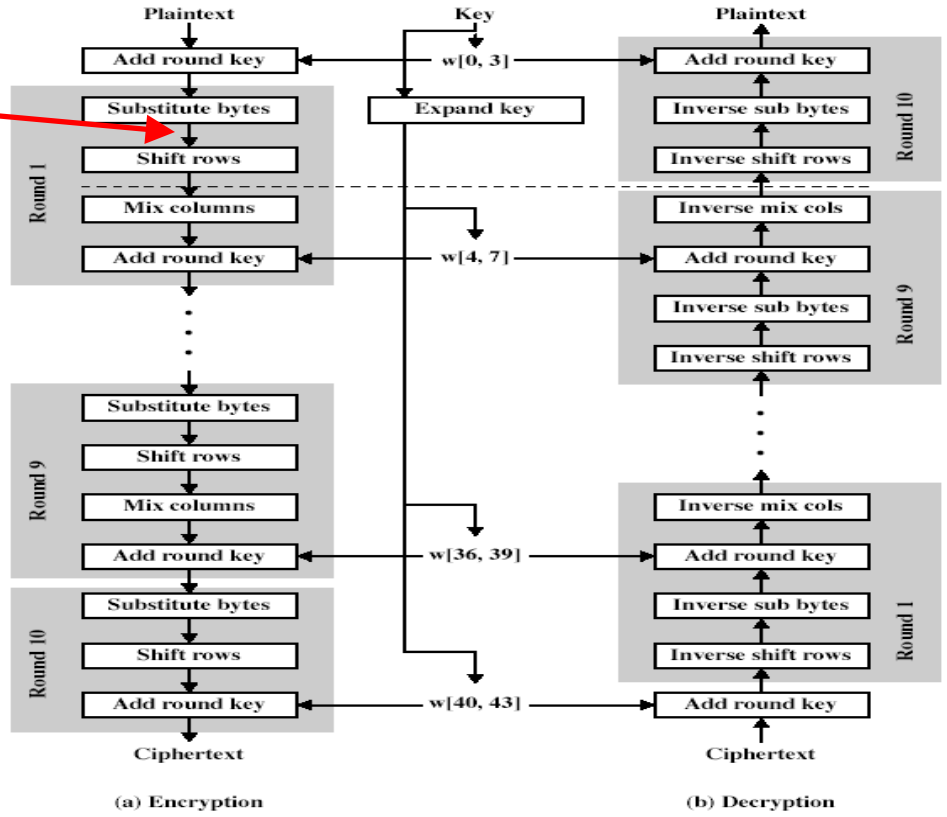
S-box is constructed using a defined transformation of the values

Designed to be resistant to all known attacks



After Byte Substitution

After the substitution, we have a 4x4 matrix of bytes comprising the 128 bits of the block



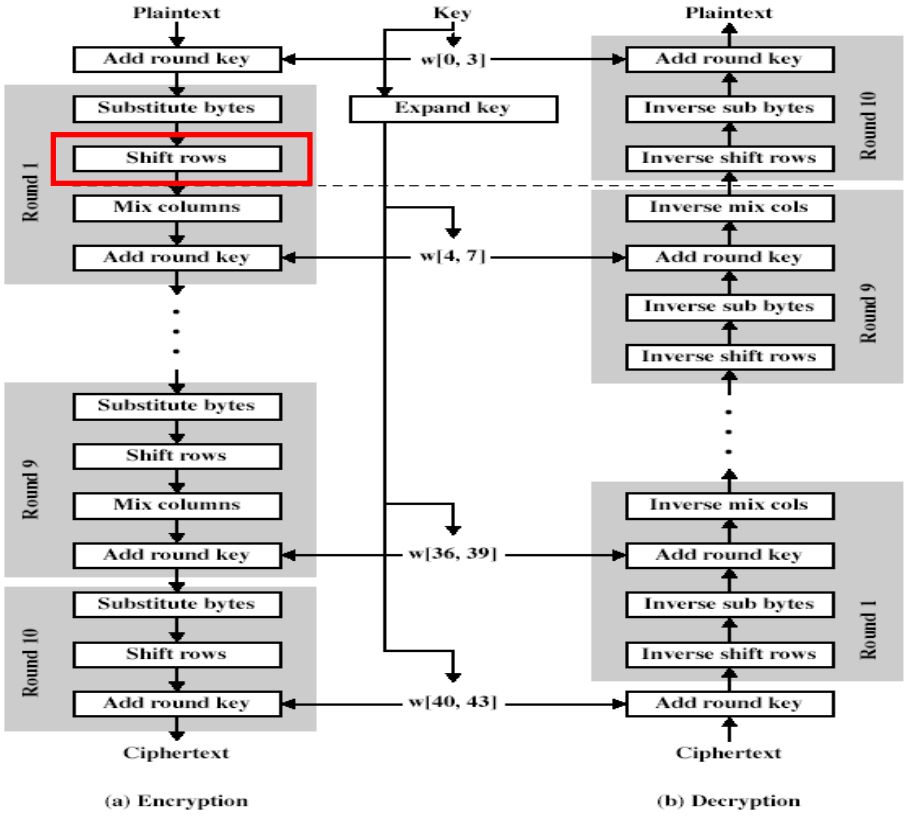
Shift Rows

A circular byte shift in each row of the matirx

- *1st row is unchanged*
- *2nd row does 1 byte circular shift to left*
- *3rd row does 2 byte circular shift to left*
- *4th row does 3 byte circular shift to left*

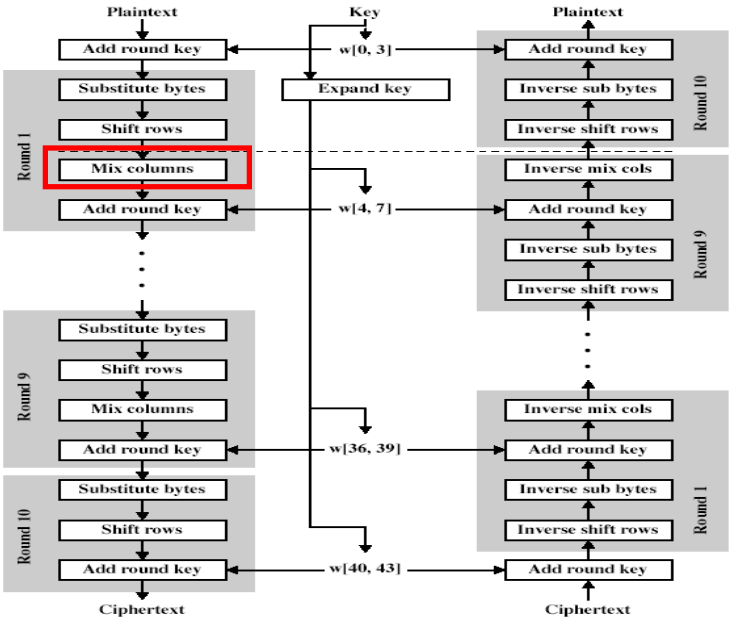
Note

- *Decrypt does shifts to right*
- *Since state is processed by columns, this step permutes bytes between the columns*



Mix Columns

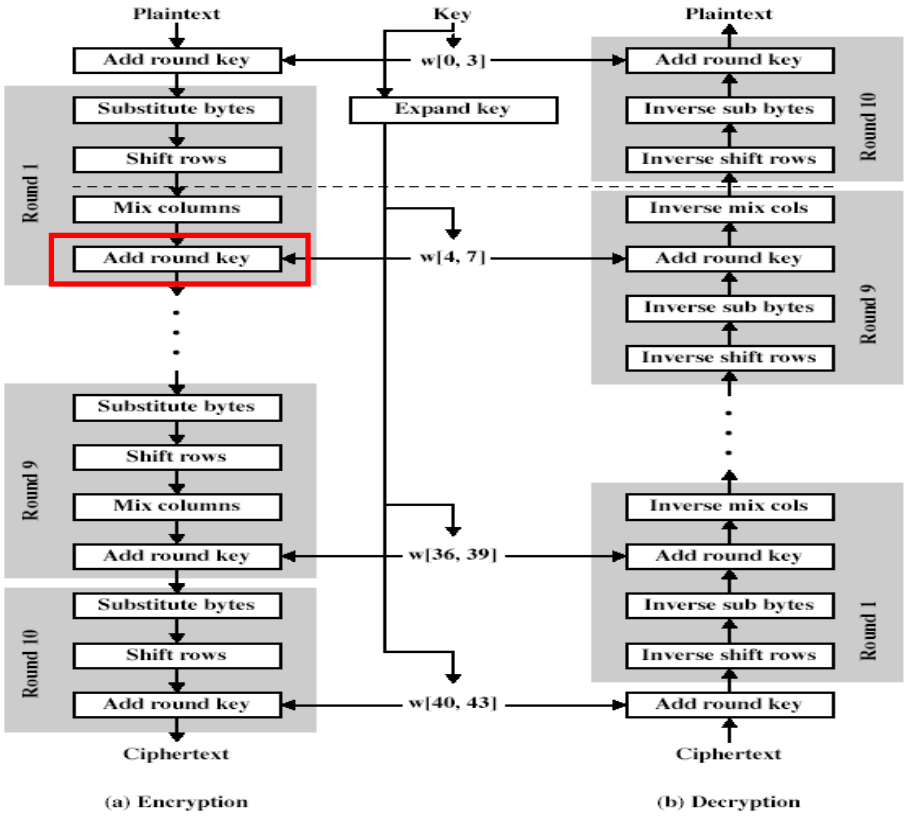
- Each column is processed separately
- Each byte is replaced by a value dependent on all 4 bytes in the column
- Effectively a matrix multiplication using the prime polynomial $m(x) = x^8+x^4+x^3+x+1$



$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Add Next Round Key

XOR the result of the matrix multiplication with 128-bits of the next round key



Iterate

1st round is now done

Iterate through 9 more rounds

AES Key Expansion

Takes 128-bit key and expands into array of 44 32-bit words

Start by copying key into first four 32-bit words

Then loop, creating words that are a function of the values the previous word & the word 4 places back

In 3 of 4 cases just XOR these together

Every 4th word is created as fp;;pws

S-box + rotate + XOR constant of previous before XOR together

Designed to resist known attacks

AES Decryption

AES decryption is almost identical to encryption

Steps are done in reverse

AES Efficiency*

Can define an equivalent inverse cipher with steps in same order as for encryption

Uses inverse of each step and different key schedule

Results in a more efficient implementation

It is possible to do this and get the same result

Swap byte substitution & shift rows

No effect since AES works with bytes

Swap mix columns & add (tweaked) round key

Result is the same if we inverse column mixing and do this 1st to the round keys

More efficient but decryption is a bit more complex

Implementation Aspects*

Can efficiently implement on 8-bit CPU

- *Byte substitution works on bytes using a table of 256 entries*
- *Shift rows is simple byte shifting*
- *Add round key works on byte XORs*
- *Mix columns requires only matrix multiply using GF(2^8) which is simpler than general matrix multiplication*
  - GF refers to Évariste Galois who founded Field Theory
  - GF(2^8) or GF(28) has the following properties
    - *Only works on bytes*
    - *Addition is simple: just XOR. No bit carries are needed*
    - *Can be pre calculated and used in a table lookup*
    - *For AES, only need to multiply by 7 constants*
      - 0x01, 0x02, 0x03, 0x09, 0x0b, 0x0d, and 0x0e

Implementation Aspects

Can efficiently implement on 32-bit CPU

Redefine steps to use 32-bit words

Can pre-compute 4 tables of 256-words

Then each column in each round can be computed using 4 table lookups + 4 XORs

At a cost of 16Kb to store tables

Designers believe this very efficient implementation was a key factor in its selection as the AES cipher

AES Vulnerabilities

AES-128 key can be determined in 2^{128} tries. Why?

AES-256 key can be determined in how many tries?

So exhaustive attacks are not considered vulnerabilities

A vulnerability is considered to exist if

The key or plaintext can be determined with significantly less effort than that of exhaustive attacks

AES Vulnerabilities
Not Necessarily of Key Recovery

AES-128 and its variants have been shown to be vulnerable

Side Information Attack

If software runs on same system as AES code, gets key in a few seconds

Others

AES-192 & 256 Vulnerability

Related Keys Attack

Monitor ciphertext created by several different but mathematically related keys

IDEA

International Data Encryption Algorithm

IDEA

IDEA is International Data Encryption Algorithm

First 128-bit key widely used

Used by browsers and PGP

Developed by Swiss

Uses 64-bit block size

Can use 4 different block chaining methods

Used in several public encryption schemes that are applied to email

Assign06

Read

Stallings & Brown Chapters 2 & 20

Chap 20 & 22(section 22.3 (PKI)) in a couple of weeks

Problems

1. Review Questions 20.1 – 20.10, p. 651

2. Problems 20.6 – 20.8, p. 652

Submit both items 1 and 2 as a single document

Must be in ***.doc*** format