Cryptography

CSE 365 – Information Assurance
Spring 2020

Adam Doupé
Arizona State University
http://adamdoupe.com

Cryptography

- Derived from the Greek words for “hidden, secret” and “writing”
- How to keep information secret or hidden?

Terminology

- Encryption
  – Process of transforming a message such that its meaning is concealed
- Decryption
  – Process of transforming an encrypted message back into original form

Terminology

- Cryptosystem
  - A system that describes how to encrypt or decrypt messages
- Plaintext
  - Message in its original form
- Ciphertext
  - Message in its encrypted form
- Cryptographer
  - Invents encryption algorithms
- Cryptanalyst
  - Breaks encryption algorithms or implementations

Security Benefits of Cryptography

- Confidentiality
- Integrity
- Authentication (as we will see)
- Non-repudiation

Cryptosystem

- Quintuple ($\mathcal{E}$, $\mathcal{D}$, $\mathcal{M}$, $\mathcal{K}$, $\mathcal{C}$)
  - $\mathcal{M}$ set of plaintexts
  - $\mathcal{K}$ set of keys
  - $\mathcal{C}$ set of ciphertexts
  - $\mathcal{E}$ set of encryption functions *e*: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
  - $\mathcal{D}$ set of decryption functions *d*: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

Caesar Cipher

"If he had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others."

- Suetonius, *Life of Julius Caesar* 56

Caesar Cipher

- $\mathcal{M}$ = { sequences of letters }
- $\mathcal{K} = \{ i \mid i$ is an integer and $0 \leq i \leq 25 \}$
- $\mathcal{E} = \{ E_k \mid k \in \mathcal{K}$ and for all letters m,

  $E_k(m) = (m + k) \bmod 26 \}$
- $\mathcal{D} = \{ D_k \mid k \in \mathcal{K}$ and for all letters c,

  $D_k(c) = (26 + c - k) \bmod 26 \}$
- $\mathcal{C} = \mathcal{M}$

Attacks

- *Adversary* is the person who wants to break the cryptosystem
  - Assume adversary knowns the algorithm used, but not the key
    - Is this a realistic assumption?
- Adversary capabilities
  - *ciphertext only*
  - *known plaintext*
  - *chosen plaintext*

Basis for Attacks

- Mathematical attacks
  - Finding flaws by analyzing the underlying mathematics of the cryptosystem
- Statistical attacks
  - Make assumptions based on the underlying language
  - Examine ciphertext, correlate properties with the assumptions
- Implementation attacks
  - Implementation of cryptosystem introduces a flaw that is not in the mathematics of the cryptosystem

Classical Cryptography

- Sender and receiver share common key
  - Keys may be the same, or trivial to derive from one another
  - Called *symmetric cryptography*
- Two basic types
  - Substitution ciphers
  - Transposition ciphers
  - Combinations are called *product ciphers*

Substitution Ciphers

- Change characters in plaintext to produce ciphertext
- Ceasar cipher
  - `HELLO WORLD`
  - Change each letter to the third letter following it (X -> A, Y -> B, Z -> C, …)
    - Key is 3 or written as a letter ‘D’
  - `KHOOR ZRUOG`

Attacking the Caesar Cipher

- Exhaustive search
  - Try all possible keys!
- Statistical analysis
  - Compare to 1-gram model of English

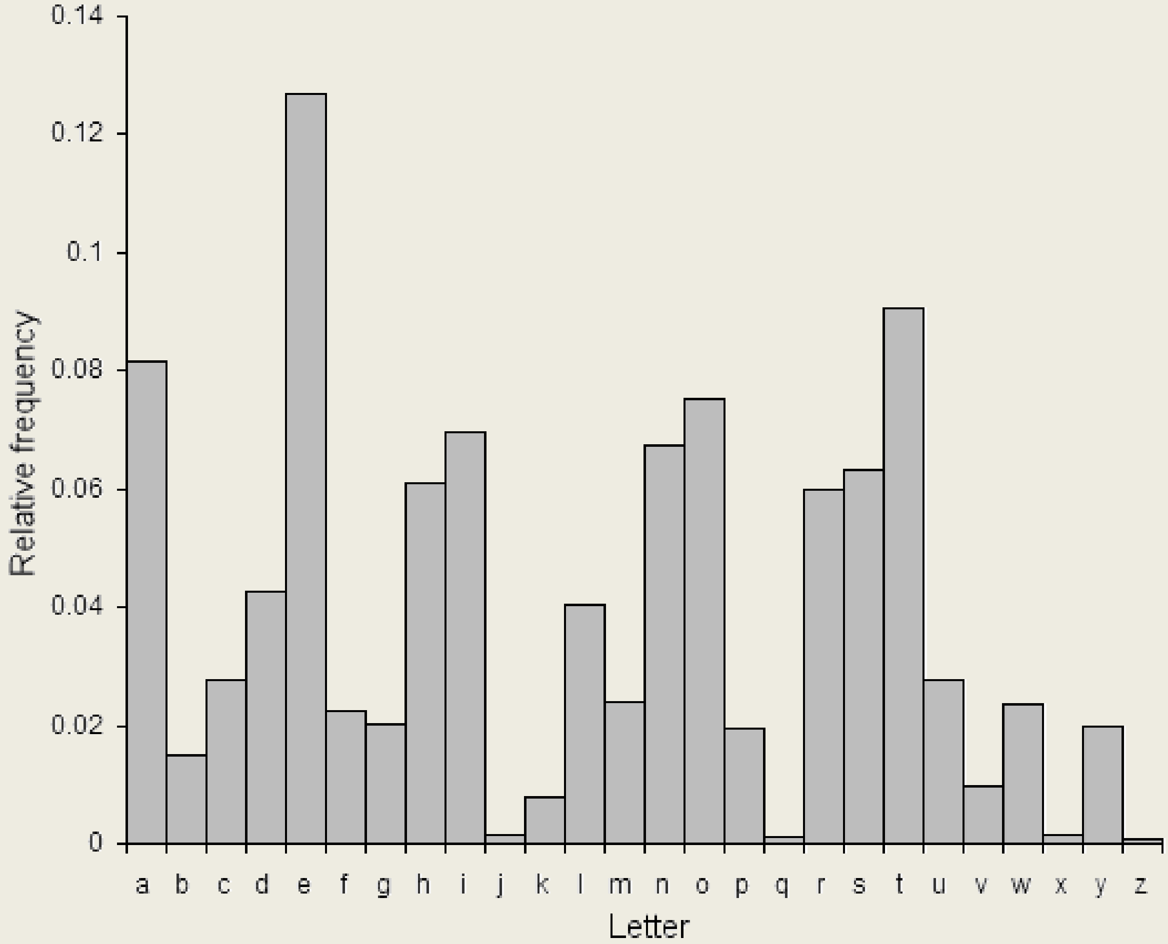
Attacking the Caesar Cipher Example

LBHFUBHYQARIREOHVYQLBHEBJAPELCGB

- Compute frequency of each letter in ciphertext

B: 0.15625	H: 0.125	L: 0.09375
E: 0.09375	Y: 0.0625	R: 0.0625
Q: 0.0625	A: 0.0625	V: 0.03125
U: 0.03125	P: 0.03125	O: 0.03125
J: 0.03125	I: 0.03125	G: 0.03125
F: 0.03125	C: 0.03125	

English Character Frequencies



Source: https://commons.wikimedia.org/wiki/File:English-slf.png

Statistical Analysis

- For every possible key, calculate the correlation of frequency of letters in ciphertext with corresponding letters in English
  - $p(x)$ is frequency of character x in English
  - $f(c)$ frequency of character c in ciphertext
  - $\varphi(i) = \Sigma_{0 \le c \le 25} f(c)p(c - i)$

φ(*i*) for 0 <= i <= 25

```
0.053979  23        0.038090  22
0.051841  13        0.037858  4
0.047364  7         0.036610  11
0.046382  20        0.034693  12
0.045911  3         0.033309  17
0.044305  0         0.033170  10
0.044097  16        0.032574  15
0.043745  24        0.032311  2
0.042421  19        0.031901  25
0.041392  14        0.029868  18
0.038844  8         0.028699  5
0.038755  1         0.026693  6
0.038513  9         0.026650  21
```

Breaking the Cipher

- LBHFUBHYQARIREOHVYQLBHEBJAPELCGB
- 23
  - IYECRYEVNXOFOBLESVNIYEBYGXMBIZDY
- 13
  - YOUSHOULDNEVERBUILDYOUROWNCRYPTO
- 7
  - SIOMBIOFXHYPYLVOCFXSIOLIQHWLSJNI

Caesar Cipher Problems

- Key is too short
  - Can be found by exhaustive search
  - Statistical frequencies not concealed well
- Make the key longer!
  - Multiple letters in key
  - Idea is so smooth the statistical frequencies to make cryptanalysis harder

Vigenère Cipher

- Similar idea to Caesar cipher, but use a phrase
- Message
  - `THE BOY HAS THE BALL`
- Key
  - `VIG`

Encipher using Caesar cipher for each letter:

```
key    VIGVIGVIGVIGVIGV
plain  THEBOYHASTHEBALL
cipher OPKWWECIYOPKWIRG
```

Frequency Analysis

- OPKWWECIYOPKWIRG
  - 0.05537 22
    - KLGSSAYEUKLGSENC
  - 0.05150 10
    - YZUGGOMSIYZUGSBQ
  - 0.05027, 4
    - STOAAIGMCSTOAMVK
  - 0.04530, 2
    - QRMYYGEKAQRMYKTI

Vigenère terms

- *Period*
  - – length of the key
- *polyalphabetic*
  - – key has several different letters

Attacking Vigenère Cipher

- Establish period; call it *n*
- Break message into *n* parts, each part being enciphered using the same key letter
- Solve each part, using techniques from breaking Caesar cipher
  – You can leverage one part from another

ADQYS MIUSB OXKKT MIBHK IZOOO

EQOOG IFBAG KAUMF VVTAA CIDTW

MOCIO EQOOG BMBFV ZGGWP CIEKQ

HSNEW VECNE DLAAV RWKXS VNSVP

HCEUT QOIOF MEGJS WTPCH AJMOC

HIUIX

Establish Period

- Kaskski: *repetitions in the ciphertext occur when characters of the key appear over the same characters in the plaintext*
- Example:

```
key    VIGVIGVIGVIGVIGV
plain  THEBOYHASTHEBALL
cipher OPKWWECIYOPKWIRG
```

Note the key and plaintext line up over the repetitions (underlined). As distance between repetitions is 9, the period is a factor of 9 (that is, 1, 3, or 9)

```
ADQYS MIUSB OXKKT MIBHK IZOOO
EQOOG IFBAG KAUMF VVTAA CIDTW
MOCIO EQOOG BMBFV ZGGWP CIEKQ
HSNEW VECNE DLAAV RWKXS VNSVP
HCEUT QOIOF MEGJS WTPCH AJMOC
HIUIX
```

Repetitions in Ciphertext

Letters	Start	End	Distance	Factors
MI	5	15	10	2, 5
OO	22	27	5	5
OEQOOG	24	54	30	2, 3, 5
FV	39	63	24	2, 2, 2, 3
AA	43	87	44	2, 2, 11
MOC	50	122	72	2, 2, 2, 3, 3
QO	56	105	49	7, 7
PC	69	117	48	2, 2, 2, 2, 3
NE	77	83	6	2, 3
SV	94	97	3	3
CH	118	124	6	2, 3

Estimate of Period

- `OEQOOG` is a good starting point
  - Period may be 1, 2, 3, 5, 6, 10, 15, or 30
- Most of the others (7/10) have 2 in their factors
- Almost as many (6/10) have 3 in their factors
- Let’s try period of 2 * 3 = 6

Check our Period Guess

- *Index of coincidence (IC)* is the probability that two randomly chosen letters from ciphertext will be the same
- Precalculated for different periods:

1	0.066	3	0.047	5	0.044
2	0.052	4	0.045	10	0.041

Large 0.038 – (Note 1/26 - random)

Compute IC

- IC = $[n\,(n - 1)]^{-1} \Sigma_{0 \le i \le 25} [F_i\,(F_i - 1)]$
  - where *n* is length of ciphertext and F_i the (integer) number of times character *i* occurs in ciphertext
- In our ciphertext, IC = 0.043
  - Indicates a key of slightly more than 5
  - A statistical measure, so it can be in error, but it agrees with the previous estimate (which was 6)

Split Ciphertext into Alphabets

- AIKHOIATTOBGEEERNEOSAI
  - IC 0.069
- DUKKEFUAWEMGKWDWSUFWJU
  - IC 0.078
- QSTIQBMAMQBWQVLKVTMTMI
  - IC 0.078
- YBMZOAFCOOFPHEAXPQEPOX
  - IC 0.056
- SOIOOGVICOVCSVASHOGCC
  - IC 0.124
- MXBOGKVDIGZINNVVCIJHH
  - IC 0.043

Solve Each Alphabet

- Can be done using techniques to attack Caesar Cipher
- Can also use information from breaking one alphabet or knowledge of English

Frequency Analysis

```
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
1   31004011301001300112000000
2   10022210013010000010404000
3   12000000201140004013021000
4   21102201000010431000000211
5   10500021200000500030020000
6   01110022311012100000030101
```

Letter frequencies are (H high, M medium, L low):

```
    HMMMHMMHHMMMMHHMLHHHMLLLLL
```

Try Decrypting

- First alphabet matches characteristics of unshifted alphabet
- Third alphabet matches if I -> A
- Sixth alphabet matches if V -> A
- Substitute into ciphertext (bold are substitutions)

AD**I**YS **RI**U**K**B O**CK**K**L** MI**GH**K **A**ZO**TO**

E**I**OO**L** **I**F**T**AG **PA**U**E**F V**A**T**A**S**S** CI**IT**W

EOC**NO** E**I**OO**L** **B**M**T**FV **EG**G**O**P C**NE**K**I**

HS**SE**W **N**EC**SE** D**D**AA**A** **R**W**C**XS **AN**S**N**P

H**HE**U**L** QO**NO**F **E**EG**OS** W**L**PC**M** **A**J**E**OC

MIU**A**X

Look For Clues

- **A**J**E** in last line suggests "are", meaning second alphabet maps A into S:

ALIYS **RICK**B O**CKSL** MI**GHS** **A**ZO**TO**

MIOO**L** **INT**AG **PACE**F V**ATIS** CI**ITE**

EOC**NO** **MI**OO**L** **BUT**FV **EGOO**P C**NESI**

HS**SEE** **N**EC**SE** **LD**AA**A** **REC**XS **ANAN**P

H**HECL** QO**NON** **E**EG**OS** **EL**PC**M** **ARE**OC

MICAX

Next Alphabet

- **MICA**X in last line suggests "mical" (a common ending for an adjective), meaning fourth alphabet maps O into A:

  ALIMS **RICKP** O**CKSL** **A**I**GHS** **AN**O**TO** **MIC**O**L**
  INTOG **PACET** V**ATIS** **Q**I**ITE** **EC**C**NO** **MIC**O**L**
  BUTTV **EGOOD** C**NESI** **V**S**SEE** **NS**C**SE** **LDO**A**A**
  RECLS **ANAND** H**HECL** **E**O**NON** **ES**G**OS** **ELD**C**M**
  ARECC **MICAL**

- Can brute force the last alphabet

Got It!

- QI means that U maps into I, as Q is always followed by U:

```
ALIME RICKP ACKSL AUGHS ANATO MICAL
INTOS PACET HATIS QUITE ECONO MICAL
BUTTH EGOOD ONESI VESEE NSOSE LDOMA
RECLE ANAND THECL EANON ESSOS ELDOM
ARECO MICAL
```

Transposition Ciphers

- Rearrange letters in plaintext to produce ciphertext
- Properties
  - Same 1-gram frequencies as English
  - Different n-gram frequencies
  - IC ~ .066

Simple Transposition Cipher

- Break message into blocks of keylength
- Key is transposition of block
  - Example: key(3, 0, 2, 1)
  - `Message: ASUI SAWE SOME`
  - `Encrypt: SIUA AEWS OEMS`

Attacking the Simple Transposition Cipher

- Brute force
  - Key sizes ~< 13 (13! = 6,227,020,800)

- English Analysis
  - Likely bigrams and trigrams
  - See more of this in Rail-Fence

Rail-Fence Cipher

- Rearrange letters in plaintext to produce ciphertext
  - Plaintext is `HELLO WORLD`
  - Rearrange as

    ```
    HLOOL
    ELWRD
    ```
  - Ciphertext is `HLOOL ELWRD`

How to decide which ciphertext is which algorithm?

- Caesar easy to test
- Index of Coincidence
- Correlation
- 1-gram, Bigram and n-gram frequencies
- Exploiting common English patterns
  - Q is always followed by a U
  - E most common letter…

Real World Examples

- Use XOR instead of shifts
  - Why?
- Everyone implements their own Crypto
  - Don't!
  - Side-Channel Attacks
  - Timing Attacks

Def Con Quals 2011

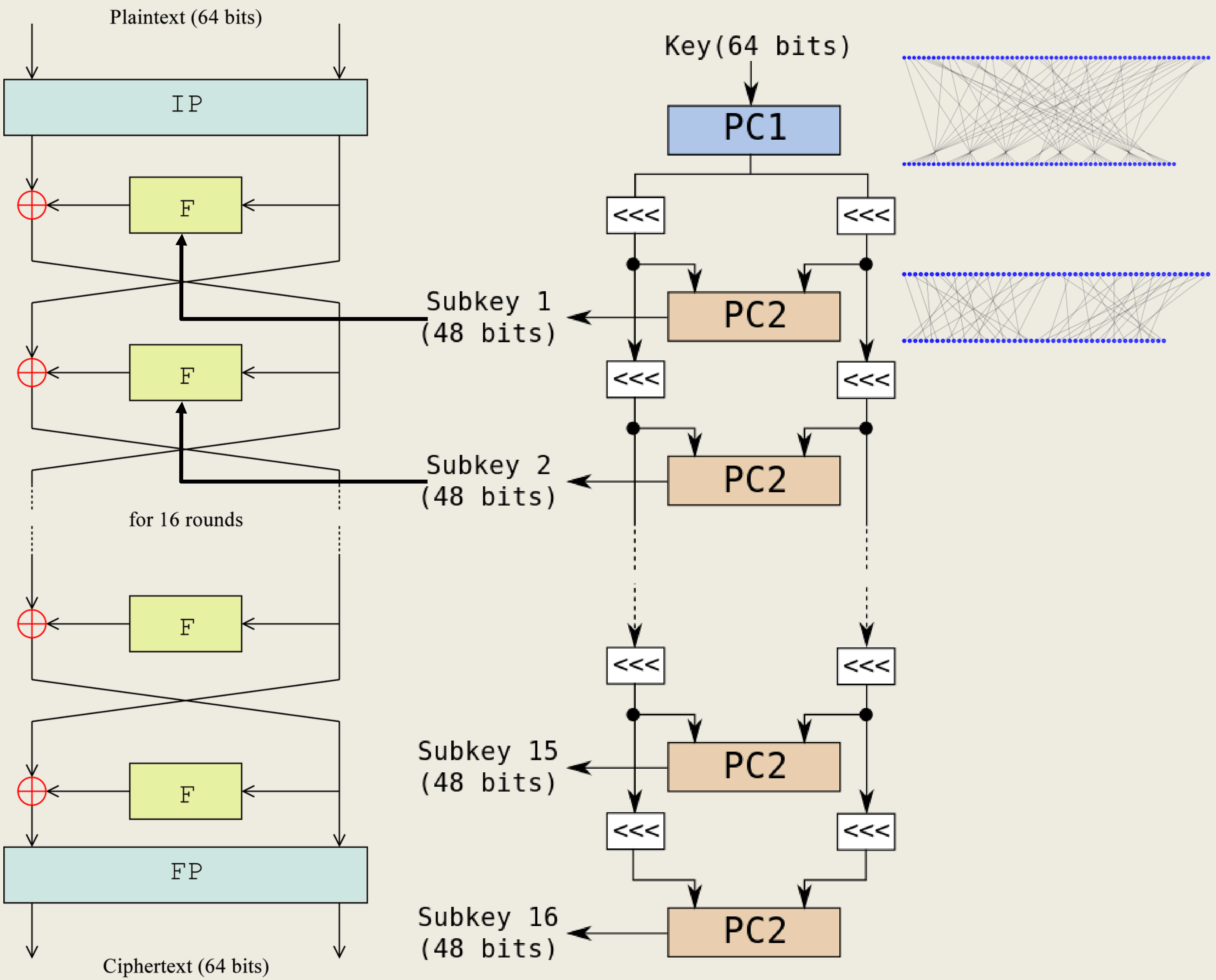
- Binary L33tness 300
- Tar archive with .dex file and .jpgs
- https://market.android.com/details?id=com.closecrowd.lokpixlite&hl=en
- Encryption was XOR 8-byte key
- Find out the key!

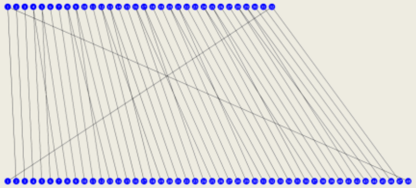
Modern Symmetric Encryption

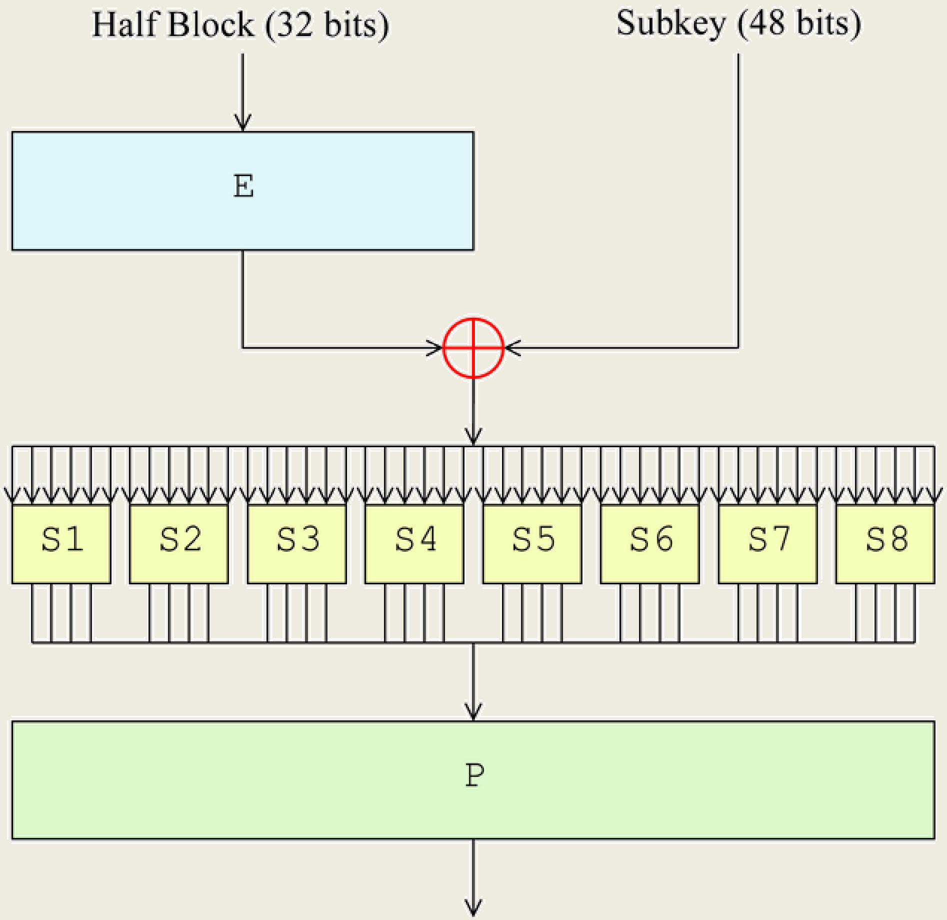
- Product Ciphers
  - Combination of substitution and transposition
- Complicated and long history
- Active area of development
- What properties do you want?

Data Encryption Standard (DES)

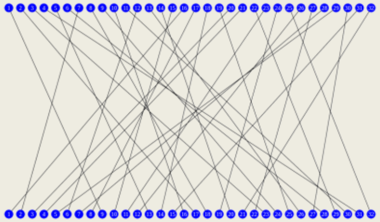
- Proposed by IBM as a standard for encrypting sensitive, unclassified government information
- Standardized in 1976/1977 (after tweaks from the submission after consultation with the NSA)
- 64 bit data block size
- 56 bit key


Plaintext (64 bits)
IP
F
F
for 16 rounds
F
F
FP
Ciphertext (64 bits)
Key(64 bits)
PC1
<<<
<<<
Subkey 1
(48 bits)
PC2
Subkey 2
(48 bits)
PC2
Subkey 15
(48 bits)
PC2
Subkey 16
(48 bits)
PC2





Half Block (32 bits)
Subkey (48 bits)
E
S1
S2
S3
S4
S5
S6
S7
S8
P



S_1	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0yyyy1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1yyyy0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
1yyyy1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
0yyyy1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
1yyyy0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
1yyyy1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
0yyyy1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
1yyyy0	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1yyyy1	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
0yyyy1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
1yyyy0	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
1yyyy1	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
0yyyy1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
1yyyy0	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
1yyyy1	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
0yyyy1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
1yyyy0	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
1yyyy1	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
0yyyy1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1yyyy0	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
1yyyy1	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

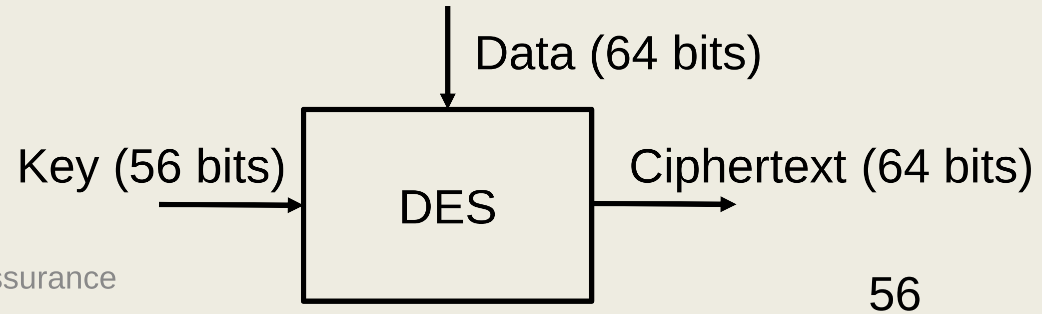
S_8	x0000x	x0001x	x0010x	x0011x	x0100x	x0101x	x0110x	x0111x	x1000x	x1001x	x1010x	x1011x	x1100x	x1101x	x1110x	x1111x
0yyyy0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
0yyyy1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1yyyy0	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
1yyyy1	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

The Fall of DES

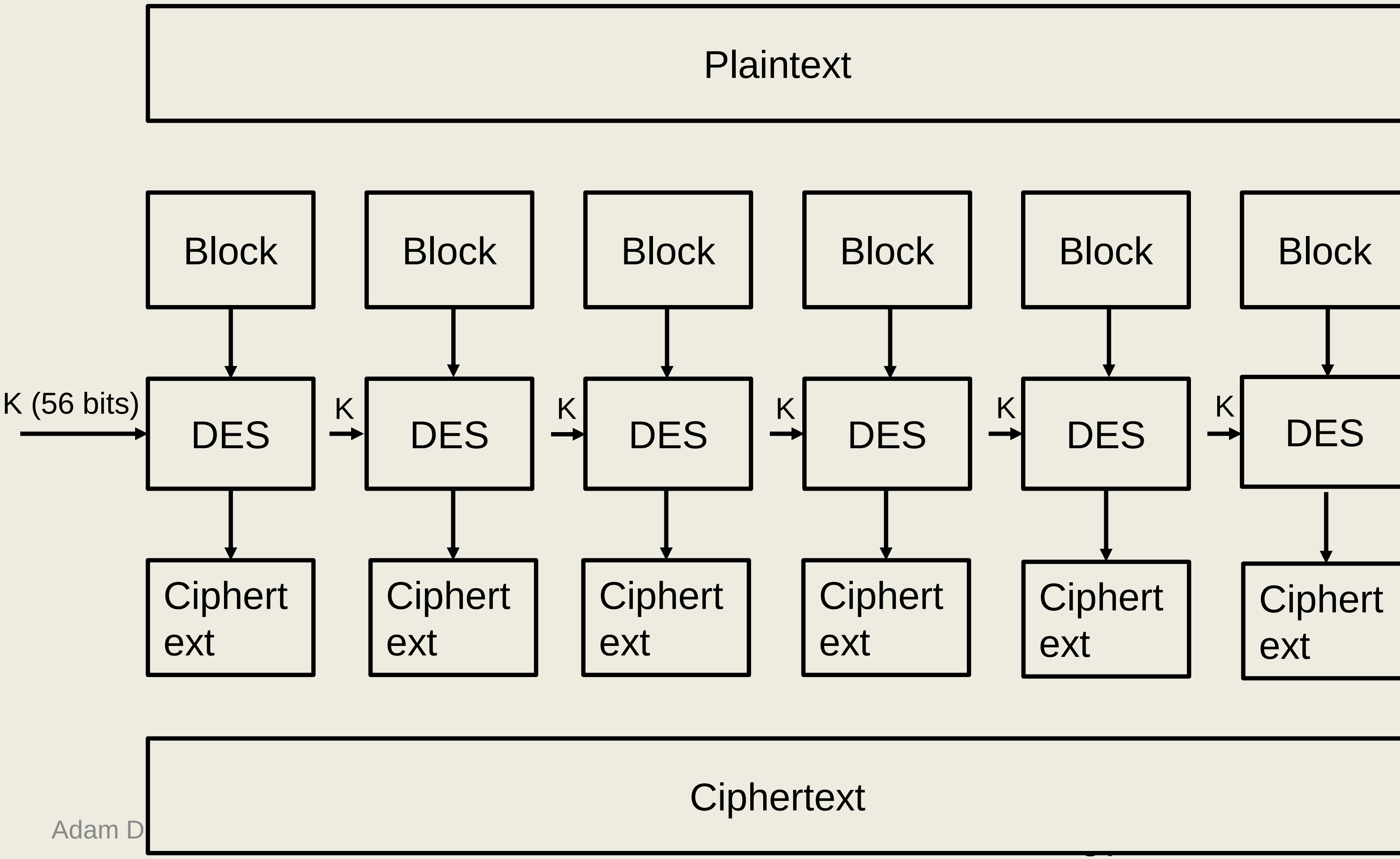
- Key size too small
  - 2^{56} or 72,057,594,037,927,936
  - 1998 the EFF built a custom DES-cracker for ~$250,000, broke key in 2 days
  - 2009 COPACOBANA machine built out of 120 FPGAs for ~$10,000 (off the shelf components)
- Differential cryptanalysis (discovered in late 1980s)
  - Prior version was vulnerable
- Linear cryptanalysis (1993)
- Withdrawn as a standard

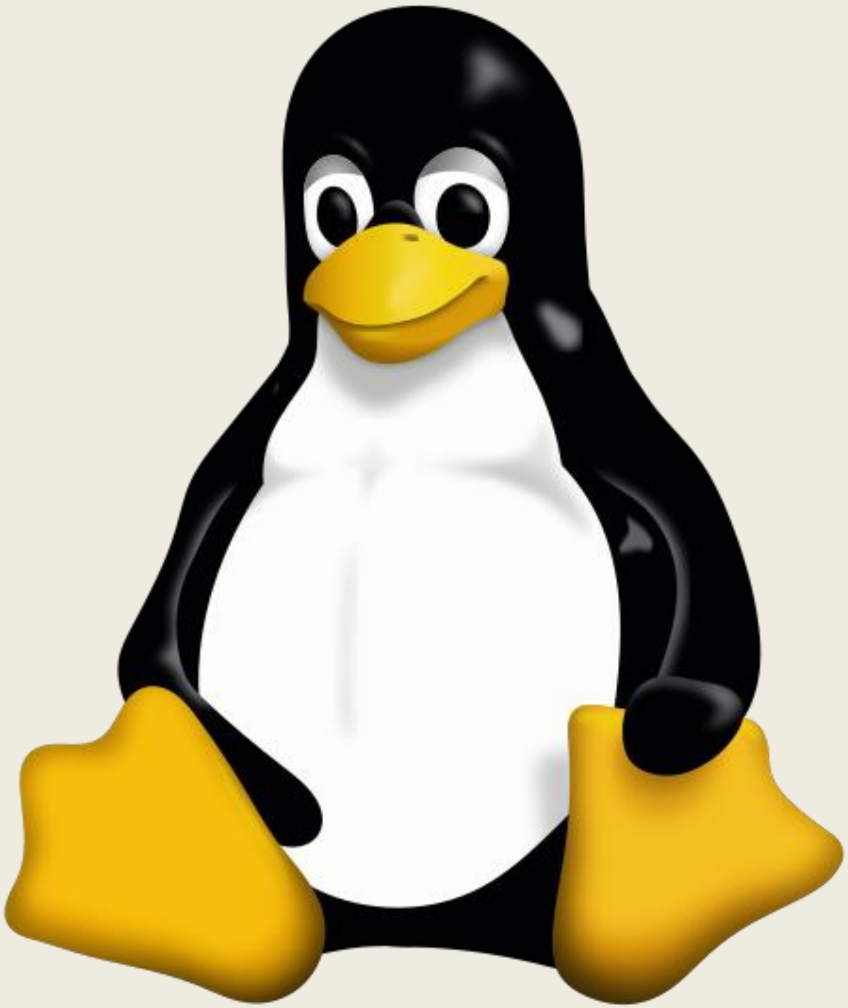
Symmetric Encryption in Practice

- Basic algorithm will only encrypt data of blocksize
  - What size of messages do we want to send?
- Different modes
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)

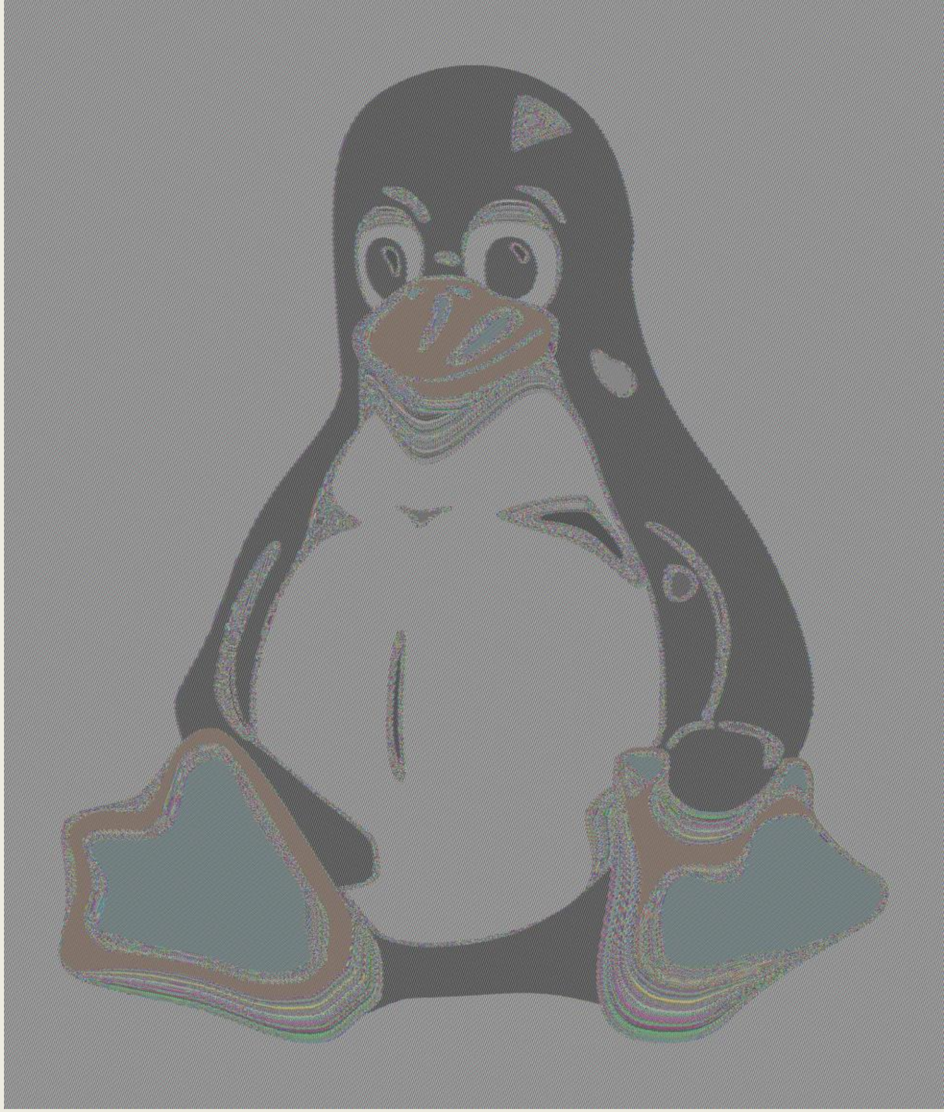


Electronic Code Book (ECB)





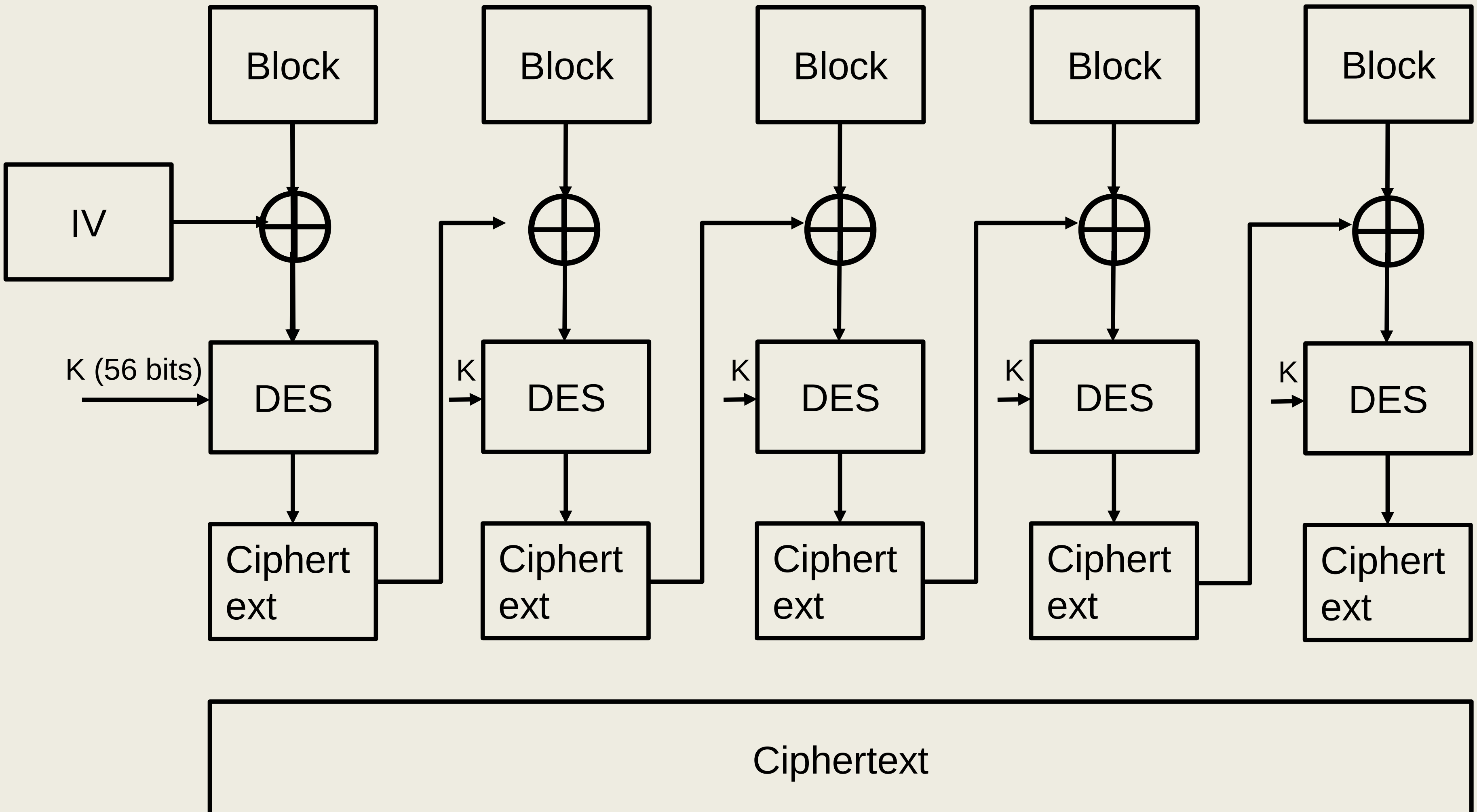
Original



ECB encrypted

image source from: https://blog.filippo.io/the-ecb-penguin/
https://commons.wikimedia.org/wiki/File:Tux.svg

Cipher Block Chaining (CBC)



Advanced Encryption Standard (AES)

- Originally called Rijndael
- Standardized in 2001
  - After five year process involving 15 competing designs
- 128 bit block size
- 128, 192, or 256 bit key size
- "The design and strength of all key lengths of the AES algorithm (i.e., 128, 192 and 256) are sufficient to protect classified information up to the SECRET level. TOP SECRET information will require use of either the 192 or 256 key lengths. The implementation of AES in products intended to protect national security systems and/or information must be reviewed and certified by NSA prior to their acquisition and use."
- Intel extended x86 to include this in hardware

One-time pad

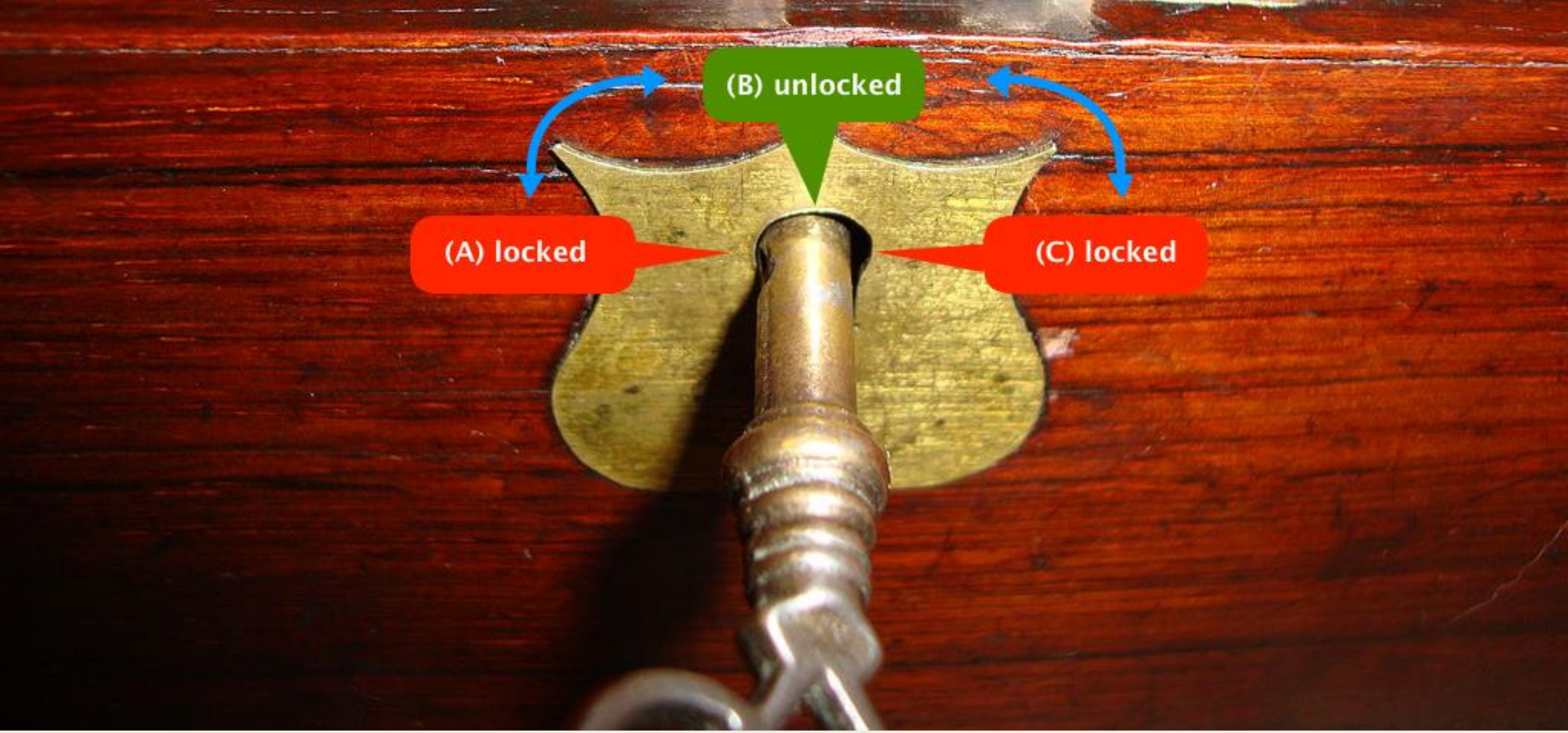
- Requires key to be the same size as the message being sent
- XOR key with message
- Never reuse key
- One-time pad is provably secure if…
  - Key is truly random
  - Key is as long as the plaintext
  - Key is never reused in whole or in part
  - Key is kept completely secret

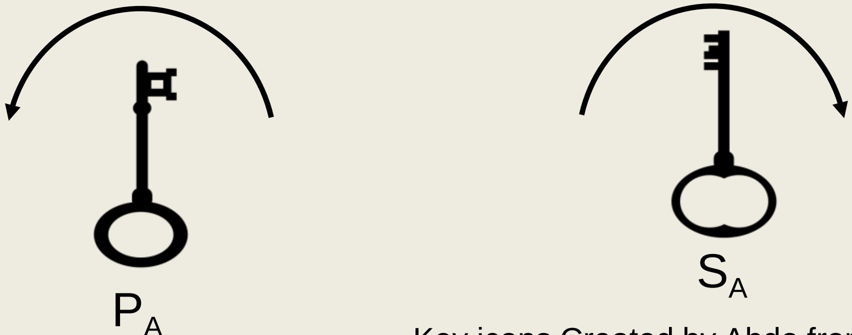
Main Drawbacks of Symmetric Cryptosystems

- Alice and Bob want to securely communicate
- How to securely transfer keys?

Asymmetric Cryptosystems

- Goal
  - How to encrypt information without requiring a secure, shared, secret key?
- Every party has two keys
  - Public Key (P)
    - P_A , P_B
  - Secret Key (S)
    - S_A, S_B
- Also called Public-key Cryptography





Idea and upper photo from
https://blog.vrypan.net/2013/08/28/public-key-cryptography-for-non-geeks/

Key icons Created by Abdo from Noun Project
https://thenounproject.com/abdulla_31/collection/keyes/?oq=key&cidx=1

Public-Key Properties

- Allows
  - Confidentiality
  - Nonrepudiation
- Requires
  - Easy to generate P and S, hard to generate S given P
  - Each party to key S private
- Both parties know P_A and P_B
  - Including the adversary, Eve (eavesdropper)
  - Everyone should know P_A and P_B

Encryption

- Alice wants to send message M to Bob
- Alice: $P_B(M)$ -> C
- Bob: $S_B(C)$ -> M
  - – What does Bob know for certain at this point?
- Eve: $P_A(C)$ -> Nothing, $P_B(C)$ -> Nothing
  - – What does Eve know at this point?

Nonrepudiation

- Alice wants to make a statement M that everyone knows is from Alice
- Alice: $S_A(M)$ -> C
- Bob: $P_A(C)$ -> M
  - What does Bob know for certain at this point?
- Eve: $P_A(C)$ -> M
  - What does Even know at this point?

Confidential and Nonrepudiation

- Alice wants to send a message M to Bob so that he knows it's from Alice
- Alice: $P_B(S_A(M))$ -> C
- Bob: $S_B(P_A(C))$ -> M
  - What does Bob know for certain at this point?
- Eve: $P_A(C)$ -> Nothing, $P_B(C)$ -> Nothing
  - What does Eve know at this point?

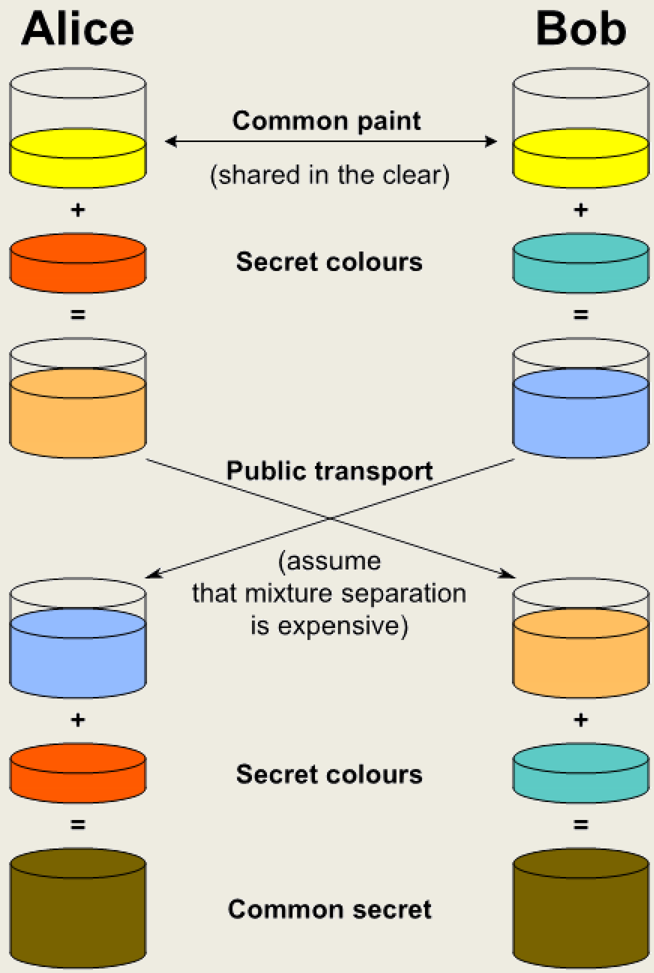
William Stanley Jevons, *The Principles of Science* (1874)

The same difficulty arises in many scientific processes. Given any two numbers, we may by a simple and infallible process obtain their product, but it is quite another matter when a large number is given to determine its factors. Can the reader say what two numbers multiplied together will produce the number 8,616,460,799? I think it unlikely that any one but myself will ever know; for they are two large prime numbers, and can only be rediscovered by trying in succession a long series of prime divisors until the right one be fallen upon. The work would probably occupy a good computer for many weeks, but it did not occupy me many minutes to multiply the two factors together. Similarly there is no direct process for discovering whether any number is a prime or not; it is only by exhaustingly trying all inferior numbers which could be divisors, that we can show there is none, and the labour of the process would be intolerable were it not performed systematically once for all in the process known as the Sieve of Eratosthenes, the results being registered in tables of prime numbers.

History of Public-Key Cryptography

- Public
  - 1976 Whitfield Diffie and Martin Hellman published a way to exchange keys
  - 1977 Ron **R**ivest, Adi **S**hamir, and Leonard **A**delman created RSA, a general public-key cryptosystem
- Classified
  - 1970 James Ellis, British Cryptographer at GCHQ conceived of “non-secret encryption”
  - 1973 Clifford Cocks (James’ colleague) implemented RSA

Diffie-Hellman Key Exchange



Alice and Bob agree to p = 23 and g = 9

Alice's $a_s = 4$, Bob's $b_s = 3$

Alice Sends Bob A
$A = 9^4 \bmod 23 = 6$

Bob sends Alice B
$B = 9^3 \bmod 23 = 16$

Alice computes s with a_s
$s = 16^4 \bmod 23 = 9$

Bob computes s with b_s
$s = 6^3 \bmod 23 = 9$

RSA Key Generation

1. Choose two distinct prime numbers p and q
2. Compute $n = p * q$
   - Given n, hard to factor p and q
   - For any a, n, and e, with $0 < a < n$ and $e > 1$, calculating $a^e \bmod n = c$ is easy
   - Given c, e, and n, hard to calculate a
3. Compute $m = (p – 1)(q – 1)$
4. Choose an e such that $1 < e < m$
5. Compute $d = e^{-1} \bmod m$
   - Can do this easily because $e * d = 1 \bmod m$
6. $P = (n, d)$
7. $S = (n, e)$

RSA Encryption

- Alice send a message M to Bob
  - Must have $P_B = (n_b, d_b)$
- Turn M into an integer m, $0 <= m < n_b$
- Alice: $m^d \bmod n \rightarrow c$
- Bob: $c^e \bmod n \rightarrow m$
- Eve: c, P_b and P_a

RSA Properties

- Allows us to send numbers less than n
- How to turn this into an actual cryptosystem?
  – Apply encryption to each letter?
  – Use RSA to transmit an AES key with AES encrypted data
    - $RSA_E(k)$, $AES_k(M)$

Message Integrity

- What if an attacker flips a bit
  - Or a bit is corrupted?
- How can the receiver know?

Cryptographic Hash Functions

- Function that maps arbitrary size data to a fixed size bit string
- One-way function
  - Easy to compute, hard to go back
  - Is it a 1-1 mapping?
- Deterministic
- Small change in input bit should completely change the output

Hash Functions Uses

- Public-Key Cryptography is fairly expensive
- Alice wants to make a statement M that everyone knows is from Alice
- Alice: $S_A(hash(M))$ -> Sig_M, M
- Bob: hash(M) =? $P_A(Sig_M)$
  – What does Bob know for certain at this point?
- What if Eve alters M to be M'?
  – Bob: hash(M') =? $P_A(Sig_M)$ -> hash(M') =? hash(M)

Hash Function Uses

- File or Message integrity
- Password verification
- Proof-of-work
- File or data identifier

Hash Function Properties

- Pre-image resistance
  - Given a hash value h, it should be difficult to find m, hash(m) = h
- Second pre-image resistance
  - Given input m_1 it should be difficult to find m_2 such that $hash(m_1) = hash(m_2)$
- Collision resistance
  - It should be difficult to find two messages m_1 and m_2 such that $hash(m_1) = hash(m_2)$

Public-Key Cryptosystem Weaknesses

- How to trust the public keys?
- Eve replaces all the public keys with their own
- Alice: $P_E(M_1) \text{ -> } C_1$
- Eve: $S_E(C_1) \text{ -> } M, P_B(M_2) \text{ -> } C_2$
- Bob: $S_B(C_2) \text{ -> } M_2$

How to trust public keys?

- Delegate/Centralization
  - Public-Key Infrastructure
- Decentralization
  - Web of trust

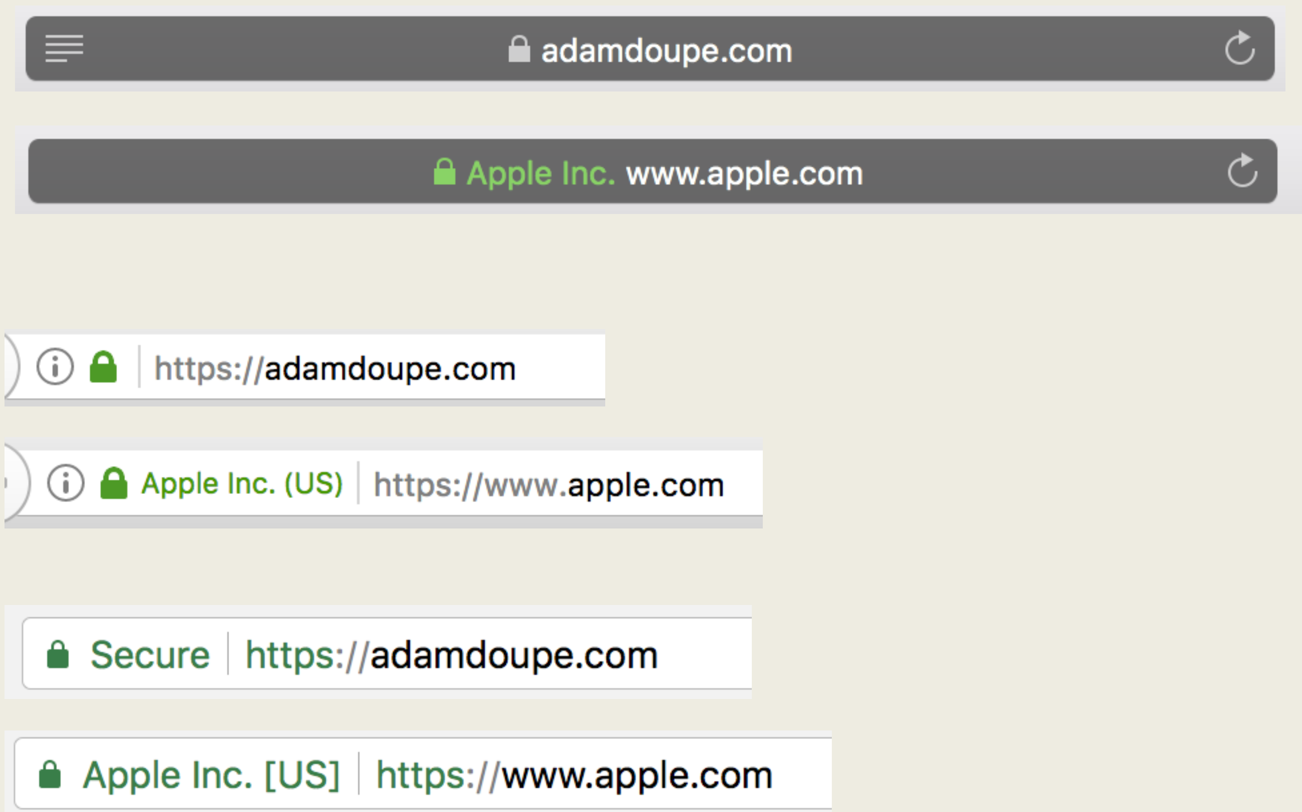
Public-Key Infrastructure (PKI)

- Certificate Authority
  - Responsible for verifying identify
  - Can delegate to other trusted Cas, creating a hierarchy
- Security goals
  - Issuing certificates
  - Revocation

The Modern Web

- HTTPS (which uses TLS/SSL) uses a PKI
- Root CAs
  - Must be distributed in OS and Software
- Different types of certificates (validation levels)
  - Domain Validated (DV) certificate
  - Organization Validation (OV) certificate
  - Extended Validation (EV) certificate

Visual Indicators of Status



Web of Trust

- Let end-users decide who to trust and to verify identity
- Propagate trust

Cryptography Research

- Breaking Crypto
  - Theory
  - Implementations
  - https://cryptopals.com/
- Securing Crypto
  - New Theory
  - New Implementations
- New types of crypto
  - Homomorphic encryption
  - Secure Multi-party computation
  - …
- Applied Crypto