# MP3: Cryptography

Due Chuenchujit

# Substitution Cipher

- Every character in an alphabet is replaced by a corresponding character from the key.
  - i.e. key = LDVRZTEKBNSXPWMYGCIHFQUJOA
  - A in message become L, B become D, … Z become A
- Decryption is done by a reverse-lookup of the key.
  - L in ciphertext become A, D become B, … A become Z

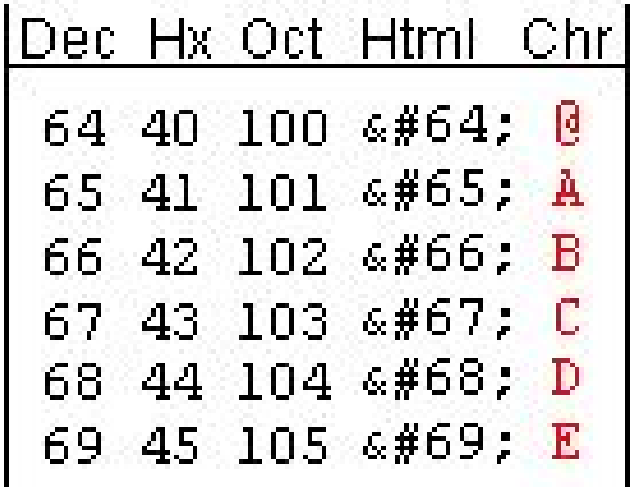
# Encoding and Decoding strings

- Encoding is when you want to represent a byte string in another format

```
>>> 'ABCD'.encode('hex')
'41424344'
>>> '\x41\x42\x43\x44'.encode('hex')
'41424344'
```

- Decoding is the opposite

```
>>> '41424344'.decode('hex')
'ABCD'
```



| Dec | Hx | Oct | Html | Chr |
|---|---|---|---|---|
| 64 | 40 | 100 | &#64; | @ |
| 65 | 41 | 101 | &#65; | A |
| 66 | 42 | 102 | &#66; | B |
| 67 | 43 | 103 | &#67; | C |
| 68 | 44 | 104 | &#68; | D |
| 69 | 45 | 105 | &#69; | E |

# Encoding and Decoding numbers

```
# string to integer
>>> x = ‘10’
>>> int(x)
10
>>> int(x, 16)
16
```

```
# integer to string
>>> y = 32
>>> hex(y)
‘0x20’
>>> bin(y)
‘0b100000’
```

# PyCrypto

- Installation: https://www.dlitz.net/software/pycrypto/
- Documentation: https://www.dlitz.net/software/pycrypto/api/current/
- Modules that will be useful for the mp
  - Crypto.Cipher.AES
    - ```
      >>> from Crypto.Cipher import AES
      >>> aes = AES.new(key, AES.MODE_CBC, iv)
      >>> aes.decrypt(ciphertext) or aes.encrypt(plaintext)
      ```
  - Crypto.Hash.SHA256
    - ```
      >>> from Crypto.Hash import SHA256
      >>> h = SHA256.new()
      >>> h.update('Hello World')
      >>> h.hexdigest()
      >>> 'a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e'
      ```

# AES - symmetric cipher

- Symmetric cipher - same key is used to encrypt and decrypt
- Block Cipher: operate on fixed-size chunks
- Key size: 128, 192, or 256 bits
- Different modes of operations

# (textbook-)RSA: a re-introduction

- Public key encryption
  - Key pair: public and private key
    - Public key: public knowledge
    - Private key: confidential
  - Messages encrypted with one key can only be decrypted by the other key
- Components of RSA
  - n - the modulus of the keys, created as a product of two large prime numbers
  - e - the public key
  - d - the private key
- Encryption with public key
  - $\text{encrypted\_text} = \text{plaintext}^{e} \bmod n$
- Decryption with private key
  - $\text{plaintext} = \text{encrypted\_text}^{d} \bmod n$

# Attacking RSA

Weiner's Attack

- ref: https://en.wikipedia.org/wiki/Wiener%27s_attack#Example
- Target key pairs with small private key
- Step:
  - Find continued fraction
    - http://www.maths.surrey.ac.uk/hosted-sites/R.Knott/Fibonacci/cfINTRO.html#section1
  - Compute convergences of continued fractions
  - Test the convergence values to find factorization of N
    - Let k/d be a convergence; let t = [(e*d) - 1] / k
    - If x^2 - (N - t + 1)x + N = 0 yield 2 real roots, we have found the factorization

# Introduction to Cryptographic Hashes

- Avalanche Effect
  - One bit flip in the input should cause roughly half the bits of the output to be flipped
- Why do we need them?
  - Quick verification of integrity
  - Otherwise the only way to verify integrity is to check every bits in the file!
- Why do people attack them?
  - Trick victim into thinking they have an authentic data
  - Two documents with the same hash (collision) with different content
- How do we attack cryptographic hashes?
  - Length-extension attack
  - Collision attack

# WHA - Weak Hashing Algorithm

```
WHA:
Input{inStr: a binary string of bytes}
Output{outHash: 32-bit hashcode for the inStr in a series of hex values}
Mask: 0x3FFFFFFF
outHash: 0
for byte in input
  intermediate_value = ((byte XOR 0xCC) Left Shift 24) OR
                       ((byte XOR 0x33) Left Shift 16) OR
                       ((byte XOR 0xAA) Left Shift 8) OR
                       (byte XOR 0x55)
  outHash =(outHash AND Mask) + (intermediate_value AND Mask)
return outHash
```

# Length Extension Attack

- MD5 and other cryptographic hash functions used Merkle–Damgård construction, which process the message in fixed-size blocks and uses the previous blocks as part of the process.  Paddings are appended to the input until it is a multiple of of a fixed number (e.g. 512 or 1024 bits).
- The function maintain an internal state after processing each block of the message.  The output of the function is the internal state after the last block of message (with padding) is processed.
- Weakness: attacker can append malicious information to a message and re-compute the hash value of the new message

# MD5 - length extension

```
m = "Use HMAC, not hashes"
x = "Good advice"

h = md5()
h.update(m)
print h.hexdigest()
>>> '3ecc68efa1871751ea9b0b1a5b25004d'

h = md5(state="3ecc68efa1871751ea9b0b1a5b25004d".decode("hex"), count=512)
h.update(x)

print h.hexdigest()

print md5(m+x).hexdigest()
```

What's missing?

# MD5 - length extension (2)

```
m = "Use HMAC, not hashes"
x = "Good advice"

h = md5()
h.update(m)
print h.hexdigest()
>>> '3ecc68efa1871751ea9b0b1a5b25004d'

h = md5(state="3ecc68efa1871751ea9b0b1a5b25004d".decode("hex"), count=512)
h.update(x)

print h.hexdigest()

print md5(m+padding(len(m)*8)+x).hexdigest()
```

# MD5 - Collision Attack

- Two messages with the same hash value
  - Attacker creates two different documents A and B, that have an identical hash value.
  - Attacker then shows document A to Alice, who agrees to what the document s
    hash and sends it back to Mallory.
  - Attacker copies the signature sent by Alice from document A to document B.
  - Attacker sends document B to Bob, claiming that Alice signed document B. Be
    digital signature matches the document hash, Bob's software is unable to dete
    modification.

# fastcoll

MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Allowed options:

| | | |
|---|---|---|
| -h | [ --help ] | Show options. |
| -q | [ --quiet ] | Be less verbose. |
| -i | [ --ihv ] arg | Use specified initial value. Default is MD5 initial value. |
| -p | [ --prefixfile ] arg | Calculate initial value using given prefixfile. Also copies data to output files. |
| -o | [ --out ] arg | Set output filenames. This must be the last option and exactly 2 filenames must be specified. |

# PostScript

- Make sure your ps file can be viewed with GhostScript in Linux or Preview in OSX
- HelloWorld: http://paulbourke.net/dataformats/postscript/3.ps
- Cheat sheet: http://www.math.ubc.ca/~cass/courses/ps.html
- Postscript stack
  - Last-In-First-Out data structure
  - Initialize a stack with one value: ( value )
  - dup: duplicate the top element of the stack and return the value
- Conditional
  - A B eq { command_if_true } if

# Generating more than 2 collisions

1. Generate a pair of collision
2. Apply Length extension
3. Repeat
4. ...
5. Profit