* Course Content
  + facwiki.cs.byu.edu/cs465
* Homework
  + Regularly assigned, due at th start of class each Tuesday
* Pro Projects
  + Due Frieday at 5PM
* Exams
  + 2 + final
* Cryptography Intro
  + Encryption
    - Hide the true meaning of something
    - Algorithms user transposition and substitution in complex ways
    - Bad Examples:
      * ROT-13
      * Vigenere Cipher
    - Strong Examples:
      * AES
      * 3DES
      * RC4
* Kerckhoff’s: everything depends on the secrecy of the KEY
* --------------------------------------------
* \*Describe Symmetric Encryption
  + draw picture
  + write a few sentences
* -Cryptographic Hash Functions
* Digest
  + Is usually a fixed size
  + The output of a cryptographic hash function
* \*Explain MAC
  + authenticity and integrity
* Cryptography
  + Encryption
  + Digital signatures
* Xtime() == times by HEX, and mOD by ‘11b’
* Access control
  + Authentication
    - You are who you claim to be
  + Authorization
    - Entities have permission to perform an action
* Confidentiality (Encryption)
  + Prevent unauthorized people from seeing something
* Integrity (No unauthorized person changing this)
  + Don’t modify this!
* Authenticity (Who someone is, ID who is invovled)
  + Is this from who it claimed to be from
* Non-repudiation
  + Can prevent someone from denying they took part in a transaction
  + Usually involves cryptographic evidence that will stand up in court
* Two requirements for strong symmetric encryption
* 1. strong algorithm (cipher)
  + attacker is unable to decrypt ciphertexxt or discover the key even if attacker has samples of ciphertext/plaintext created using the secret key
* 2. sender and receiver must securely obtain and store the secret key
* One-Time Pad
  + Works good
  + Just a little unfeasible for large data
  + Hard to get unique key everytime
* ECB
  + Weakness – it can reveal patterns in the plain text.
* Stream Modes – we don’t have to do padding.
* Padding
  + Relevate to ECB and CBC
  + Schemes
    - \*\*Read Padding article
    - Pad with the value of the padding needed
    - 0x80 followed by 0x00
    - Pad with 0x00
      * Last byte is equal to the number of padding bytes
    - Pad with 0x00 or spaces
  + FIX some of the edge cases: just always pad!
* \*\* Properties of a Hash Function
  + H can be applied to a block of data of any size
  + H produces a fixed-length output
  + H(x) si relatively easy to compute for any given x
  + For any given value h, it is computationally infeasible to find x such that H(x) = h
  + For any given block x, it is computationatonally infeasible to find y != x with h(y) = h(x) (weak collision resistance)
  + It is computationally infeasible to find ANY pair (x,Y) such that H(x) = H(y) (strong collision resistance) x != y
* CBC – Is the most common mode of systems
* \* How does AES provide integrity?
  + No not really
  + It is ment to provide confidentiality
* AES is just a simple small scope algorithm. Modes and blocks extra are all on the outside of AES.
* Know the Hash properties.
  + Talk about he attacks and their costs
* Hash is good for
  + Digital signatures
  + File downloads
  + ?
* Feistel Cipher Structure
  + Symmetric encryption algorithms
    - DES, IDEA, Blowfish, …
    - NOT AES
  + Applies to block cipher
* MAC – provide authentication and integrity
  + Sometimes called a digial signature
  + CBC-Mac
    - Use cbc mode and a block cipher
    - Expensive
  + Hash it and the Encrypt the digest
  + Hash the message along with a shared key
* HMAC – mac generated using hashing
  + Vulnerable to a message extension attack
* \* H(K||H(K||M))
  + Just prepend the key twice.
* HMAC
  + General term means any MAC that is implemented using hashing
  + Other times it’s the Government standard
* **Asymmetric Encryption**
* Why public key crypto is cool
  + Is a linear solution to the key distribution problem
    - Symmetric is expoinential
  + Send messages to people you don’t share a secret key with
    - So only they can read it
    - They know it came from you
* Number Theory
* Diffie – Hellman
  + Protects a gains a passive attacker
* Cost of Modular Exponentiation:
  + Cost of the number of one bits
  + Cost of updating the base every time
* RSA
  + Co-prime == relatively prime
* \*\*Describe public key cryptography
* \*\*Describe digital signatures
* RSA
  + P = Large Prime Number
  + Q = Large Prime Number
  + E = public exponent
    - Co-prime, Relatively prime to phi(n)
    - Common values 3,65537
  + D = private exponent
    - It’s the mulciplicative inverse of E (mod phi(n))
  + N = modulus
    - N = P\*Q
  + C=m^e%n
  + M = c^d%n
* \*computing phi(n) in RSA
  + == (p-1)(q-1)
* \*Memorize Steps for RSA Encryption
* Why is RSA Secure?
* MI
  + Extended Euclidean algorithm