CMSC 426
Principles of Computer Security

Lecture 11

Introduction to Cryptography (continued)

Last Class We Covered

- Introduction to crypto
  - Definitions
  - Ciphers
- Block ciphers
  - DES
  - 3DES
  - AES

- Confusion and diffusion
- Parallelization

Any Questions from Last Time?

Today's Topics

- Block cypher modes

- Asymmetric encryption
  - Diffie-Hellman
  - RSA
  - Math (for real this time)

Modes of Operation

Modes of Operation

- Block ciphers themselves are only good for encrypting a block
  - Repeatedly applying a block cipher to larger amounts of data requires a mode of operation
  - Some modes require an Initialization Vector (IV) to get started

- Different modes of operation exist for different purposes
  - Efficiency
  - Parallel encrypt and/or decrypt
  - Encrypting a stream

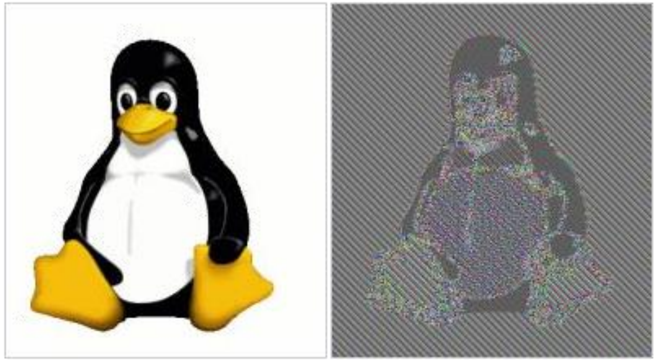
Notation

- $E_K(P)$
  - Encryption of plaintext *P* with key *K* using an arbitrary block cipher
- $D_K(C)$
  - Decryption of cipher *C* with key *K* using an arbitrary block cipher
- *Arbitrary block cipher*
  - For example, DES, 3DES, or AES

Electronic Codebook Mode (ECB)

- Simplest and most naïve mode of operation
  - Simply encrypts/decrypts each block with the same key

- Pros:
  - En/decryption can be performed in parallel
- Cons:
  - Requires padding of plaintext
  - Low diffusion



Original image | Encrypted using ECB mode

$$C_i = E_K(P_i)$$

$$P_i = D_K(C_i)$$

Image taken from https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

Quick Note: Padding

- Padding involves adding garbage/filler to the end of the plaintext so that it perfectly fits within a block size

- Downside is <u>not</u> the space “wasted” on the extra text
- Rather, padding can allow an adversary to examine and learn things about the plaintext by examining the padded ciphertext
  - Not something we’ll go into in depth in class
  - Read about “padding oracle attacks” for more information

Cipher Block Chaining Mode (CBC)

- Each block of plaintext is **XOR**ed with the previous ciphertext block before being encrypted
  - Uses an initialization vector for the first plaintext block

- Pros:
  - Much better diffusion
- Cons:
  - Requires padding
  - Can't parallelize encryption
    - But can parallelize decryption – why?

$$C_i = E_K(P_i \oplus C_{i-1})$$

$$P_i = D_K(C_i) \oplus C_{i-1}$$

Cipher Feedback Mode (CFB)

- Each block of plaintext is **`XOR`**ed with the previous ciphertext block <u>after</u> the previous ciphertext is re-encrypted
  - Plaintext never directly “touches” the encryption algorithm
  - Uses an initialization vector for the first plaintext block

- Block cipher is now a “stream cipher”
  - Uses the block cipher as a “key generator”
  - Digits can be encrypted one at a time, which means no padding is necessary
  - Encryption cannot be parallelized

$$C_i = E_K(C_{i-1}) \oplus P_i$$

$$P_i = E_K(C_{i-1}) \oplus C_i$$

Counter Mode (CTR)

- Also works as a stream cipher
- Requires a pseudo-random seed, *S*, to function
  - For each successive en/decrypt, the seed “counts” up by one

- Pros:
  - Encryption can be parallelized, as seed simply counts up
  - Decryption can be parallelized as well
  - Plaintext does <u>not</u> need to be padded
- Cons:
  - ???

$$C_i = E_K(S + i - 1) \oplus P_i$$

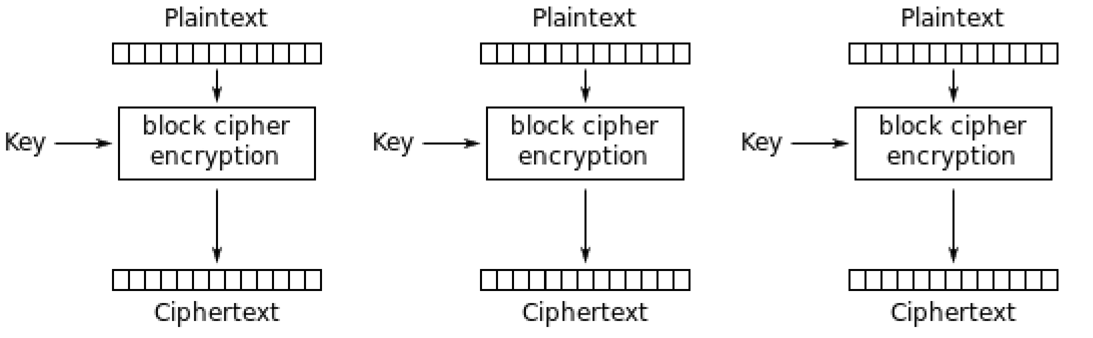
$$P_i = E_K(S + i - 1) \oplus C_i$$

Comparison of Modes of Operation

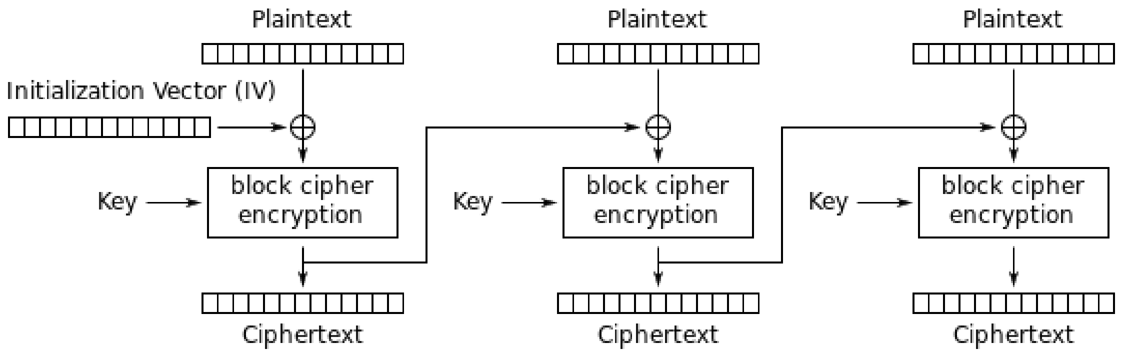
	Parallel Encrypt	Parallel Decrypt	Padding Required	Stream Cipher	Initialization Vector	Repeats in Cipher[1]
ECB	✓	✓	✓			✓
CBC		✓	✓		✓	
CFB		✓		✓	✓	
CTR	✓	✓		✓		

[1] Encrypting structured or repeating plaintext results in repeating cipher blocks

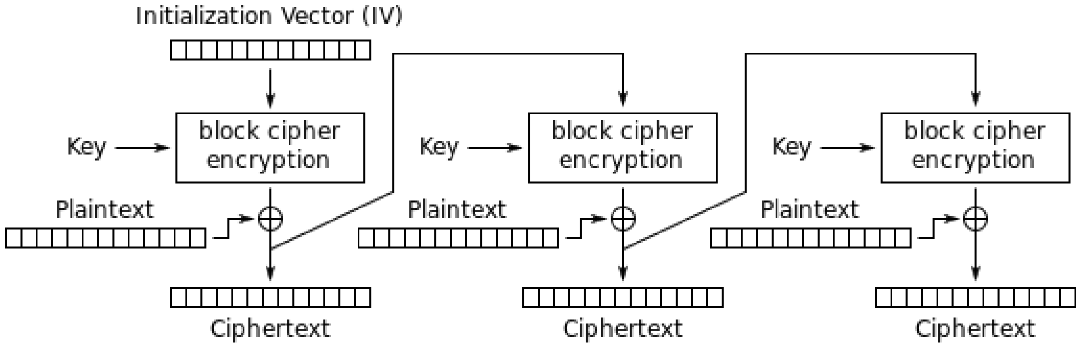
Enc. Algorithms of Modes of Operation



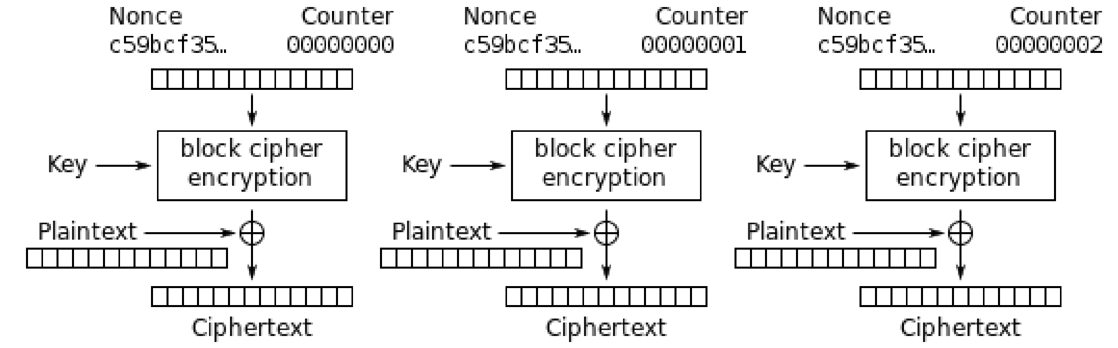
Electronic Codebook (ECB) mode encryption



Cipher Block Chaining (CBC) mode encryption



Cipher Feedback (CFB) mode encryption



Counter (CTR) mode encryption

Images taken from https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

Diffie-Hellman

Shortcomings of Symmetric Encryption

- Symmetric key must remain secret to be secure

- But how do you communicate what the secret key is?
  - Without already having a secret key?
  - ???
  - You can't!

- Need some way to share keys over an unsecured channel

Diffie-Hellman Key Exchange

- Named after Whitfield Diffie and Martin Hellman

- It is a way for two parties to
  - Use insecure communication to
  - Agree on a cryptographic key
  - Without anyone else being able to figure out what it is
- Neither party “chooses” the key, but that doesn’t matter
  - They just need the same one
- How to achieve this?
  - Math!

Basic Diffie-Hellman Algorithm

- Choose two non-secret values p and g
  - p is prime
  - g is generator, a primitive root modulo p (don't worry about this right now!)

- Each party
  - Chooses an integer Y in the range 1 to p - 1 (inclusive)
  - Calculates $y = g^Y \% p$ and transmit y across the clear channel
  - Use the other party's transmitted integer (x) to calculate $K = x^Y \% p$
- Both parties now have the same value K, for a symmetric key

Example Diffie-Hellman Algorithm

- Alice and Bob agree to use p = 37 and g = 11
  - Normally they would use large numbers, but this is an example

- Alice chooses the integer A = 2, Bob chooses B = 9
  - $a = g^A \% p$ $a = 11^2 \% 37$ $a = 10$
  - $b = g^B \% p$ $b = 11^9 \% 37$ $b = 36$
  - Over the clear channel, Alice transmits 10 and Bob transmits 36
- Each now calculates the key K
  - Alice: $K = b^A \% p$ $K = 36^2 \% 37$ $K = 1$
  - Bob: $K = a^B \% p$ $K = 10^9 \% 37$ $K = 1$

Diffie-Hellman: The Math

- Alice calculates $a = g^A \% p$
- Bob calculates $b = g^B \% p$
  - They transmit these values of a and b to each other, then…

- Alice calculates $K = b^A \% p$ same thing as $(g^B \% p)^A \% p$
- Bob calculates $K = a^B \% p$ same thing as $(g^A \% p)^B \% p$
  - Both of which simplify to $g^{AB} \% p$
    - (Because ~*~math~*~)

Diffie-Hellman Security

- Only *p*, *g*, *a*, and *b* are transmitted in the clear
  - So any attacker could have those

- But to calculate *K*, they also need either *A* or *B*
  - Which they could solve for with the formula $\log_g B \,\%\, p$
  - But this is really hard to do when *p* is 600 digits long
    - (For now – if this changes, we're all in deep trouble.)
- Private keys (*A* and *B*) should also be large numbers
  - Makes them difficult to calculate for an attacker, or even for the other legitimate person in the communication

RSA (not a real acronym)

RSA Overview

- RSA stands for Rivest, Shamir, and Adleman, its inventors
  - Is not necessarily a method for key exchange
- Is a form of asymmetric encryption
  - Uses two separate keys: public and private
- Public key is available to anyone and everyone
- Private key must be kept secret

RSA Key Generation Algorithm

- Pick two secret prime numbers, *P* and *Q*
  - With those values, calculate *n* = *P* * *Q*
- Choose a valid public exponent *e*
  - Software today uses 65537 (0x10001) to make calculations faster
    - A valid *e* is not a factor of *n*, and must be less than (*P*-1)*(*Q*-1) (~*~math~*~)
- Calculate a private exponent *D*
  - Such that *e* is congruent to *D* % (*P* -1) * (*Q* -1) (more ~*~math~*~)

- Public key components are *n* and *e*
- Private key components are *n* and *D* (normally save *P* and *Q* too)

Using RSA Keys

- Encryption
  - The plaintext P is converted into an integer M
    - (Don't worry about this for now)
  - $c = M^e \% n$ (remember, e and n were our public key components)
- Decryption
  - $M = c^D \% n$ (remember, D and n were our private key components)
- Mathematical proof
  - Outside of the scope of this class (number theory, etc.)
  - Read the paper if you're really interested

RSA Example: Key Generation

- Key generation:
  - Choose $P = 43$ and $Q = 59$
  - Calculate $n = P * Q$ $\quad n = 43 * 59 \quad n = 2537$
  - Choose $e = 67$
  - Calculate $D = 1927$

- Public key: $n = 2537$, $e = 67$
- Private key: $n = 2537$, $D = 1927$

RSA Example: Encryption/Decryption

- Now, someone wants to send you a message M = 42
  - To encrypt it, they use your public key: n = 2537, e = 67
  - $c = M^e \% n$ $c = 42^{67} \% 2537$ $c = 1332$
  - This ciphertext of 1332 is sent over a clear channel

- After receiving the message 1332, you want to read it
  - To decrypt, you'll use your private key: n = 2537, D = 1927
  - $M = c^D \% n$ $M = 1332^{1927} \% 2537$ $M = 42$

RSA Security

- An attacker has access to only *n* and *e*
  - They need access to *D* to have a complete private key
  - If they could factor *P* and *Q* out of *n*, they could calculate *D*

- Fortunately, calculating the large primes that are the only factors for a large number is ***hard***
  - The larger the primes, the harder it is to factor

- Fun fact: the largest known prime is $2^{77,232,917} - 1$
  - It has 23,249,425 digits

RSA: Digital Signatures

- Encryption and decryption are inverses of each other
- If something is <u>en</u>crypted with the private key, it can be <u>de</u>crypted with the public key
  - What does this allow us to do?
  - State “only this person could have <u>en</u>crypted this”
- This is part of something called a ***digital signature***, and is meant to prove the message came from a specific individual
  - Digital signatures are more complex than just this; we’ll discuss the details next time

(Pseudo)-Random Number Generation

- **`rand()`** is <u>not</u> an acceptable (pseudo) random number generator for anything that has an actual purpose

- If you want something statistically viable, you need to use an actually good pseudorandom number generator (PRNG)

- If you're going to use the numbers for security-related purposes, use a **cryptographically secure pseudorandom number generator (CSPRNG)**
  - If you don't know if it's a CSPRNG, it probably isn't

Quantum Computing

- ***If*** a sufficiently large quantum computer is ever built:

- RSA and Diffie-Hellman are completely broken by an algorithm called Shor's algorithm

- The bit length of symmetric ciphers is effectively halved
  - If it would previously require 2^{128} computations to crack something, it would only require 2^{64} quantum computations

Announcements

- Lab 2 is due Thursday night
- Paper 1 will be coming out soon
- Exams are graded and available for pickup