Coursera Cryptography I (Stanford)

Introduction

Cryptography is Everywhere

- Secure Communication
  - Web Traffic : HTTPS
  - Wireless Traffic : 802.11i WPA2, WEP, GSM, Bluetooth
- Encrypting Files on Disk
  - EFS
  - TrueCrypt
- Content Protection (e.g. DVD, Blu-ray)
  - CSS
  - AACS
- User Authentication

Secure Communication

- No tampering.
- No eavesdropping.

Secure Socket Layer / TLS

- Handshake Protocol : Establish shared secret key using public-key cryptography
- Record Layer : Transmit data using shared secret key
  - Ensure confidentiality and integrity

- Encryption algorithm is publicly known.

- Single Use Key (One Time Key) :
  - Key is only used to encrypt one message.
    - Encrypted Email : New key generated for every email
- Multi Use Key (Many Time Key) :
  - Key used to encrypt multiple messages.
    - Encrypted Files : Same key used to encrypt many files
    - Need more machinery than for one-time key

Applications

- Digital Signatures
- Anonymous Communication
- Anynomous Digital Cash
    - Spend a digital coin without anyone knows who I am
    - Prevent double spending
- Protocols
    - Elections
    - Private Auctions
    - Secure Multi-party Computation
- Privately Outsourcing Computation
- Zero Knowledge(Proof of Knowledge)

Theorem

- Anything that can be done with trusted authority can also be done without it.

A Rigorous Science

- Precisely specify threat model.
- Propose a construction
- Prove that breaking construction under threat mode will solve an underlying hard problem

Examples (Most Badly Broken)

- Substitution Cipher
- Caesar Cipher
- Vigener Cipher
    - + mod 26
    - Suppose most common = “H”, first letter of key = “H” − “E” = “C
- Rotor Machines
    - the Hebern Machine(Single Rotor)
    - Enigma(3-5 rotors)
        - keys = 26^4 = 2^{18}
        - Actually 2^{36} due to plugboard
- How to break them?
    - By uneven frequency of letters or letter pairs appear in English texts.
- Data Encryption Standard

- Keys = 2^{56}
- Block Size = 64 bits
- AES
- Salsa20

Discrete Probability

- U : Finite Set
  - Def : Probability distribution P over U is a function $P : U \rightarrow [0, 1]$ such that $\sum_{x \in U} P(x) = 1$
- Uniform Distribution
  - $\forall x \in U, P(x) = \frac{1}{|U|}$
- Point Distribution at x_0
  - $P(x_0) = 1$
  - $\forall x \neq x_0, P(x) = 0$
- Events
  - For a set $A \subseteq U$: $Pr[A] = \sum_{x \in A} P(x) \in [0, 1]$
    - $Pr[U] = 1$
  - The set A is called an event.
- The Union Bound
  - For events A_1 and A_2, $Pr[A_1 \cup A_2] \leq Pr[A_1] + Pr[A_2]$
  - Equals if $A_1 \cap A_2 = \emptyset$
- Random Variables
  - Def : A random variable X is a function $X : U \rightarrow V$
  - More generally, random variable X introduces a distribution on V: $Pr[X = v] := Pr[X^{-1}(v)]$
- The Uniform Random Variable
  - Let U be some set, e.g. $U = 0, 1^n$
  - We write $r \xleftarrow{R} U$ to denote a uniform distribution variable over U for all a $\in U$.

- $Pr[r = a] = \frac{1}{|U|}$
    - Formally, r is the identity function: $r(x) = x$ for all $x \in U$
- Randomized Algorithms
    - $y \leftarrow A(m, r)$ where $r \xleftarrow{R} \{0, 1\}^n$
    - Output is a random variable $y \xleftarrow{R} A(m)$
- Independence
    - Events A and B are independent if $Pr[A \text{ and } B] = Pr[A] \cdot Pr[B]$
    - The defintion of random variables is similar.
- XOR
    - Bitwise addition mod 2

Theorem

- Y is a random variable over $\{0, 1\}^n$, X is a independent uniform variable on $\{0, 1\}^n$.

  Then $Z := Y \oplus X$ is uniform variable on $0, 1^n$

- The Birthday Paradox
    - Let $r_1, \ldots, r_n \in U$ be independent identically distributed random variables

Theorem

- When $n = 1.2 \times |U|^{\frac{1}{2}}$, then $Pr[\exists i \neq j : r_i = r_j] \geq \frac{1}{2}$.

Stream Cipher

Symmetric Ciphers

- Definition : A cipher defined over (K, M, C) is a pair of efficient algorithms (E, D)

  where $E : K \times M \rightarrow C$, $D : K \times C \rightarrow M$ such that $\forall m \in M, k \in K$:

  $D(k, E(k, m)) = m$

    - E is often randomized.
    - D is always deterministic.

The One Time Pad

- First example of a secure cipher.

- $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$
- Key is a random bit string as long as the message.
- $E(k,m) = k \oplus m$
- $D(k,c) = k \oplus c$

- Very fast encryption and decryption, but long keys.

What is a Secure Cipher?

- Attacker's ability : CT only attack
- Shannon : Cipher text should reveal no information about plaintext

Information Theoretic Security

- Def : A cipher (E,D) over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ has perfect secrecy if

  $\forall m_0, m_1 \in \mathcal{M}, len(m_0) = len(m_1)$ and $\forall c \in \mathcal{C}$,

  $Pr[E(k,m_0) = c] = Pr[E(k,m_1) = c]$ where k is uniform in $\mathcal{K}$.

  - Given the ciphertext only, we cannot tell the message is m_0 or m_1.

Lemma

- One Time Pad has perfect secrecy.
  - $\forall m, c : \text{if } E(k,m) = c \Rightarrow k \oplus m = c \Rightarrow k = m \oplus c \Rightarrow \#\{k \in \mathcal{K} : E(k,m) = c\} = 1 \Rightarrow$

    OTP has perfect secrecy.$_\#$

Theorem

- Perfect Secrecy $\Rightarrow |\mathcal{K}| \geq |\mathcal{M}|$
  - One time pad is hard to use in practice.

Psuedorandom Generators

- Stream Ciphers : Making OTP Practical
  - Place random key by pseudorandom key
  - PRG is a function $G : \{0,1\}^s (\text{seed space}) \rightarrow \{0,1\}^n, n >> s$
    - Efficiently computable by a deterministic algorithm
  - Stream Ciphers cannot have perfect secrecy because the key is shorter than the message.
- PRG must be unpredictable.

- Definition : PRG is unpredictable if it is not predictable. $\Rightarrow$ $\forall i$: no efficient adversary can predict bit (i+1) for non-neg ϵ
- Linear congruent generators are weak PRGs

Negligible v.s Non-negligible

- In practice : ϵ is a scalar.
  - Negligible : $\epsilon \leq \frac{1}{2^{80}}$
  - Non-negligible : $\epsilon \geq \frac{1}{2^{30}}$
- In theory : ϵ is a function, $\epsilon : Z^{\geq 0} \to R^{\geq 0}$
  - Negligible : $\forall d, \lambda \geq \lambda_d : \epsilon(\lambda) \leq \frac{1}{\lambda^d}$
  - Non-negligible : $\exists d : \epsilon(\lambda) \geq \frac{1}{\lambda^d}$
- PRGs : The Rigorous Theory View
  - PRGs are parameterized by a security parameter λ.
  - If λ increases, PRG becomes more secure, seed lengths and output lengths grows with λ.
  - For every λ, there is a different PRG $G_\lambda : K_\lambda \to \{0,1\}^{n(\lambda)}$
  - If a PRG is predictable at position i if there exists a polynomial time algorithm A such that $Pr_{k \leftarrow K_\lambda}[A(\lambda, G_\lambda(k)|_{1,\dots,i}) = G_\lambda(k)_{i+1}] > \frac{1}{2} + \epsilon(\lambda)$ for some non-negligible $\epsilon(\lambda)$.

Attacks on OTP and Stream Ciphers

- Attack 1 for two-time pad
  - $C_1 \leftarrow m_1 \oplus PRG(k)$

    $C_2 \leftarrow m_2 \oplus PRG(k)$
  - Eavesdropper does $C_1 \oplus C_2 \to m_1 \oplus m_2$
  - Then it can use the redundancy in English to figure out the messages.
  - Examples : Project Venona, MS-PPTP(Windows NT), 802.11b WEP, Disk Encryption
  - 802.11b WEP
    - Length of initalization vector(IV) : 24bits
    - Repeated IV after 2^{24} frames

        - One some cards, IV resets to 0 after power cycle.
    - A better construction
        - Different keys for every frame, user a stronger encryption method
    - Summary
        - Never use a key more than once.
        - Network traffic : negotiate a new key for every session(TLS)
        - Disk encryption : typically do not use a stream cipher
- Attack 2 : No Integrity(OTP is malleable)
    - Modification to ciphertexts are undetected and have predictable impact on plaintext

Real World Stream Ciphers

- Old Example : RC4
    - Used in HTTPS and WEP
    - Weaknesses
        - Bias in inital output $Pr[2^{nd}\ \text{byte} = 0] = \frac{2}{256}$
        - Probability of $(0, 0)$ is $\frac{1}{256^2} + \frac{1}{256^3}$
        - Related Key attacks
- Old Example : CSS
    - Linear Feedback Shift Register(LFSR)
    - DVD, GSM, Bluetooth are all broken
    - Seed = 5 bytes = 40bits
    - Easy to break in time 2^{17}
    - For all possible initial settings of 17-bit LFSR do :
        - Run 17-bit LFSR to get 20 bytes of output
        - Subtract from CSS prefix $\Rightarrow$ Candidate 20 bytes output 25-bit LFSR
        - If consistent with 25-bit LFSR, found correct initial settings of both.
- Modern Stream Ciphers : eStream
    - $PRG : \{0,1\}^s(\text{seed}) \times R(\text{nonce}) \to \{0,1\}^n$
    - Nonce : A non-repeating value for a given key
        - $E(k, m; r) = m \oplus PRG(k; r)$
        - The pair $(k; r)$ is never used more than once
    - eStream : Salsa20
        - $\{0,1\}^{128 \text{ or } 256} \times \{0,1\}^{64} \to \{0,1\}^n, \max(n) = 2^{73}$ bits

        - $(k; r) := H(k, (r, 0))||H(k, (r, 1))|| \ldots$
        - h : invertible function
    - No known provably secure PRGs
    - No known attack better than exhaustive search
- Generating Randomness
    - Continuously add entropy to internal state
    - Entropy sources
        - Hardware RNG
        - Timing : Hardware interrupts

PRG Security Definitions

- Let $G : K \to \{0, 1\}^n$ be a PRG.
- Goal : Define what it means that a PRG is indistinguishable from a RNG
- Statistical Tests
    - An Algorithm A $s.t. A(x)$ outputs 0(not random) or 1(random)
- Advantage
    - Define : $Adv_{PRG}[A, G] = |\Pr_{K \xleftarrow{R} K}[A(G(k)) = 1] - \Pr_{r \xleftarrow{R} \{0,1\}^n}[A(r) = 1]| \in [0, 1]$
        - Close to 1 : Distinguishable
        - Close to 0 : Not Distinguishable
- Secure PRGs : Crypto Definition
    - Def : We say that $G : K \to \{0, 1\}^n$ is a secure PRG if for all efficient statistical tests A the advantage is negligible.
    - Easy fact : A secure PRG is unpredictable.

Theorem

- if $\forall i \in \{0, \ldots, n-1\}$ PRG G is unpredictable at position i, then G is a secure PRG.
- More generally, let P_1 and P_2 be two distributions over $0, 1^n$, we say that P_1 and P_2 are computationally indistinguishable (denoted $P_1 \approx_p P_2$) if for all efficient statistical tests A $|\Pr_{x \leftarrow P_1}[A(x) = 1] - \Pr_{x \leftarrow P_2}[A(x) = 1]| < negligible$
    - Example : Uniform Distribution

Semantic Security

- For $b = 0, 1$, define EXP(0) and EXP(1) as for b = 0,1: $W_b :=$ [event that $EXP(b) = 1$]

  , $Adv_{SS[A,E]} := |Pr[W_0] - Pr[W_1]| \in [0, 1]$

  - E is semantically secure if for all efficient A $Adv_{SS}[A, E]$ is negligible.

  - For all explicit $m_0, m_1 \in M$: $\{E(k, m_0)\} \approx_p \{E(k, m_1)\}$

- For all A: $Adv_{SS}[A, OTP] = |Pr[A(k \oplus m_0) = 1] - Pr[A(k \oplus m_1) = 1]|$

Stream Ciphers are Semantically Secure

Theorem

- $G : K \to \{0, 1\}^n$ is a secure PRG $\Rightarrow$ stream cipher E derived from G is semantically secure.
- For all semantic secure adversary A, there exists a PRG adversary B such that $Adv_{SS}[A, E] \leq 2 \cdot Adv_{PRG}[B, G]$

Block Ciphers

What is a Block Cipher?

- $R(k, m)$ is called a round function.

- Psuedo Random Function(PRF) defined over (K,X,Y):

$$F : K \times X \to Y$$

  such that exists efficient algorithm to evaluate $F(k, x)$.

- Psuedo Random Permutation(PRP) defined over (K,X):

$$E : K \times X \to X$$

  such that

  - exists a deterministic algorithm to evaluate $E(k, x)$.

  - The function $E(k, \dot{})$ is one to one.

  - Exists efficient inversion algorithm D(k,y).

- Functionally, any PRP is also a PRF – A PRP is a PRF where X=Y and is efficiently invertible.
- Secure PRFs
  - Let $F : K \times X \to Y$ be a PRF

    - $$\begin{cases} Funs[X, Y] : \text{the set of all functions from X to Y} \\ S_F = \{F(k, \cdot) s.t. k \in K\} \subseteq Funs[X, Y] \end{cases}$$

- Intuition : a PRF is secure if a random function in $Funs[X, Y]$ is indistinguishable from a random function in S_F.

- Secure PRPs(Secure Block Cipher)
  - Let $E : K \times X \rightarrow Y$ be a PRP
    - $$\begin{cases} Perms[X] : \text{the set of all one-to-one functions from } X \text{ to } Y \\ S_F = \{E(k, \cdot) s.t. k \in K\} \subseteq Perms[X, Y] \end{cases}$$
  - Intuition : a PRP is secure if a random function in $Perms[X, Y]$ is indistinguishable from a random function in S_F.

- An easy application($PRF \Rightarrow PRG$)
  - Let $F : K \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a secure PRF. Then the following $G : K \rightarrow \{0, 1\}^{nt}$ is a secure PRG :

$$G(k) = F(k, 0)||F(k, 1)|| \cdots ||F(k, t-1)$$

  - Key Property : Parallelizable
  - Security from PRF property

The Data Encryption Standard(DES)

- Core idea : Feistel Network
  - Given functions $f_1, \ldots, f_d : \{0, 1\}^n \rightarrow \{0, 1\}^n$
  - Goal : Build invertible function $F : \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}$
  - In symbols :

$$R_i = f_i(R_{i-1}) \oplus L_{i-1}$$
$$L_i = R_{i-1}$$

- Claim : For all $f_1, \ldots, f_d : \{0, 1\}^n \rightarrow \{0, 1\}^n$, Fiestel network $F : \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}$ is invertible
  - $$R_i = L_{i+1}$$
$$L_i = R_{i+1} \oplus f(R_i)$$

- Inversion is basically the same circuit with functions f_i applied in reverse order.

Theorem (Luby Rackoff)

- $f : K \times \{0,1\}^n \to \{0,1\}^n$ a secure PRF⇒ 3-round Fiestel

  $f : K^3 \times \{0,1\}^{2n} \to \{0,1\}^{2n}$ a secure PRP.

- DES : 16 round Fiestel Network
  - $f_i : \{0,1\}^{32} \to \{0,1\}^{32}$, $f_i(x) = F(k_i, x)$
  - 64 bits input → IP → Key Expansion and 16 round Feistel network → IP^{-1} → 64 bits output
  - To invert, use keys in reverse order
  - S-box : function $\{0,1\}^6$ to $\{0,1\}^4$, implemented as a looked-up table.
    - A bad example : inner product with mod
    - $S_i = A_i \cdot x$, we say that S_i is a linear function.
    - Then entire DES cipher will be linear.
  - Choosing the S-boxes and P-box at random would result in an insecure block cipher(2^{24} outputs)
    - No output bits should be close to a linear function of the input bits
    - S-boxes are 4-1 maps.

Exhaustive Search Attack

- Goal : given a few input output pairs($m_i, c_i = E(k, m_i)$) $i = 1, 2, 3$, find key k

Lemma

- Suppose DES is an ideal cipher(2^{56} random invertible functions), then $\forall m, c$ there is at most one key k such that $c = DES(k, m)$ with probability $\geq 99.5\%$.

- For two DES pairs($m_1, c_1 = DES(k, m_1)$), ($m_2, c_2 = DES(k, m_2)$) unicity probability $\approx 1 - \frac{1}{2^{71}}$
  - For AES 128, the probability is $1 - \frac{1}{2^{128}}$
  - Two input/output pairs is enough for exhaustive search
- DES Challenge
  - 56-bit ciphers should not be used.
- Strengthening DES
  - Triple-DES

- Let $E : K \times M \to M$ be a block cipher
- Define $3E : K^3 \times M \to M$ as

  $3E((k_1, k_2, k_3), m) = E(k_1, D(k_2, E(k_3, m)))$
- Key-size : 168 bits
- k_2=k_1=k_3 to be DES
- 3 times slower
- Simple attack in time 2^{118}
- Prevent meet-in-the-middle attack

- If Double-DES
  - 112 bits for DES
  - Build table and sort on 2nd column
  - For all $k \in \{0, 1\}^{56}$ do test if D(k,C) is in 2nd column
  - If so then $E(k^i, M) = D(k, C) \Rightarrow (k^i, k) = (k_2, k_1)$
- Meet in the middle attack
  - Time : 2^{63}
  - On 3 DES become 2^{118}
  - Space : 2^{56}
- DESx
  - $E : K \times \{0, 1\}^n \to \{0, 1\}^n$ be a block cipher
  - Define EX as $EX((k_1, k_2, k_3), m) = k_1 \oplus E(k_2, m \oplus k_3)$
  - Key-len : 64 + 56 + 64 = 184 bits
  - Attack in time 2^{120}

More Attack on Block Ciphers

- Implementation
  - Side Channel Attacks
    - Measure time to do encode and decode, measure power for encode and decode
  - Fault Attacks
    - Computing errors in the last round
- Linear and Differential Attacks
  - Given many input output pairs, we can recover keys in time less than 2^{56}

- Linear Cryptanalysis
  - Suppose for random k, m :
    $Pr[m[i_1] \oplus \cdots \oplus m[i_r] \oplus c[j_j] \oplus \cdots \oplus c[j_v] = k[l_1] \oplus \cdots \oplus k[l_u]] = \frac{1}{2} + \epsilon$
  - For DES, this ϵ exists with $1\frac{1}{2^{21}}$

Theorem

- Given $\frac{1}{\epsilon^2}$ random (m, c=DES(k,m)) pairs then

  $k[l_1, \ldots, l_u] = MAJ[m[i_1] \oplus \cdots \oplus m[i_r] \oplus c[j_j] \oplus \cdots \oplus c[j_v]]$ with probability 97.7%

  $\Rightarrow$ with $1/\epsilon^2$ pairs can find the key in time $1/\epsilon^2$

  - For DES, use 2^{42} to find 14 bits key in 2^{42}
  - Brute force remaining in time 2^{42}
  - Total 2^{43}
  - Problem from S_5

- Quantum Attacks
  - Generic Search Problems
    - Let $f : X \rightarrow \{0, 1\}$ be a function.
    - Goal : Find $x \in X$ such that f(x) = 1
  - Classical
    - Best in $O(|X|)$
  - Quantum
    - $O|X|^{\frac{1}{2}}$
    - Unknown for construction

The AES Block Cipher

- Key Size : 128, 192, 256 bits
- Block Size : 128 bits
- Sub-perm Network
  - Byte Substitution
    - A 1-byte S box. 256 byte table
    - Easily computable
  - Row Shifting

- Column Mixing
  - 10 rounds
- Key Expansion
  - 16 bytes to 176 bytes

- Trade-offs
  - Pre-compute round functions(largest code, fastest lookup with table and xors)
  - Pre-compute S box only(smaller code, slower lookup)
  - No pre-computation(smallest and slowest)
- Faster on OpenSSL than on hardwares
- Attacks
  - Best key recovery attack
    - 4 times faster than exhaustive search
  - Related key attack on AES 256
    - Given 2^{99} input/output pairs from four related keys can recover in time 2^{99}

Block Ciphers from PRGs

- Let $G : K \rightarrow K^2$ be a secure PRG.
- Define one-bit PRF $F : K \times \{0, 1\} \rightarrow K$ as F(k,x\in{0,1})=G(k)[x]$

Theorem

- If G is a secure PRG, then F is a secure PRF.

- Extension
  - Let $G : K \rightarrow K^2$, define $G_1 : K \rightarrow K^4$ as $G_1(k) = G(G(k)[0])||G(G(k)[1])$
    - 2-bit PRF
  - Extends more
- Extending more : the GGM PRF
  - Not used in practice due to slow performance

Using Block Ciphers

Review : PRPs and PRFs

- Block Ciphers : Crypto Work Horse
- PRF is defined if there exists an efficient algorithm to evaluate the output.
- PRP is defined there exists efficient deterministic algorithm to evaluate the output.

- The function is one-to-one.
- Also exists efficient inversion algorithm.

- A PRF is secure if a random function in F is indistinguishable from a random function in S_F.
  - The difference between two advatanges with different experiments is negligible.
- A secure PRP is like a secure PRF, but it is invertible.
  - Example : AES, 3DES

PRF Switching Lemma

- Any secure PRP is also a secure PRF, if $|X|$ is sufficiently large.
  - Lemma : Let E be a PRP over (K, X). Then for any q-query adversary A :

    $|Adv_{PRF}[A, E] - Adv_{PRP}[A, E]| < \frac{q^2}{2|X|}$

    - We try to make the lastest component negligible.

Modes of Operation : One Time Key

- Example : Encrypted Email, New key for every message
- Goal : Build secure encryption from a secure PRP
  - Adversary's Power : Sees only one ciphertext
  - Adversary's Goal : Learn information about plaintext from ciphertext(Semantic security)
- Incorrect use of a PRP
  - Electronic Code Block(ECB)
    - Problem : If $m_1 = m_2$, then $c_1 = c_2$.
    - In pictures, the contour can be observed.
    - Not semantically secure if the message contain more than one block.
- $Adv_{SS}[A, OTP] = |Pr[EXP(0)] = 1 - Pr[EXP(1)] = 1|$ should be negligible.
- Secure Construction I :
  - Deterministic counter mode
  - A stream cipher with a counter, built from a PRF.

Theorem

- For any $L > 0$, if F is a secure PRF over (K,X,X) then E_{DETCTR} is semantic secure cipher over (K, X^L, X^L). In particular, for any efficient adversary A attacking

E_{DETCTR}, there exists an efficient PRF adversary B such that

$Adv_{SS}[A, DETCTR] = 2 \cdot Adv_{PRF}[B, F]$

- RHS is negligible

Security for Many-Time Key

- Example : File Systems, IPSec
- Key used more than once ⇒ adversary sees many ciphertexts with same key
  - Adversary's power : Chosen-Plaintext Attack(CPA)
    - Obtain the encryption of arbitrary message of his choice
    - Real life modeling
  - Adversary's Goal : Break semantic security
- If the adversary wants $c = E(k, m)$, it queries with $m_{j,0} = m_{j,1} = m$
- Definition : E is semantically secure under CPA if for all efficient A: $Adv_{CPA}[A, E] = |Pr[EXP(0) = 1] - Pr[EXP(1)] = 1|$ is negligible.
  - Not secure if always give the same output for the same message.
    - The attacker can learn that two encrypted files, packets are the same
    - Leads to significant attack if the message space M is small.
  - Output should be different when using many-time keys.
- Solution 1 : Randomized Encryption
  - Ciphertext must be longer than plaintext
    - CT size = PT size + # of random bits
- Solution 2 : Nonce-based Encryption
  - Nonce : A value that changes from message to message
    - (k,n) pair never uses more than once
  - Method 1 : Nonce is a counter(Packet Counter)
    - Keep states
    - If decryptor has same state, need not send nonce with cipher text.
  - Method 2 : Encryptor chooses a random nonce.
    - Systems should be secure if nonces are chosen adversarially
    - All nonces must be distinct.

Modes of Operation : Many-Time Key(CBC)

- Construction 1 : CBC with random IV
  - Choose random $IV \in X$

- $c[0] = E(k, IV \oplus m[0])$

CBC Theorem

- For any L > 0, if E is a secure PRP over (K,X) then E_{CBC} is a semantically secure under CPA over (K, X^L, X^{L+1}).
- In particular, for a q-query adversary A attacking E_{CBC}, there exists a PRP adversary B such that $Adv_{CPA}[A, E_{CBC}] \leq 2 \cdot Adv_{PRP}[B, E] + \frac{2q^2L^2}{|X|}$
    - Only secure if the second item is negligible.
    - Can be calculated for how many blocks is fine before we need to change key.
- Warning : An attack on CBC with random IV
    - If the attacker can predict the IV, then it is not CPA-secure.
        - Bug in SSL / TLS 1.0
    - Nonce-based CBC
- An example Crypto API(OpenSSL)
    - When the nonce is not random, it needs to be encrypted before used
- A CBC technicality: Padding
    - Removed during encryption
    - If no pad needed, add a dummy block.

Modes of Operation : Many-Time Key(CTR)

- Construction 1 : Random CTR-Mode
    - Chooses a random IV, and +1 every block.
    - Parallelizable

Counter-Mode Theorem

- For any L > 0, If F is a secure PRF over (K,X,X) then E_{CTR} is semantic secure under CPA over (K, X^L, X^{L+1}).
    - In particular, $Adv_{CPA}[A, E_{CTR}] \leq 2 \cdot Adv_{PRF}[B, F] + \frac{2q^2L}{|X|}$
    - Better than CBC for the second item.
    - In AES, after 2^{32} ciphertexts each of len 2^{32}, the key must be changed.(Total of 2^{64} AES blocks)
- Construction 2 : Nonce CTR-Mode
    - To ensure $F(k, x)$ is never used more than once, choose IV as :

        - IV(128bits) = nonce(64bits) + counter(64bits)
        - Counter starts at 0 for every message.

	CBC	ctr mode
uses	PRP	PRF
parallel processing	No	Yes
Security of rand. enc.	q^2 L^2 << |X|	q^2 L << |X|
dummy padding block	Yes	No
1 byte msgs (nonce-based)	16x expansion	no expansion

(for CBC, dummy padding block can be solved using ciphertext stealing)

- Neither mode ensures data Integrity.
- Further Reading
    - A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation, M. Bellare, A. Desai, E. Jokipii and P. Rogaway, FOCS 1997
    - Nonce-Based Symmetric EncrypAon, P. Rogaway, FSE 2004

Message Integrity

Message Authentication Codes(MAC)

- Goal : Integrity, no confidentiality.
- Examples : Protecting public binaries on disk, Protecting banner ads on web page.
- Definition : MAC |= (S,V) defined over (K,M,T) is a pair of algorithms
    - S(k,m) outputs t in T
    - V(k,m,t) outputs yes or no
- Integrity requires a secret key.
    - Cyclic Reduandancy Check
    - Attacker can easily modify message m and re-compute CRC.
    - CRC designed to detect random, not malicious errors.
- Attackers power : Chosen Message Attack
    - For messages, attackers are given tags.
- Attackers Goal : Existential Forgery
    - Produce some new message tag pair
        - Attacker cannot produce a valid tag for a new message.
        - Given (m,t), attacker cannot produce (m, t')
- Secure MACs

- A MAC is secure if the challenger gives a new message tag pair, and the probability of the verifier saying yes is negligible.
- Example : Protecting System Files
  - MAC can detect every modified file.
    - If the key derives from user's password, then only the user know the key.

MAC Based on PRFs

- Secure PRF $\Rightarrow$ Secure MAC
  - Accept message if tag = F(k,m)

Theorem

- If $F : K \times X \to Y$ is a secure PRF and $\frac{1}{|Y|}$ is negligible, then I_F is a secure MAC.
  - $Adv_{MAC}[A, I_F] \le Adv_{PRF}[B, F] + \frac{1}{|Y|}$

- Examples
  - AES : A MAC for 16-bytes messages.
  - How to convert small MAC into big MAC?
    - CBC-MAC
    - HMAC
  - Convert a small PRF to big PRF.
- Truncating MACs based on PRFs

Lemma

- Suppose $F : K \times X \to \{0,1\}^n$ is a secure PRF. Then so is

  $F_t(k, m) = F(k, m)[1, \dots, t]$ for all $1 \le t \le n$ as long as $\frac{1}{2^t}$ is still negligible.

CBC-MAC and NMAC

- Goal : Given a PRF for short messages(AES), construct a PRF for long messages.
  - Let $X = \{0,1\}^{128}$
- Construction 1 **Encrypted CBC-MAC**
  - Let $F : K \times X \to X$ be a PRP. Define a new PRF $F_{ECBC} : K^2 \times X^{\le L} \to X$
  - $X^{\le L} = \cup_{i=1}^{L} X^i$
- Construction 2 **NMAC (Nested MAC)**

- Let $F : K \times X \rightarrow K$ be a PRF. Define a new PRF $F_{NMAC} : K^2 \times X^{\leq L} \rightarrow K$

- If the last step isn't applied to ECBC or NMAC, then the MAC can be forged with one chosen message query.
  - Choose an arbitrary one-block message $m \in X$
  - Request tag for message. Get $t = F(k, m)$
  - Output t as MAC forgery for the 2-block message $(m, t \oplus m)$

Theorem for Analysis

- For any L > 0, for every efficient q-query PRF adversary A attacking F_{ECBC} or F_{NMAC}

  , there exists an efficient adversary B such that :
  $Adv_{PRF}[A, F_{ECBC}] \leq Adv_{PRP}[B, F] + 2\frac{q^2}{|X|}$ and

  $Adv_{PRF}[A, F_{NMAC}] \leq q \cdot L \cdot Adv_{PRP}[B, F] + \frac{q^2}{2|K|}$

  + Secure when q $<< |X|^{\frac{1}{2}} / |K|^{\frac{1}{2}}$

- The security bounds are tight.
  - Suppose the underlying PRF is a PRP.
  - Then both PRFs
    - $\forall x, y, w : F_{BIG}(k, x) = F_{BIG}(k, y) \Rightarrow F_{BIG}(k, x||w) = F_{BIG}(k, y||w)$
  - Generic Attack
    - Issue $|Y|^{\frac{1}{2}}$ message queries for random messages in X. Obtain (m_i, t_i) for $i = 1, \ldots, |Y|^{\frac{1}{2}}$
    - Find a colllison (By Birthday Paradox)
    - Choose some w and query for $t := F_{BIG}(k, m_u||w)$
    - Output forgery $(m_v||w, t)$

- Better Security : A Random Construction
  - Let $F : k \times X \rightarrow X$ be a PRF.
  - Result : MAC with tags in X^2
  - Security : $Adv_{MAC}[A, I_{RCBC}] \leq Adv_{PRP}[B, F] \cdot (1 + 2\frac{q^2}{|X|})$
    - For 3DES : 2^{32} messages with one key

- Comparison

- ECBC-MAC is a commonly-used AES MAC.
  - CCM Encryption Mode
  - CMAC
- NMAC is usually not used with 3DES or AES
  - Since it needs to change key on every block, requring recomputing AES key expansion.
  - Basis for HMAC

MAC Padding

- Deal with the situation for the length of the message is not multiple of block-size.
  - If pad with 0000...
    - $pad(m) = pad(m||0)$
  - Padding must be invertible(one-to-one).
- ISO : Pad with 10000...00. Add new dummy block if needed.
  - 1 indicates the beginning.
- CMAC
  - Variants of CBC-MAC where $key = (k, k_1, k_2)$
  - No final encryption step
  - No dummy block

PMAC and Cater-Wegman MAC

- ECBC and NMAC are sequential.
  - Can we build a parallel one from a small PRF?
- Construction 3 : **PMAC-Parallel MAC**
  - P(k, i) : an easy-to-compute function
  - $key = (k, k_1)$
  - Padding similar to CMAC
  - Let $F : K \times X \to X$ be a PRF. Define a new PRF $F_{PMAC} : K^2 \times X^{\leq L} \to X$

PMAC Theorem for Analysis

- If F is a secure PRF over (K,X,X), then F_{PMAC} is a secure PRF over $(K, X^{\leq L}, X)$.
  - $Adv_{PRF}[A, F_{PMAC}] \leq Adv_{PRF}[B, F] + \frac{2q^2L^2}{|X|}$
  - Secure as long as $qL << |X|^{\frac{1}{2}}$

- PMAC is incremental.
  - Deal with modification easily because of independent blocks.
- One Time MAC
  - Def : If $I(S,V)$ is a secure MAC if for all efficient A
    - $Adv_{1MAC}[A,I] = Pr[\text{Chal. outputs } 1]$ is negligible.
    - Outputs 1 means the case of the verifier says "yes" but $(m,t) \neq (m_1,t_1)$
  - One-time MAC can be secure against all adversaries and faster than PRF-based MACs/
- One-time Security(Unconditional)
  - $\forall m_1 \neq m_2, t_1, t_2 : Pr_{a,b}[S((a,b),m_1) = t_1 | S((a,b),m_2) = t_2] \leq \frac{L}{q}$
- One Time MAC ⇒ Many Time MAC
  - Let (S,V) be a secure one-time MAC over $(K_I, M, \{0,1\}^n)$
  - Let $F : K_F \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRF.
  - Carter-Wegman MAC
    - $CW((k_1,k_2),m) = (r, F(k_1,r) \oplus S(k_2,m))$ for random r

Theorem

- If (S,V) is a secure one-time MAC and F a secure PRF. Then CW is a secure MAC outputting tags in $0,1^{2n}$

Collision Resistance

Introduction

- Let $H : M \to T$ be a hash function.(|M|>>|T|)
  - A function H is collision resistant if for all efficient algorithms A: $Adv_{CR}[A,H] = Pr[A \text{ outputs collision for } H]$ is negligible.
  - SHA-256
- MACs from collision Resistance
  - Let $I = (S,V)$ be a MAC for short messages over (K,M,T).
  - Let $H : M^{big} \to M$

- Definition : $I^{big} = (S^{big}, V^{big})$ over (K, M^{big}, T) as

  $S^{big}(k, m) = S(k, H(m)); V^{big}(k, m, t) = V(k, H(m), t)$

Theorem

- If I is a secure MAC and H is collision resistant, then I^{big} is a secure MAC.
  - Suppose the adversary can find the collision, then S^{big} is unsecure under 1-chosen message attack.
    - Ask for $t \leftarrow S(k, m_0)$
    - Output (m_1, t) as forgery
- Examples
  - Software Packages
    - Attacker cannot modify package without detection
    - No key needed(public verifiability)
    - Requires read-only space

Generic Birthday Attack

- Let $H : M \rightarrow \{0, 1\}^n$ be a hash function($|M| >> 2^n$)
  - A generic algorithm to find a collision in time $O(2^{\frac{n}{2}}) hashes$
    - Chooses $2^{\frac{n}{2}}$ random messages in M: $m_1, m_2, \ldots, m_{2^{\frac{n}{2}}}$
    - For $i = 1, \ldots, 2^{\frac{n}{2}}$ compute $t_i = H(m_i) \in \{0, 1\}^n$
    - Look for a collision, if not found, go back to the first step.

Theorem

- Let $r_1, r_2, \ldots, r_n \in \{1, \ldots, B\}$ be independent distributed integers.
  - When $n = 1.2 \times B^{\frac{1}{2}}$ then $Pr[\exists i \neq j : r_i = r_j] \geq \frac{1}{2}$
  - Expected Number of Iteration : 2
  - Time : $O(2^{\frac{n}{2}})$
  - Space : $O(2^{\frac{n}{2}})$

- Best known collision finder for SHA-1 requires 2^{51} hash evaluations.
- Quantum Collision Finder
  - For exhaustive search, it gives the original time complexity a square root.

- For hash function collision finder, it reduces the square root to cubic root.

The Merkle-Damgard Paradigm

- Goal : Collsion resistant hash functions
  - Give C.R. function for short messages, construct one for long messages
- Give $h : T \times X \rightarrow T$ (Compression Function), we obtain $H : X^{\leq L} \rightarrow T$
  - H_i : Chaining Variables
  - PB : Padding Block (100...0||Message Length(64bits))
    - If no space for PB, add another block.

Theorem

- If h is collision resistant then so is H.
  - Proof : Collision on H $\Rightarrow$ Collision on h
    - It will not iterate all the way to the beginning because it will cause a contradiction $M = M'$
- To construct a C.R. functino, it suffices to construct compression function.

Constructing Compression Functions

- Let $E : K \times \{0,1\}^n \rightarrow \{0,1\}^n$ a block cipher.
- The Davies-Meyer compression function
  - $h(H, m) = E(m, H) \oplus H$

Theorem

- Suppose E is an ideal cipher(collection of |K| random permutations)
  - Finding a collision $h(H, m) = h(H', m')$ takes $O(2^{\frac{n}{2}})$ evaluations.
  - Best possible

- Other instructions
  - Miyaguchi-Preneel : $h(H, m) = E(m, H) \oplus H \oplus m$,

    $h(H, m) = E(H \oplus m, m) \oplus m$ and other 10 variants
- SHA-256
  - 512-bit key + 256-bit block + SHACAL-2 = 256-bit block
- Provable Compression Function

- Choose a random 2000-bit prime p and random $1 \leq u, v \leq p$.
  - For $m, h \in \{0, \dots, p-1\}$ define $h(H, m) = u^H \cdot v^m \pmod p$

- Fact : Finding collision for h is as hard as solving discrete-log mod p

HMAC : a MAC from SHA-256

- Standardized Method : HMAC(Hash-MAC)
  - H : hash function
    - Example : SHA-256, output is 256 bits
  - $S(k, m) = H(k \oplus opad || H(k \oplus ipad || m))$
  - Similar to the NMAC PRF
    - Main difference : k_1, k_2 are dependent.

  - Properties :
    - Built from a black-box implementation of SHA-256
    - Assumed to be a secure PRF, which can be proven under certain PRF assumptions
    - Security bounds similar to NMAC : It needs $\frac{q^2}{|T|}$ to be negligible.

Timing Attacks on MAC Verification

- Warning : Verification Timing Attacks
  - If using byte-by-byte comparison
    - Return false when the first inequality found
  - To compute tag for target message m do :
    - Query server with random tag
    - Loop over all possible first bytes and query server. Stop when verification takes a little longer than step 1
    - Repeat all tag bytes
  - Defense : Make string comparator always take the same time
    - It can be difficult to ensure due to optimizing compiler.
  - Defense : Check the return value of H

Authenticated Encryption

Active Attacks on CPA-Secure Encryption

- Encryption secure against tampering
  - Ensure both confidentiality and integrity

- Example : In TCP / IP
  - Change the information of the package(to different destination)
    - Use IPSec to verify the information
    - Use CBC with random IV (the attacker may get partial information)
- An Attack Using only Network Access
  - Remote terminal app : each key stroke encrypted with CTR mode
    - $\mathrm{checksum}(\mathrm{hdr}, D) = t \oplus \mathrm{checksum}(\mathrm{hdr}, D \oplus s) \Rightarrow D$
- Lesson
  - CPA security cannot guarantee secrecy under active attacks.

Definitions

- An authenticated encryption system (E, D) is a cipher where

  $E : K \times M \times N \rightarrow C, D : K \times C \times N \rightarrow M \cup \{\perp\}$

  - $\perp$: Ciphertext is rejected
  - Security
    - Semantic security under CPA attack.
    - Ciphertext integrity(Attackers cannot create new ciphertexts that decrypt properly)
  - (E, D) has ciphertext security if for all efficient A

    $\mathrm{Adv}_{\mathrm{CI}}[A, E] = Pr[\text{Challenger outputs } 1]$ is negligible.

    - CBC with random IV does not provide authenticated encryption.(No $\perp$)
    - IV manipulation attack
- Implication 1 : Authencity
  - Attacker cannot fool Bob into thinking a message was sent from Alice.
    - Message could be a replay.
- Implication 2 : Authenticated Encryption $\Rightarrow$ Security against chosen ciphertext attacks

Chosen Ciphertext Attacks

- Example
  - Adversary has a ciphertext c that it wants to decrypt. Often the adversary can fool the server into decrypting certain ciphertexts.
  - Often, the adversary can learn partial information about the plaintext.
- Adversary's Power

- Both CPA and CCA
- Obtain arbitrary messages of his choice.
- Decrypt ciphertext of his choice, other than challenge.
- Goal : Break semantic security

- E is CCA secure if for all efficient A
  $\mathrm{Adv}_{CCA}[A, E] = |Pr[EXP(0) = 1] - Pr[EXP(1)] = 1|$ is negligible.

Theorem

- Let (E,D) be a cipher that provides authenticated encryption. Then (E,D) is CCA secure.
  - $\mathrm{Adv}_{CCA}[A, E] \leq 2q \cdot \mathrm{Adv}_{CI}[B_1, E] + \mathrm{Adv}_{CPA}[B_2, E]$

- Ensures confidentiality against an active adversary that can decrypt some ciphertexts.
  - Does not prevent replay attacks
  - Does not account for side channels(timing attacks)

Constructions from Ciphers and MACs

- Combining Encryption and MAC
  - Let (E, D) be CPA secure cipher and (S,V) secure MAC.
    - Encrypt-then-MAC always provide A.E.
  - MAC-then-Encrypt may be insecure against CCA attacks.
    - However, when (E,D) is random-counter mode or rand-CBC, M-then-E provides A.E.
    - For random-CTR mode, one-time MAC is sufficient.

- Standards
  - All support authenticated encryption with associated data.
  - All are nonce-based.

Case Study : TLS

- Unidirectional Keys
  - $k_{b \to s}, k_{s \to b}$

- Stateful Encryption
  - Each side maintains two 64-bit counters
    - Initialized to 0 when session started, and +1 for every record.
    - Purpose : Replay defense

- Browser Side : $enc(k_{b \to s}, data, ctr_{b \to s})$

- Step 1 : $\mathrm{tag} \leftarrow S(k_{mac}, [++\ ctr_{b \to s} || header || data]$
    - The counter is not transmitted in the packet.
- Step 2 : Pad $[header||data||tag]$ to AES block size
- Step 3 : CBC encrypt with k_{enc} and new random IV
- Step 4 : Prepend header

- Server Side : $dec(k_{b \to s}, record, ctr_{b \to s})$
  - Step 1 : CBC decrypt record using k_{enc}
  - Step 2 : Check pad format
  - Step 3 : Check tag on $[++\ ctr_{b \to s} || header || data]$

- If IV for CBC is predictable
  - IV for next record is last ciphertext block of current record. Not CPA secure.
  - BEAST attack
- Padding oracle during decryption
  - If the pad is invalid, send decryption failed alert.
  - If the MAC is invalid, send bad_record_MAC alert.
    - Attacker learns info about plaintext.
- When decryption fails, do not explain why.
- The TLS header leaks the length of TLS records
  - Also can be inferred by network traffic.
  - Leaking the length may reveal sensitive information.
  - No easy solution.
- WEP : CRC
  - Two time pads
  - Related PRG seeds
  - Active attack
    - By using the linearity and predictability of CRC

CBC Paddings Attacks

- Two types of error
  - Padding error
  - MAC error
- Suppose the attacker can differentiate two errors
  - Padding Oracle : Attacker submits ciphertext and learns if last bytes of plaintext are a valid pad.

        - Chosen Ciphertext Attack
        - Bad pad / Bad MAC
    - Using a pad oracle(Attacker has ciphertext $c = (c[0], c[1], c[2])$ and it wants $m[1]$ in CBC encryption)
        - Let g be a guess for the last byte of $m[1]$
        - Submit $(IV, c^{'}[0], c[1])$ to padding oracle $\Rightarrow$ attacker learns if last-byte = g.
        - Repeat with $g = 0, 1, \ldots, 255$ to learn last byte of $m[1]$
        - Then use a (02, 02) pad to learn the next byte and so on...
- IMAP over TLS
    - TLS renegotiates the key when an invalid record is received
        - Every 5 minute the client sends login message to the server
        - Exact same attack works despite new keys, and it can recover the password in a few hours.

Attacking Non-Atomic Decryption

- SSH Binary Packet Protocol
    - Decryption
        - Decrypt packet length field only
        - Read as many packets as the length specifies
        - Decrypt remaining cipher blocks
        - Check MAC tag and send error response if invalid
    - Send bytes one at a time, and the attacker learns 32 LSB bits of m.
    - Solution
        - Send the length field unencrypted but MACed
        - Add a MAC of (seq-num, length) right after the len field.

Odds and Ends

Key Derivation

- Deriving many keys from one
    - Typical Scenario
        - A single source key(SK) is sampled from hardware random number generator / a key change protocol.
    - Need many keys to secure session
        - Unidirectional keys / Multiple keys for nonce-based CBC
    - Goal : Generate many keys from this one source key

- Define Key Derivation Function(KDF) as :
  - $\mathrm{KDF}(\mathrm{SK}, \mathrm{CTX}, \mathrm{L}) := \mathrm{F}(\mathrm{SK}, (\mathrm{CTX}||0))||\mathrm{F}(\mathrm{SK}, (\mathrm{CTX}||1))|| \cdots ||\mathrm{F}(\mathrm{SK}, (\mathrm{CTX}||\mathrm{L}))$
  - CTX : A string that uniquely identifies the application
    - Even if two apps sample the same SK, they will get independent keys.
  - If the source key is not uniform, the output of the PRF may not look random.
  - Source key is usually not uniformly random.
    - Key Exchange Protocol : Keys are uniform in some subset of K
    - Hardware RNG : Produce biased output
- Extract-then-Expand Paradigm
  - Extract pseudo-random key k from source key SK.
    - Salt : A fixed non-secret string chosen at random
  - Expand k by using it as a PRF key as before.
- HKDF : a KDF from HMAC
  - Implements the EtE paradigm
    - Extract : Use $\mathrm{k} \leftarrow \mathrm{HMAC}(\mathrm{salt}, \mathrm{SK})$
    - Expand : Using HMAC as a PRF with key k
- Password-Based KDF(PBKDF)
  - Deriving key from passwords
    - Do not use HKDF : insufficient entropy
    - Derived keys will be vulnerable
  - Defenses
    - Salt and a slow hash function
  - Standard approach : PKCS#5(PBKDF1)
    - $\mathrm{H}^{(c)}(\mathrm{pwd}||\mathrm{salt})$

Deterministic Encryption

- No nonce
  - Enable later lookup
- Problem : Cannot be CPA secure
  - The attacker can tell if the two ciphertexts encrypt the same message.
    - Leaks information
  - Leads to significant attacks if the message space is small.
    - Equal ciphertexts mean the same index
- A solution : Unique messages

- Suppose the encryptor never encrypts the same message twice.
  - When the encryptor chooses message at random from a large message space.
  - Message structure ensures uniqueness(User ID)
- E is semantically secure under deterministic CPA if for all efficient A

  $Adv_{dCPA}[A, E] = |Pr[EXP(0) = 1] - Pr[EXP(1)] = 1|$ is negligible.

  - All messages are distinct.
- CBC with fixed IV is not deterministic CPA secure.
  - Counter mode with FIV is also not CPA secure.

Deterministic Encryption Constructions : SIV and Wide PRP

- Deterministic encryption needs for maintaining an encrypted database index.
  - Lookup records by encrypted index
- Deterministic CPA security
  - Security if never encrypt same message twice using the same key.
    - The pair (key, msg) is unique.
- Construction 1 : Synthetic IV(SIV)
  - Let (E,D) be a CPA-secure encryption.
    - $E(k, m; r) \rightarrow c$
  - Let $F : K \times M \rightarrow R$ be a secure PRF.
    - Define : $E_{det}((k_1, k_2), m) = \begin{cases} r \leftarrow F(k_1, m) \\ c \leftarrow E(k_2, m; r). \\ output r \end{cases}$

Theorem

- E_{det} is semantically secure under CPA.
  - Well suited for messages longer than 1 AES block.

- Ensuring ciphertext integrity
  - Goal : Deterministic authenticated encryption
  - SIV-CTR
    - SIV where cipher is counter mode with random IV.

Theorem

- If F is a secure PRF and CTR from F_{ctr} is CPA-secure. Then SIV-CTR from F, F_{ctr} provides DAE.

- Construction 2 : PRP

Theorem

- (E,D), which is a secure PRP, is semantically secure under deterministic CPA.
  - Using AES : Deterministic CPA secure encryption for 16 byte messages. If we want longer messages, then need PRPs on larger message spaces.

- EME : Constructing a wide block PRP
  - a PRP on $\{0,1\}^N$ for $N >> n$
  - Key = (K, L)
  - $M \leftarrow MP \oplus MC$
  - Performance : 2x slower than SIV
- PRP-based DAE

Theorem

- Let (E,D) be a secure PRP.
  - $E : K \times (X \times \{0,1\}^n) \rightarrow X \times \{0,1\}^n$
- If $\frac{1}{2^n}$ is negligible $\Rightarrow$ PRP-based encryption provides DAE.
  - $Pr[LSB_n(\pi^{-1}(c)) = 0^n] \leq \frac{1}{2^n}$

Tweakable Encryption

- Disk Encryption : No Expansion
  - Sectors on disk are fixed size.
    - Encryption cannot expand plaintext
    - Deterministic encryption, no integrity

Lemma

- If (E,D) is a deterministic CPA secure cipher with M=C, then (E, D) is a PRP.
  - Every sector will be encrypted with a PRP.

- Problem : Two sectors may have the same content.
  - Leaks some information as the ECB mode.
  - Use different keys for every block
- Problem : Attacker can tell if a sector is changed and then reverted

- Managing keys : $k_t = PRF(k, t)$
- Goal : Construct many PRPs from a key $k \in K$
    - Syntax : $E, D : K \times T \times X \rightarrow X$ for every $t \in T$ and $k \leftarrow K$
        - $E(k, t, \cdot)$ is an invertible function on X, indistinguishable from random.
- Application : Use sector number as the tweak.
    - Every sector gets its own PRP.
- Example 1 : Trivial Construction
    - $E_{tweak}(k, t, x) = E(E(k, t), x)$
        - to encrypt n blocks need 2n evaluations of E
- Example 2 : XTS Tweakble Block Cipher
    - $E_{tweak}((k_1, k_2), (t, i), x)$
    - $N \leftarrow E(k_2, t)$
        - to encrypt n blocks need n + 1 evaluations of E
    - Block level PRP, not sector level
- It is necessary to encrypt the tweak before using it.

Format Preserving Encryption(FPE)

- Encrypting credit card numbers
    - Goal : End-to-end encryption
        - Intermediate processors expect to see a credit card number
        - Encrypted credit card should look like a credit card
- Given $0 \leq s \leq 2^n$, build a PRP on 0 to s-1 from a secure PRF $F : K \times 0, 1^n \rightarrow 0, 1^n$
    - s : total number of credit cards
    - To encrypt a credit card number
        - map given CC# to $\{0, \dots, s - 1\}$
        - Apply PRP to get an in $\{0, \dots, s - 1\}$
        - map output back to CC#
    - From $\{0, 1\}^n$ to $\{0, 1\}^t$
        - Let t be such $2^{t-1} \leq s \leq 2^t$
        - Method : Luby-Rackoff with $F' : K \times \{0, 1\}^{\frac{t}{2}} \rightarrow \{0, 1\}^{\frac{t}{2}}$

    - Given PRP $(E, D) : K \times \{0,1\}^t \to \{0,1\}^t$, we build

      $(E', D') : K \times \{0, \dots, s-1\} \to \{0, \dots, s-1\}$

        - $E'(k, x)$: on input $x \in \{0, \dots, s-1\}$ do $y \leftarrow x$; do $\{y \leftarrow E(k, y)\}$ until y is in 0 to s-1, then output y
        - Expected Iteration : 2
        - $p = \frac{s}{2^t}$
- Security
    - For all A, there exists B: $PRP_{adv}[A, E] = PRP_{adv}[B, E']$
        - Intuition : For all sets $y \subseteq X$, random permutation $\pi : X \to X$ gives another random permutation
    - No integrity

Basic Key Exchange

Trusted Third Parties

- Key Management
    - Problem : n users and storing mutual secret keys is difficult.
        - Total : O(n) keys per user
    - Online Trusted Third Party
        - Every user only remembers one key.
- A toy protocol to generate keys
    - Alice wants a shared key with Bob.
    - Eavesdropping security only.
    - Eavesdropper sees $E(k_A, A, B||k_{AB}), E(k_B, A, B||k_{AB})$
        - (E,D) is CPA-secure $\Rightarrow$ Eavesdropper learns nothing about k_{AB}
        - TTP needed for every key exchange and knows all session keys.
        - Basis of Kerberos system
    - Insecure against active attacks
        - Replay attacks
        - Attacker records session between Alice and Bob and replays session to Bob
- Without the TTP
    - Available to do so
        - Merkle, Diffie-Hellman, RSA, ID-based encryption, Functional Encryption

Merkle Puzzles

- Goal : Alice and Bob want a shared key but unknown to eavesdropper.
  - Security against eavesdropping only(no tampering)
- Generic Symmetric Crypto(Merkle Puzzles)
  - Inefficient
  - Puzzles : Problems that can be solved with some effort
  - Alice : Prepare 2^{32} puzzles
    - For every puzzle, choose random $P_i \in \{0,1\}^{32}$ and $x_i, k_i \in \{0,1\}^{128}$
    - Set $puzzle_i \leftarrow E(0^{96}||P_i, Puzzle\#\ x_i||k_i$
    - Send the puzzles to Bob
  - Bob : Choose a random puzzle and solve it.
    - Obtain (x_j, k_j) and send x_j to Alice.
  - Alice : Lookup puzzle with x_j. Use k_j as the shared secret
  - Alice, Bob : $O(n)$
  - Eavesdropper : $O(n^2)$
- Impossibility Result
  - Unknown for a better gap with symmetric cipher
  - Quadratic gap is best possible if the cipher is a black box oracle

The Diffie-Hellman Protocol

- For a large prime p, fix an integer g in $\{1, \dots, p\}$
  - Alice : Choose random a in 1 to p - 1
  - Bob : Choose random b in 1 to p - 1
  - $B^a (\bmod p) = (g^b)^a = k_{AB} = g^{ab} (\bmod p) = (g^a)^b = A^b (\bmod p)$
  - Security
    - Eavesdropper sees $p, g, A = g^a (\bmod p), B = g^b (\bmod p)$, it wants to compute $g^{ab} (\bmod p)$
  - Define $DH_g(g^a, g^b) = g^{ab} \bmod p$
- Suppose p is n bits long, best known algorithm GNFS
  - $e^{\tilde{O}(\sqrt[3]{n})}$

- Slow transition away from mod p to elliptic curves
- Insecure against man-in-the-middle attack
  - The attacker sends its a', b' to Alice and Bob

Public Key Encryption

- Goal : Alice and Bob want shared secret unknown to the eavesdropper.
- Definition
  - A public encryption system is a triple of algorithms (G, E, D)
    - G(): Randomized algorithm that outputs a key pair (pk, sk)
    - E(pk, m): Randomized algorithm that takes $m \in M$ and outputs $c \in C$
    - D(sk, c): Deterministic algorithm that takes $c \in C$ and outputs $m \in M$ or $\perp$
  - $D(sk, E(pk, m)) = m$
- Security(Eavesdropping)
  - Adversary sees $pk, E(pk, x)$ and wants $x \in M$
  - Able to derive session key from x
  - The protocol is vulnerable to MitM.
- Constructions generally rely on hard problems from number theory and algebra.

Introduction to Number Theory

Notation

- The base of key exchange protocols, digital signatures and public-key-encryption.
  - N denotes a positive integer.
  - p denotes a prime.
  - $Z_N = \{0, 1, 2, \ldots, N-1\}$
    - Do addition and multiplication under modulo n
    - Arithmetic works as normal.
- Greatest Common Divisor
  - For integers x and y, $\gcd(x, y)$ is the greastest common divisor of x,y.
    - For all integers x and y, there exists integers a, b such that $ax + by = \gcd(x, y)$, a, b can be found efficiently using the extended Euclidean algorithm.
  - If gcd(x,y) = 1, then x and y are relatively prime.

- Modular Inversion
  - The inverse of x in Z_n is an element y in Z_n such that $xy = 1$, y is denoted x^{-1}.
    - Let N be an odd integer, the inverse of 2 in Z_N is $\frac{N+1}{2}$.

Lemma

- x in Z_N has an inverse iff $gcd(x, N) = 1$

- $Z_N^* = \{x \in Z_N : gcd(x, N) = 1\}$
  - For prime p, Z_N^* is 1 ~ p -1
  - No 0
  - Can find x^{-1} using extended Euclidean algorithm
- Solve $ax + b = 0$ in Z_N
  - Solution : $x = -ba^{-1}$ in Z_N
  - Using Extended Euclidean algorithm to find a^{-1} with time $O(\log^2 N)$

Fermat and Euler

- Find inverses using Euclidean Algorithm, time : $O(n^2)$

Fermat Theorem

- Let p be a prime, $\forall x \in (Z_p)^* : x^{p-1} = 1$ in Z_p
  - $x^{-1} = x^{p-2}$ in Z_p
  - Less efficient

- Generating random primes(1024 bits)
  - Choose a random integer p $\in [2^{1024}, 2^{1025} - 1]$
  - Test if $2^{p-1} = 1$ in Z_p
    - If so, output p and stop
    - If not, choose again
  - $Pr[\text{p is not a prime}] < 2^{-60}$

Theorem

- $(Z_p)^\star$ is a cyclic group, that is $\exists g \in (Z_p)^\star$ such that $\{1, g^2, g^3, \ldots, g^{p-2}\} = (Z_p)^\star$, g is a generator of $(Z_p)^\star$
  - Not every element is a generator.

- Order
  - For $g \in (Z_p)^\star$, $\{1, g, g^2, g^3, \ldots\}$ is called the group generated by g, denoted $< g >$
  - The order of $g \in (Z_p)^\star$ is the size of $< g >$
    - $ord_p < g > = | < g > | =$ smallest a > 0 such that $g^a = 1$ in Z_p

Theorem

- $\forall g \in (Z_p)^\star : ord_p(g)$ divides $p - 1$

- For an integer N define $\phi(N) = |(Z_N)^\star|$
  - $\phi(p) = p - 1$
  - For $N = p \cdot q$, $\phi(N) = N - p - q + 1 = (p - 1)(q - 1)$

Theorem

- $\forall x \in (Z_N)^\star : x^{\phi(N)} = 1$ in Z_N
  - Generalization of Fermat, and it is the basis of the RSA cryptosystem.

Modular e'th Root

- Let p be a prime and $c \in Z_p$
  - $x \in Z_p \ s.t. x^e = c$ in Z_p is called the e'th root of c.
    - Not all e'th root exists
- Suppose $gcd(e, p - 1) = 1$, then for all c in $(Z_p)^\star$, $c^{\frac{1}{e}}$ exists in Z_p and is easy to find.
- If p is an odd prime, then $gcd(2, p - 1) \neq 1$
  - $in Z_p^\star$, $x \rightarrow x^2$ is a 2-to-1 function.
  - x in Z_p is a quadratic residue if it has a square root in Z_p
    - If p is an odd prime, then the number of quadratic residue in Z_p is $\frac{p-1}{2} + 1$

Theorem

- $x \in Z_p^*$ is a quadratic residue ⇔ $x^{\frac{p-1}{2}} =$ in Z_p
  - p is an odd prime.
  - $x \neq 0 \Rightarrow x^{\frac{p-1}{2}} = 1^{\frac{1}{2}} \in \{1, -1\}$ in Z_p
  - $x^{\frac{p-1}{2}}$ is called the legendre symbol of x over p.

Lemma

- If $c \in Z_p^*$ is a quadratic residue, then $\sqrt{c} = c^{\frac{p+1}{4}}$ in Z_p
  - $p \equiv 3 \mod 4$
- If $p \equiv 1 \mod 4$, then takes longer $O(\log^3 p)$

- Solving Quadratic Equations mod p
  - $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ in Z_p
  - Find $(2a^{-1})$ using Extended Euclidean Algorithm
  - Find square root of $b^2 - 4ac$ using a square root algorithm
- Let N be a composite number and $e > 1$
  - Requires factorization of N to
    - See if $c^{\frac{1}{e}}$ in Z_N exists.
    - See if we can compute if efficiently

Arithmetic Algorithms

- Representing an n-bit integer on a 64-bit machine.
  - Some processors have 128-bit registers and support multiplication on them.
  - $\frac{n}{32}$ blocks
- Addition and Subtraction : $O(n)$
- Multiplication : Naively $O(n^2)$ but can be faster
  - Best $O(n \log n)$
- Division with remainder $O(n^2)$
- Exponentiation

- In a finite cyclic group G, given g in G and x to compute g^x
- The repeated squaring algorithm
  - Given g in G and x > 0, Output g^x
  - Turn x into binary
  - $y \leftarrow g, z \leftarrow 1$
  - Do n times the steps : if $x[i] == 1$, then $z \leftarrow yz$, also, $y = y^2$
  - output z
- Less than $O(n^3)$

Intractable Problems

- Given composite in N and x in Z_N, find x^{-1} in Z_N
  - Easy
- Given prime p and polynomial $f(x)$ in $Z_p[x]$, find x in Z_p such that $f(x) = 0$ in Z_p
  - Easy, running time is linear in deg(f)
- Find a prime p > 2 and $g \in (Z_p)^*$ with order q
  - Consider the function $x \rightarrow g^x$ in Z_p
  - Consider the inverse $D\log_g(g^x) = x$ where x is 0 to q - 1
- DLOG
  - Let G be a finite cyclic group and g be a generator of G, q is called the order of G
  - We say that DLOG is hard in G if for all efficient algorithm A : $Pr_{g \leftarrow G, x \leftarrow Z_q}[A(G, q, g, g^x) = x] <$ negligible
    - Z_p for large p
    - Elliptic curve groups mod p
- An application : Collision Resistance
  - Choose a group G where DLOG is hard. Let $q = |G|$ be a prime, choose generators g, h of G. For $x, y \in \{1, \dots, q\}$, define $H(x, y) = g^x \cdot h^y$ in G.

Lemma

- Finding collision for H is as hard as computing $D\log_g(h)$

- Consider the set of integers $Z_2(n) := \{N = pq \text{ where p,q are n-bit primes}\}$
  - Problem 1 : Factor a random N in the set
  - Problem 2 : Given a polynomial f(x) where its degree > 1 and a random N, find the root in Z_N.

- The factoring problem
  - Best known algorithm : NFS in $\exp(\tilde{O}(\sqrt[3]{n}))$

Public Key Encryption from Trapdoor Permutations

Public Key Encryption : Definitions and Security

- Bob generates (PK, SK) and gives PK to Alice
- Session Setup(Only eavesdropping security)
  - Non-interactive Applications(Email)
    - Bob sends email to Alice encrypted using $pkAlice$
    - Bob needs pk_{Alice} : Key management

- Relations to Symmetric Cipher Security
  - One-time Security and Many-time Security(CPA)
  - One-time Security cannot imply CPA
    - The attacker can encrypt himself
  - Public key encryption must be randomized
- Active Attacks : Symmetric Cipher v.s Public Key Encryption
  - Secure Symmetric Cipher provides authenticated encryption
    - Attacker cannot create new ciphertexts
    - CCA secure
  - Public Key settings
    - Attacker can create new ciphertext
    - Directly require CCA

Constructions

- Goal : Construct CCA secure Public Key Encryption
- Trapdoor functions(TDF)
  - A trapdoor function $X \rightarrow Y$ is a triple of algorithms.
    - G() : Randomized algorithm outputs a key pair (pk, sk)
    - F(pk, .) : Deterministic algorithm that defines a function $X \rightarrow Y$

- $F^{-1}(sk, .)$: Defines a function $Y \to X$ that inverts F(pk, .)
  - $\forall(pk, sk)$ output by G, $\forall x \in X: F^{-1}(sk, F(pk, x)) = x$
  - (G,F,F^{-1}) is secure if F(pk, .) is a one-way function.
    - It can be evaluated, but cannot be inverted without sk.
    - $Adv_{OF}[A, F] = Pr[x = x'] <$ negligible
  - Public Key Encryption from TDFs
    - (G, F, F^{-1}) : secure TDF $X \to Y$
    - (E_s, D_s) : Symmetric authenticated encryption defined over (K, M, C)
    - $H : X \to K$: a hash function
    - The Key generation algorithms are the same.
  - Header : $F(pk, x)$
  - Body : $E_s(H(x), m)$

Theorem

- If (G, F, F^{-1}) is a secure TDF, (E_s, D_s) provides authenticated encryption and $H : X \to K$ is a random oracle. Then (G, E, D) is CCA^{ro} secure.
  - No directly encrypt F to a plaintext
    - Deterministic : Cannot be semantically secure
    - Other attacks

The RSA Trapdoor Permutation

- SSL / TLS : Certificates and Key Exchange
- Secure E-mail and File Systems
- G : Chooses random primes p,q about 1024 bits, set N=pq. Choose integers e,d such that $ed = 1(\mathrm{mod}\,\phi(N))$
  - Output pk=(N,e), sk = (N,d)
  - F(pk, x) : $Z_N^* \to Z_N^*$
  - $RSA(x) = x^e$ in Z_N
  - $F^{-1}(sk, y) = y^d$, $y^d = RSA(x)^d = x^{ed} = x^{k\phi(N)+1} = x$
- RSA Assumption : RSA is a one-way permutation
  - For all efficient algorithms A :

        - $Pr[A(N,e,y) = y^{\frac{1}{e}}] < negligible$

- The RSA trapdoor permutation is not a encryption scheme.
    - Suppose a random-session key k is 64 bits from 0 to 2^{64}. Eve sees $c = k^e$ in Z_N
    - If $k = k_1 \cdot k_2$ where $k_1, k_2 < 2^{34}$, then $\frac{c}{k_1^e} = k_2^e$ in Z_N
        - Probability is about 20 %.
    - Step 1 : Build a table $\frac{c}{1^e} \sim \frac{c}{2^{34e}}$
    - Step 2 : for k_2 from 0 to 2^{34}, test if k_2^e is in the table.
    - Output matching and the total attack time about $2^{40} << 2^{64}$

PKCS1

- Message Key $\xrightarrow{Preprocessing} \xrightarrow{RSA}$ Ciphertext
    - Preprocess
    - Security about the resulting system
- PKCS1 v1.5
    - Mode 2
    - 02(16bits) + random pad + FF + msg (RSA modulus size)(2048 bits)
      + Resulting value is RSA encrypted
      + Widely deployed ex : in HTTPS
    - Bleichenbacher attack
        - Attacker can test if 16 MSBs of plaintext = '02'
        - CCA to decrypt a given ciphertext C do :
            - Choose $r \in Z_N$. Compute $c' \leftarrow r^e \cdot c = (r \cdot PKCS1(m))^e$
            - Send c' to web server and use response
    - Baby Bleichenbacher
        - Suppose $N = 2^n$(an invalid RSA modulus)
        - Sending c reveals msb(x)
        - Sending $2^e c$ reveals msb(2x mod N)=$msb_2(x)$
        - and so on...
    - HTTP Defense
        - Generate a string R of 46 random bytes
        - Decrypt the message to recover the plaintext M
        - If the PKCS1 Padding is not correct, pre_master_secret = R

- Preprocessing Function : OAEP (Optimal Asymmetric Encryption Padding)
  - Check the pad on decryption, reject if the ciphertext is invalid.

Theorem

- RSA is a trap-door permutation $\Rightarrow$ RSA-OAEP is CCA secure when H,G are random oracles.
  - Use SHA-256 for H and G.

- OAEP+
  - During encryption, validate W(m,r) field.
  - Add a new random oracle W.
  - Problems : Timing information leaks type of error, then the attacker can decrypt any ciphertext.
- SAEP+
  - RSA(e=3)
  - Delete G, add W.

Is RSA a One-Way Function?

- To invert without d, the attakcer must compute x from $c = x^e$ (mod N)
  - Factor N(hard)
  - Compute e'th roots modulo p and q(easy)
  - Use reduction to show that there is no shortcut.
  - To speed up RSA decryption, use small private key d(2^{128}), but it will be insecure.

    Private key d can be found from (N, e)
- Wiener's attack
  - $e \cdot d = k \cdot \phi(N) + 1 \Rightarrow |\frac{e}{\phi(N)} - \frac{k}{d}| = \frac{1}{d \cdot \phi(N)} \leq \frac{1}{\sqrt{N}}$
  - If $d \leq \frac{N^{0.25}}{3}$, the attacker can use continued fraction expansion to find the private key d.
  - $e \cdot d = 1(\mathrm{mod}\ k) \Rightarrow \gcd(d, k) = 1 \Rightarrow find\ d\ from\ \frac{k}{d}$

RSA in Practice

- To speed up encryption, use a small e : e=3(min)
  - Recommended value : e = 65537(17 multiplications)
- Fast encryption / Slow decryption
- Attacks

  - Timing Attack : The time it takes to compute $c^d (\bmod N)$ can expose d.
  - Power Attack : The power consumption of a smartcard while it is computing c^d (mod N) can expose d.
  - Faults Attack : A computer error can expose d.
    - Fix one, make one incorrect, then $\gcd((x')^e - c, N) = p$
    - Solution : Check output, but 10% slowdown.
- RSA Key Generation Trouble
  - Suppose poor entropy at start.
  - Same p will be generated by multiple devices, but different q.
  - $\gcd(N_1, N_2) = p$

Public Key Encryption from Diffie-Hellman

The ElGamal Public-Key System

- Public Key Encryption Applications
  - Key Exchange (HTTPS)
  - Encryption in non-interactive settings
    - Secure Email : Bob has Alice's public key and sends her email.
    - Encrypted file systems
    - Key escrow : Data recovery without the key of Bob.
- Constructions
  - Based on trapdoor functions(RSA)
  - Based on Diffie-Hellman Protocol
    - ElGamal encryption
  - Goals : Chosen Ciphertext Security
- ElGamal
  - Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order n
  - Fix a generator g in G
  - Alice : Choose random a from 1 to n
    - Treat $A = g^a$ as a public key.
  - Bob : Choose random b from 1 to n
  - Compute $g^{ab} = A^b$, derive symmetric key k, encrypt message m with k
  - Decryption
    - Compute $g^{ab} = B^a$, derive k, and decrypt

- The ElGamal System
  - G : Finite cyclic group of order n
  - (E_s, D_s) : Symmetric authenticated encryption defined over (K, M, C)
  - $H : G^2 \rightarrow K$: a hash function
  - Key generation
    - Choose random generator g in G and random a in Z_n
    - Output sk = a, pk = $(g, h = g^a)$
  - Performance
    - Encryption : 2 exponent
    - Can do pre-compute with fixed-basis
    - Speed up 3x
    - Decryption : 1 exponent with variable basis

ElGamal Security

- Computational Diffie-Hellman Assumption
  - CDH assumption holds in G if g, g^a, g^b cannot compute g^{ab} easily
- Hash Diffie-Hellman Assumption
  - HDH assumption holds for (G, H) if $H(g^b, g^{ab})$ acts as R
  - H acts as an extractor : Strange distribution on $G^2 \Rightarrow$ uniform on K
  - ElGamal is semantically secure under HDH
  - To prove CCA security, it needs stronger assumption.
- Interactive Diffie-Hellman in group G(IDH)
  - The attacker can request the oracle.

Security Theorem

- If IDH holds in group G, (E_s, D_s) provides authenticated encryption and $H : G^2 \rightarrow K$ is a random oracle, then $ElGamal$ is CCA^{ro} secure.

ElGamal Variants with Better Security

- Prove CCA based on CDH
  - Use group G where CDH=IDH(bilinear group)
  - Change the ElGamal System
- Twin ElGamal

- $a_1, a_2 \leftarrow Z_n$
- output $pk = (g, h_1 = g^{a1}, h_2 = g^{a2})$
- $sk = (a_1, a_2)$

Security Theorem

- If CDH holds in group G, (E_s, D_s) provides authenticated encryption and $H : G^{ˇ} \to K$ is a random oracle, then twin $ElGamal$ is CCA^{ro} secure.
  - One more exponentiation in Encryption and decryption
    - No one knows if it is worth it.
- Prove CCA security without random oracles
  - Use HDH assumption in bilinear groups
    - Special elliptic curve with more structures
  - Use DDH assumption in any group

A Unifying Theme

- Generic One-way Functions
  - Let $f : X \to Y$ be a secure PRG where (|Y|>>|X|)
    - f is built using deterministic counter mode
    - No special properties, hard to use for key exchange

Lemma

- f is a secure PRG ⇒ f is one-way

- The DLOG one-way function
  - Fix a finite cyclic group G of order n
  - g : a random generator in G
  - Define $f : Z_n \to G$ as $f(x) = g^x \in G$

Lemma

- If DLOG is hard in G, then G is one-way
  - Use exponent property $f(x + y) = f(x)f(y)$
  - Key Exchange and Public-key encryption

- RSA one-way function

- $f(xy) = f(x)f(y)$ and f has a trapdoor
- Conclusion
  - PKE is made possible by one-way functions with special properties.
    - Homomorphic properties and trapdoor