

Week 1 Notes

Uday Manchanda

October 6, 2019

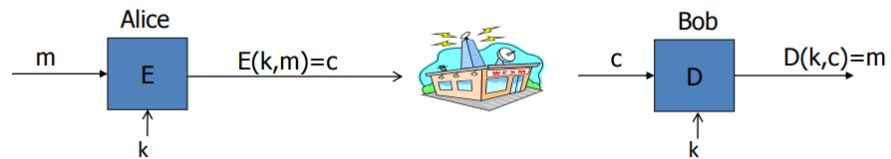
Abstract

Week 1 Notes in the Coursera Course on Cryptography

1 What is Cryptography About?

1.1 Course Overview

- Secure communication: HTTPS, WPA2, GSM, Bluetooth, SSL/TLS
- SSL/TLS has two main parts:
 - Handshake Protocol
 - Record Layer
- Encrypting files on disk: EFS, TrueCrypt
- Symmetric Encryption



– Where:

- * E, D: cipher
- * k : secret key (ex: 128 bits)
- * m: plaintext
- * c: ciphertext

– Encryption Algorithm is **publicly known**

- Use Cases
 - Single use key (one time key)

- * key is used to encrypt a single message
- * new key generated for each email
- Multi use key (many time key)
 - * key encrypts multiple messages
 - * same key used to encrypt many files

1.2 What is Cryptography

- Confidentiality and integrity
- Digital signatures, anonymous communication, digital cash
- If something can be done with a trusted authority, it can also be done without one
- Three steps in cryptography
 - Precisely specify threat model
 - Propose a construction
 - Prove that breaking construction under threat mode will solve an underlying hard problem

1.3 History

- Substitution cipher
 - Find most common letter ("E") via frequencies
 - Use frequency of pairs of letters
- Caesar Cipher
 - Shift letters by three
 - Size of key space: $|\kappa| = 26!$
- Vigenere Cipher
 - Encrypt message m with some cipher k
 - For each letter in m , add it to corresponding letter in k
 - k will "wrap around" until end of message
 - Take added result and mod 26 to obtain result between 0 and 25
- Rotor Machines
- Data Encryption Standard (DES)
 - keys: 2^{56} , block size: 64 bits
 - today AES is in use

2 Discrete Probability

2.1 Crash Course Part I

- Let U be some finite set, eg: $U = \{0, 1\}^n$
- Probability distribution P over U is a function $P : U \rightarrow [0, 1]$, such that $\sum_{x \in U} P(x) = 1$
- EX:
 - Uniform Distribution: for all $x \in U : P(x) = \frac{1}{|U|}$
 - Point Distribution at $x_0 : P(x_0) = 1, \forall x \neq x_0 : P(x) = 0$
- Distribution vector: $(P(000), P(001), P(010), \dots, P(111))$
- Events
 - For a set $A \subseteq U : Pr[A] = \sum_{x \in U} P(x) \in [0, 1]$
 - Set A is an **event**
 - EX: $U = \{0, 1\}^8$
 - $A = \{ \text{all } x \text{ in } U \text{ such that } lsb_2(x) = 11 \} \subseteq U$ for the uniform distribution on $0, 1^8 : Pr[A] = \frac{1}{4}$
- The Union Bound
 - For events A_1 and $A_2 : Pr[A_1 \cup A_2] \leq Pr[A_1] + Pr[A_2]$
 - $A_1 \cap A_2 = \emptyset \rightarrow Pr[A_1 \cup A_2] = Pr[A_1] + Pr[A_2]$
 - * $A_1 = \{ \text{all } x \text{ in } 0, 1^n \text{ s.t } lsb_2(x) = 11 \}$
 - * $Pr[lsb_2(x) = 11 \text{ or } msb_2(x) = 11] = Pr[A_1 \cup A_2] \leq \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$
- Random Variables
 - Definition: a random variable X is a function: $X : U \rightarrow V$
 - EX: $X : \{0, 1\}^n \rightarrow \{0, 1\}; X(y) = lsb(y) \in \{0, 1\}$
 - For the uniform distribution on $U : Pr[X = 0] = \frac{1}{2}, Pr[X = 1] = \frac{1}{2}$
 - more generally: random variable X induces a distribution on V : $Pr[X = v] := Pr[X^{-1}(v)]$
- Uniform random variable
 - Let U be some set, eg $U = \{0, 1\}^n$
 - We write $r \leftarrow U$ to denote a **uniform random 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 $x \in U$
 - Let r be a uniform random variable on $\{0, 1\}^2$

- Define random variable $X = r_1 + r_2$
- Then $Pr[X = 2] = \frac{1}{4}$
- Hint: $Pr[X = 2] = Pr[r = 11]$
- Randomized Algorithms
 - Deterministic algorithm: $y \leftarrow A(m)$
 - Randomized algorithm: $y \leftarrow A(m; r)$ where $r \leftarrow \{0, 1\}^n$
 - EX: $A(m; k) = E(k, m), y \leftarrow^n A(m)$

2.2 Crash Course Part II

- Independence
 - Events A and B are **independent** if $Pr[A \text{ and } B] = Pr[A] \times Pr[B]$
 - EX: $U = \{0, 1\}^2 = \{00, 01, 10, 11\}$ and $r \leftarrow U$
 - define r.v X and Y as: $X = \text{lsb}(r), Y = \text{msb}(r)$
 - $Pr[X = 0 \text{ and } Y = 0] = Pr[r = 00] = \frac{1}{4} = Pr[X = 0] \times Pr[Y = 0]$
- XOR
 - XOR of two strings in $\{0, 1\}^n$ is their bit-wise addition mod 2
 - XOR Chart

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

* (Yeah I made that)

- EX: $0110111 \oplus 1011010 = 1101101$
- Important property of XOR
 - Theorem: Y a random variable over $\{0, 1\}^n$, X is an independent uniform variable on $\{0, 1\}^n$
 - Then $Z := Y \oplus X$ is a uniform variable on $\{0, 1\}^n$
 - Proof for $n = 1$
 - * $Pr[Z = 0] = Pr[(x, y) = (0, 0) \text{ or } (x, y) = (1, 1)]$
 - * $Pr[(x, y) = (0, 0)] + Pr[(x, y) = (1, 1)] = \frac{P_0}{2} + \frac{P_1}{2} = \frac{1}{2}$

- The Birthday Paradox
 - Let $r_1, \dots, r_n \in U$ be independent identically distributed random variables
 - When $n = 1.2 \times |U|^{1/2}$ then $Pr[\exists i \neq j : r_i = r_j] \geq 1/2$
 - notation: $|U|$ is the size of U
 - EX: Let $U = \{0, 1\}^{128}$
 - * After sampling about 2^{64} random messages from U , some two sampled messages will likely be the same

3 Stream Ciphers

3.1 One Time Pad

- Symmetric Ciphers
 - Def: a cipher defined over (K, M, C) is a pair of "efficient" algorithms (E, D) , where:
 - * $E = K \times M \rightarrow C$
 - * $D = K \times C \rightarrow M$
 - * $\forall m \in M, k \in K : D(k, E(k, m)) = m$
 - E is often randomized, D is always deterministic
- One Time Pad
 - First example of a "secure" cipher
 - $M = C = \{0, 1\}^n, K = \{0, 1\}^n$
 - key = (random bit string as long as the message)
 - $C := E(k, m) = k \oplus m$
 - $D(k, c) = k \oplus c$
 - $D(k, E(k, m)) = D(k, k \oplus m) = k \oplus (k \oplus m) = (k \oplus k) \oplus m = 0 \oplus m = m$
 - You are given a message and its OTP encryption (c). Can you compute the OTP key from m and c ?
 - * Yes, the key is: $k = m \oplus c$
 - * Very fast encryption/decryption...but long keys (as long as plaintext)
- What is a secure cipher
 - Attacker's abilities: CT only attack (for now)
 - Possible security attempts
 - * 1. Attacker cannot recover secret key. $E(k, m) = m$ would be secure

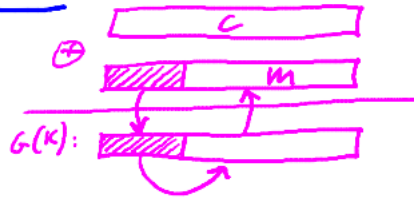
- * 2. Attacker cannot recover all of the plaintext. $E(k, m_0 || m_1) = m_0) || k \oplus m_1$ would be secure
- * Shannon's idea: CT should reveal no "info" about the PT
- Information Theoretic security
 - Definition: A cipher (E, D) over (K, M, C) has **perfect secrecy** if:
 - * $\forall m_0, m_1 \in M, (len(m_0) = len(m_1)), \forall c \in C$
 - * $Pr[E(K, m_0) = c] = Pr[E(k, m_1) = c]$
 - * where k is uniform in K ($k \leftarrow K$)
 - * Given ciphertext can't tell if message is m_0 or m_1 (for all m_0, m_1)
 - * most powerful adversary learns nothing about plaintext from ciphertext
 - * no ciphertext only attack (other attacks are possible)
 - Lemma: OTP has perfect secrecy. Proof:
 - * $\forall m, c : Pr[E(k, m) = c] = \frac{\text{number of keys } \kappa \in K \text{ s.t. } E(k, m) = c}{|K|}$
 - * If $\forall m, c : \text{number} = \{\kappa \in K : E(k, m) = c\} = \text{const}$
 - * Therefore, cipher has perfect secrecy
 - * Let $m \in M$ and $c \in C$. How many OTP keys map m to c ? **one**
 - * However: implies that: $|\kappa| \geq |M| \rightarrow$ hard to use in practice, (key length > message length)

3.2 Pseudorandom Generators

- Stream Ciphers: Making OTP practical
 - idea: replace "random" key by "pseudorandom" key
 - PRG is a function: $G : \{0, 1\}^s \rightarrow \{0, 1\}^n$
 - $n \gg s$
 - computable by a deterministic algorithm
 - $C := E(k, m) = m \oplus G(k)$
 - $D(k, c) = C \oplus G(k)$
 - Can a stream cipher have perfect secrecy: No since the key is shorter than the message
 - Stream ciphers cannot have perfect secrecy
 - * Need a different definition of security
 - * Security will depend on specific PRG
- PRG must be unpredictable
 - Suppose PRG is predictable:

- $\exists i : G(k)|_{1,\dots,i} \rightarrow^{alg} G(k)|_{i+1,\dots,n}$

Then:



- We say that $G : K \rightarrow \{0,1\}^n$ is **unpredictable** if:
 - * \exists "eff" alg A and $\exists_0 \leq i \leq n-1 \leq t$
 - * $Pr[A(G(k))|_{1,\dots,i} = G(k)|_{i+1}] > \frac{1}{2} + \epsilon$
 - * For non-negligible ϵ , eg $\frac{1}{2^{30}}$
 - * Definition: PRG is unpredictable if it is not predictable, no "efficient" adv. can predict bit (i+1) for "non-neg" ϵ
 - * Suppose $G : K \rightarrow \{0,1\}^n$ is such that for all k: $XOR(G(k)) = 1$. Is G predictable? \rightarrow Yes given the first (n-1) bits I can predict the n'th bit

- Weak PRGs (do not use for crypto!)

- glibc random():
- $r[i] \leftarrow (r[i-3] + r[i-31]) \% 2^{32}$
- output $r[i] >> 1$
- Never use random() for crypto

3.3 Negligible vs. non-negligible

- Negligible vs. non-negligible

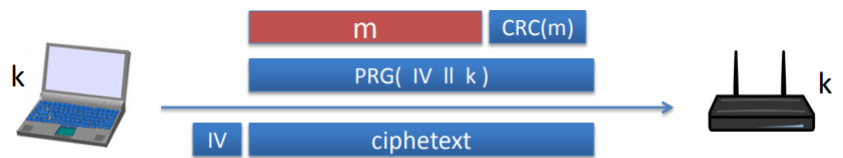
- In practice: ϵ is a scalar and
 - * ϵ non-neg $\epsilon \geq \frac{1}{2^{30}}$ (likely to happen over 1 GB of data)
 - * ϵ negligible $\epsilon \leq \frac{1}{2^{80}}$ (won't happen over life of key)
- In theory: ϵ is a function: $\epsilon : Z^{\geq 0} \rightarrow R^{\geq 0}$
 - * ϵ non-neg $\exists d : \epsilon(\lambda) \geq \frac{1}{\lambda^d} (\epsilon \geq \frac{1}{poly}, \text{ for many } \lambda)$
 - * ϵ negligible $\forall d, \lambda \geq \lambda_d : \epsilon(\lambda) \leq \frac{1}{\lambda^d} (\epsilon \leq \frac{1}{poly}, \text{ for large } \lambda)$
- Few Examples
 - * $\epsilon(\lambda) = \frac{1}{2^\lambda}$: negligible
 - * $\epsilon(\lambda) = \frac{1}{\lambda^{1000}}$: non-negligible
 - *

$$\begin{cases} \frac{1}{2^\lambda} & \text{for odd } \lambda \\ \frac{1}{\lambda^{1000}} & \text{for even } \lambda \end{cases}$$

- PRGs: the rigorous theory view
 - PRGs are parametrized by a security parameter: λ
 - * PRG becomes "more secure" as λ increases
 - Seed lengths and output lengths grow with λ
 - For every $\lambda = 1, 2, 3, \dots$ there is a different PRG: G_λ
 - $G_\lambda : K_\lambda \rightarrow \{0, 1\}^{n(\lambda)}$
 - We say that the previous equation is **predictable** at position i if:
 - * there exists a polynomial time (in λ) algorithm A s.t.
 - * $\Pr_{k \leftarrow K_\lambda}[A(\lambda, G_\lambda(k))|_{1, \dots, i} = G_\lambda(k)|_{i+1}] > \frac{1}{2} + \epsilon(\lambda)$
 - * For some non-negligible function $\epsilon(\lambda)$

3.4 Attacks on OTP and Stream Ciphers

- Attack 1: Two time pad is insecure
 - Never use stream cipher key more than once
 - $C_1 \leftarrow m_1 \oplus PRG(k)$
 - $C_2 \leftarrow m_2 \oplus PRG(k)$
 - Eavesdropper does: $C_1 + C_2 \rightarrow m_1 \oplus m_2$
 - Enough redundancy in English and ASCII encoding that: $m_1 \oplus m_2 \rightarrow m_1, m_2$
 - Real World Examples
 - * Project Venona
 - * MS-PPTP
 - Have one key for interaction between server and client
 - Another one between client and server
 - * 802.11b WEP



- Length of IV: 24 bits
- Repeated IV after $2^{24} \approx 16$ M frames
- On some 802.11 cards: IV resets to 0 after power cycle
- Avoid related keys
- * Disk encryption (one-time pad does not work)

- As you make changes to a file, the encrypted contents can leak as a hacker can figure out where in memory those changes were made
- Attack 2: OTP provides no integrity, is malleable
 - $(m \oplus k) \oplus p$ (p is the message used by the hacker)
 - Decrypting the previous expression will yield $m \oplus p$
 - OTP is malleable because it is easy to change ciphertext

3.5 Real World Stream Ciphers

- Old Example 1: RC4
 - Input: variable sized seed
 - Expands into 2048 bits, executes very simple loop to generate one byte/round
 - Used in HTTPS and WEP
 - Weaknesses: bias in initial output, probability of (0,0) is $\frac{1}{256^2} + \frac{1}{256^3}$, Related key attacks
- Old Example 2: CSS(Content Scrambling System)
 - Badly broken
 - Linear feedback shift register
 - * a register that contains cells
 - * certain taps are in certain cells that feed into an XOR
 - * at every clock cycle register shifts to the left
 - DVD encryption uses this
 - seed = 5 bytes
- Modern Example: eStream
 - PRG: $\{0,1\}^s * R \rightarrow \{0,1\}^n$
 - Nonce: a non-repeating value for a given key
 - First input is a seed, R is the nonce
 - $E(k, m : r) = m \oplus PRG(k; r)$
 - the pair k, r is never used more than once
 - can reuse the key because (k, r) is unique
 - eStream: Salsa20 (SW + HW)
 - * 128,256 bit seed
 - * 64 bit nonce
 - * $\{0,1\}^{128 \text{ or } 256} * \{0,1\}^{64} \rightarrow \{0,1\}^n$
 - * $\text{Salsa20}(k;r) := H(k, (r, 0)) || H(k, (r, 1)) || \dots$
 - * Expand the states into 64 bytes long

3.6 PRG Security Defs

- Let $G : K \rightarrow \{0, 1\}^n$ be a PRG
- Goal: define what it means that: $[K \leftarrow \kappa, \text{output } G(k)]$
- This distribution of pseudo random string is "indistinguishable" from a truly uniform distribution
- $[r \leftarrow \{0, 1\}^n, \text{output } r]$
- Statistical Test
 - Statistical test on $\{0, 1\}^n$: an algorithm A s.t. $A(x)$ outputs 0 or 1
 - Takes input x and decides whether its random or not
 - Examples:
 - * $A(x) = 1$ iff $|\#0(x) - \#1(x)| \leq 10 * \sqrt{n}$
 - * $A(x) = 1$ iff $|\#00(x) - \frac{n}{4}| \leq 10 * \sqrt{n}$
 - * $A(x) = 1$ iff $\max_num_of_0(x) \leq 10 * \sqrt{n}$
- Advantage: $Adv_{PRG}[A, G] := |Pr[A(G(k)) = 1] - Pr[A(r) = 1]| \leftarrow [0, 1]$
- If the advantage is close to 1, it means that the statistical test behaved differently when we gave it pseudorandom inputs versus random inputs
- If the advantage is close to 0, it means the pseudorandom inputs and random inputs behaved the same (or similarly)
- Suppose $G : K \rightarrow \{0, 1\}^n$ satisfies $msb(G(k)) = 1$ for 2/3rds of the keys in K
 - Define statistical test $A(x)$ as: *if*[msb = 1 output 1 else output 0]
 - Then: $Adv_{PRG} = |Pr[A(G(k)) = 1] - Pr[A(r) = 1]| = \frac{2}{3} - \frac{1}{2} = \frac{1}{6}$
 - "A breaks G with advantage 1/6"
- Secure PRGs: Crypto definition
 - Def: We say that $G : K \rightarrow \{0, 1\}^n$ is a **secure PRG** if