Tutorial 1 Crypto: Symmetric Encryption, Math Prelim

March 6, 2018

Symmetric Ciphers

Symmetric Encryption: Alice and Bob have a pre-shared key, k. The goal is to ensure confidentiality over the channel between Alice and Bob : the attacker can only 'eavesdrop' (no tampering).

- A cipher is defined over the triple (K, M, C)
  - K the set of all possible keys "key space"
  - M the set of all possible messages "message space"
  - C the set of all possible ciphertexts
- A cipher is a pair of efficient encryption and decryption algorithms (E,D) such that D(E(k,m)) = m (also called the *consistency equation*), where $E : K \times M \to C$ and $D : K \times C \to M$
- Notation: CT - ciphertext, PT - plaintext

One-Time Pad (Vernam 1917)

- A simple example of a cipher where $E = k \oplus m$ and $D = k \oplus c$

One-Time Pad (Vernam 1917)

- A simple example of a cipher where $E = k \oplus m$ and $D = k \oplus c$
- Does it satisfy the consistency equation?

One-Time Pad (Vernam 1917)

- A simple example of a cipher where $E = k \oplus m$ and $D = k \oplus c$
- Does it satisfy the consistency equation?
- Yes!
  $D(E(k, m)) = k \oplus c = k \oplus (k \oplus m) = (k \oplus k) \oplus m = 0 \oplus m = m$

One-Time Pad (Vernam 1917)

- A simple example of a cipher where $E = k \oplus m$ and $D = k \oplus c$
- Does it satisfy the consistency equation?
- Yes!
  $D(E(k, m)) = k \oplus c = k \oplus (k \oplus m) = (k \oplus k) \oplus m = 0 \oplus m = m$
- Is OTP a **good** cipher?

Perfect Secrecy (Shannon 1949)

- Ideally, we would like the attacker to not be able to learn any information about PT from CT

Perfect Secrecy (Shannon 1949)

- Ideally, we would like the attacker to not be able to learn any information about PT from CT
- A cipher (E,D) has perfect secrecy if for every 2 messages with the same length the probability that a ciphertext c is encryption for m_0 with key k is the same as the probability that the ciphertext is an encryption of m_1 with key k.

Perfect Secrecy (Shannon 1949)

- Ideally, we would like the attacker to not be able to learn any information about PT from CT
- A cipher (E,D) has perfect secrecy if for every 2 messages with the same length the probability that a ciphertext c is encryption for m_0 with key k is the same as the probability that the ciphertext is an encryption of m_1 with key k.
- Turns out, OTP has **perfect secrecy**.

Lemma: OTP has perfect secrecy

Proof:
$\forall m, c \; Pr[E(k,m) = c] = \frac{\#keys \quad k \in K \quad E(k,m)=c}{|K|}$. If
$\#keys \quad k \in K$ such that $E(k,m) = c$ is a constant then the cipher has perfect secrecy.
$c = m \oplus k$ 1 key
For OTP if E(k,m) = c then
$k \oplus m = c \implies k = m \oplus c \implies \#keys \quad k \in K$ such that
$E(k,m) = c$ is 1 $\forall m, c$

OTP: The downside

However, Shannon proved that in order for a cipher to have perfect secrecy $|K| \geq |M|$ which makes it very difficult to use in practice.

How to make OTP practical?

We introduce *pseudorandom generators* (PRG).

- Idea: replace the random key with a pseudorandom key. From a small key deterministically generate a much larger key that appears to be random

How to make OTP practical?

We introduce *pseudorandom generators* (PRG).

- Idea: replace the random key with a pseudorandom key. From a small key deterministically generate a much larger key that appears to be random
  - $G : \{0,1\}^s \rightarrow \{0,1\}^l, l >> s$

How to make OTP practical?

We introduce *pseudorandom generators* (PRG).

- Idea: replace the random key with a pseudorandom key. From a small key deterministically generate a much larger key that appears to be random
  - $G : \{0,1\}^s \rightarrow \{0,1\}^l, l >> s$
- A PRG is secure if observing up to n bits of the output an attacker cannot predict the next bit.

How to make OTP practical?

We introduce *pseudorandom generators* (PRG).

- Idea: replace the random key with a pseudorandom key. From a small key deterministically generate a much larger key that appears to be random
  - $G : \{0,1\}^s \rightarrow \{0,1\}^l, l >> s$
- A PRG is secure if observing up to n bits of the output an attacker cannot predict the next bit.
- Is XOR(G(k)) = 1 a PRG?

How to make OTP practical?

We introduce *pseudorandom generators* (PRG).

- Idea: replace the random key with a pseudorandom key. From a small key deterministically generate a much larger key that appears to be random
    - $G : \{0,1\}^s \rightarrow \{0,1\}^l, l >> s$
- A PRG is secure if observing up to n bits of the output an attacker cannot predict the next bit.
- Is XOR(G(k)) = 1 a PRG?
- No, given n bits I can predict the n+1 bit such that XOR = 1

Stream Ciphers

- We can construct a **stream cipher** from a PRG:
  - $E(k, m) = G(k) \oplus m$
  - $D(k, c) = G(k) \oplus c$
- If we have a secure PRG then the stream cipher built with it is also secure!
- The existence of provably secure PRGs implies $P \neq NP$
- The goal of a PRG is that it is indistinguishable from random $\rightarrow$ statistical test

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$
  - $c_2 \leftarrow m_2 \oplus PRG(k)$

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$
  - $c_2 \leftarrow m_2 \oplus PRG(k)$
- Attacker obtained c_1, c_2, what can he do?

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$
  - $c_2 \leftarrow m_2 \oplus PRG(k)$
- Attacker obtained c_1, c_2, what can he do?
- $c_1 \oplus c_2 = (m_1 \oplus PRG(k)) \oplus (m_2 \oplus PRG(k)) = m_1 \oplus m_2$

Two-time pad is insecure

- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$
  - $c_2 \leftarrow m_2 \oplus PRG(k)$
- Attacker obtained c_1, c_2, what can he do?
- $c_1 \oplus c_2 = (m_1 \oplus PRG(k)) \oplus (m_2 \oplus PRG(k)) = m_1 \oplus m_2$
- there's enough redundancy in English to brute force and find m_1, m_2

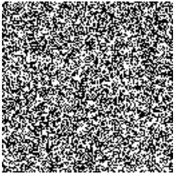
Two-time pad is insecure

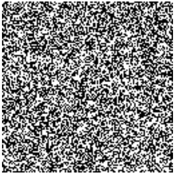
- Same pad is used to encrypt 2 messages m_1, m_2:
  - $c_1 \leftarrow m_1 \oplus PRG(k)$
  - $c_2 \leftarrow m_2 \oplus PRG(k)$
- Attacker obtained c_1, c_2, what can he do?
- $c_1 \oplus c_2 = (m_1 \oplus PRG(k)) \oplus (m_2 \oplus PRG(k)) = m_1 \oplus m_2$
- there's enough redundancy in English to brute force and find m_1, m_2
- never use stream cipher more than once!

OTP Reuse in Pics

- Message 1

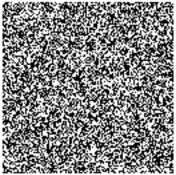
- The binary one-time pad and the resulting cipher 1:



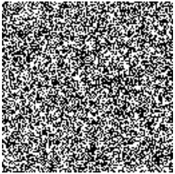


OTP Reuse in Pics

- Message 2

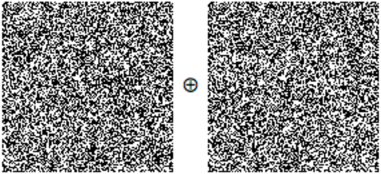
 $\oplus$ 

- Cipher 2



OTP Reuse in Pics

- Xor cipher 1 and cipher 2



- Resulting $m_1 \oplus m_2$

Redefine Security: Semantic Security

- Stream ciphers do not have perfect secrecy as defined by Shannon (key length much smaller than message length)
- For a computationally bound attacker, define semantic security (one-time key)

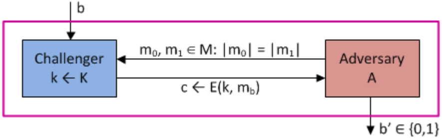
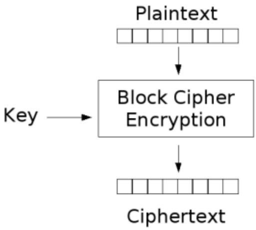


Figure: Semantic Security, Dan Boneh's course

- E is semantically secure if for all efficient adversaries $Adv_{SS}[A, E] = |PR[EXP(0) = 1] - PR[EXP(1) = 1]|$ is negligible.
- The adversary cannot distinguish between the ciphertexts corresponding to m_0 and m_1.

Block Ciphers

- Examples: DES, input block size = 64 bits, key length = 56 (broken) by exhaustive search
- 3DES, input block size = 64bits, key length = 168 bits
- AES, input block size = 128 bits, key length = 128, 192, 256 bits
- Idea: introduce PRF to reason about the security of block ciphers
- more in lecture notes...



Some Math: GCD

- For all ints (x, y) gcd(x, y) is their greatest common divisor (e.g. gcd(12, 18) = 6).
- Computing the GCD is a tractable problem - Euclid's algorithm
- Also finding (a,b) in $a \cdot x + b \cdot y = gdc(x, y)$ is efficient - extended Euclid's algorithm

Some Math: Modular Arithmetic

- $Z_N = \{0, 1, ..., N-1\}$

Some Math: Modular Arithmetic

- $Z_N = \{0, 1, ..., N - 1\}$
- Z_N denotes a ring where addition and multiplication are done modulo N

Some Math: Modular Arithmetic

- $Z_N = \{0, 1, ..., N - 1\}$
- Z_N denotes a ring where addition and multiplication are done modulo N
- Examples in Z_{12}: $9 + 8 = ?$, $5 \times 7 = ?$, 5 - 7 = ?

Some Math: Modular Arithmetic

- $Z_N = \{0, 1, ..., N - 1\}$
- Z_N denotes a ring where addition and multiplication are done modulo N
- Examples in Z_{12}: $9 + 8 = ?$, $5 \times 7 = ?$, 5 - 7 $= ?$
- $9 + 8 = 5$, $5 \times 7 = 11$, 5 - 7 $= 10$

Some Math: Groups

- Ring vs group? A ring is a group with an extra operation.
- A **group** has a single binary operation and certain properties (see lecture notes).
- Z_N^* group
  - the set of positive integers smaller than and co-prime to an integer N or the set of all invertible elements in Z_N
  - with $(\cdot)$ followed by a reduction mod N as the group operator
  - $Z_{12}^* = \{1, 5, 7, 11\}$
- Z_p^* is a cyclic group if there exists a generator g that generates all the elements of $Z_p = \{1, g, g^2, ..., g^{p-2}\}$

Some Math: Groups

- Example generators

Some Math: Groups

- Example generators
  - 3 is a generator for Z_7^* $(\{1, 3, 3^2(mod 7), 3^3(mod 7), 3^4(mod 7), ...\})$

Some Math: Groups

- Example generators
  - 3 is a generator for Z_7^* $(\{1, 3, 3^2(mod 7), 3^3(mod 7), 3^4(mod 7), ...\})$
  - 2 is not a generator for Z_7^* $(\{1, 2, 4\})$

Some Math: Groups

- Example generators
  - 3 is a generator for Z_7^* $(\{1, 3, 3^2(mod7), 3^3(mod7), 3^4(mod7), ...\})$
  - 2 is not a generator for Z_7^* $(\{1, 2, 4\})$
- The order of a group is its cardinality

Some Math: Fermat-Euler

- The inverse of x in Z_N is an element y such that $x \cdot y = 1$ in Z_N
- x in Z_N has inverse if and only if gcd(x, N) $= 1$.
- Fermat: Let p be a prime. $\forall x \in (Z_p)^*, x^{p-1} = 1$ in Z_p (p is a prime)
- Euler's generalization: For an integer N define the totient function $\phi(N) = |Z_N^*|$. We know $\forall x \in (Z_N^*) \;\; x^{\phi(N)} = 1$ in Z_N.
    - $\phi(N)$, also called Euler's totient function, is defined as the number of positive integers $\leq N$ that are relatively prime to (i.e., do not contain any factor in common with) N
    - example: $\phi(12) = |\{1, 5, 7, 11\}| = 4$

Hard Problems

Public key crypto mostly relies on the hard problems of factoring and discrete logarithms.