COMPSCI 501: Formal Language Theory

Lecture 38: Cryptography

Marius Minea
marius@cs.umass.edu

University of Massachusetts Amherst

29 April 2019

Cryptography: Fundamental Problems

Goal Today:

- fundamental cryptographic problems and primitives
- expressing them formally

- Basic Goals: confidentiality, integrity, availability
- More: authentication, signatures, timestamping, witnessing, anonymity, non-repudiation, ownership, revocation, etc.

- Communication: send message over insecure channel (intruder has access)

Symmetric Encryption

Same key used for encryption and decryption (shared by two parties)

Perfect security: **one-time pad**

$c = m \oplus k$: XOR with key of message length, never reused
pause: impractical, must somehow transmit same information quantity (key)

Short keys: vulnerable to brute-force search of key space
Or: cryptanalysis of message stream (if output not perfectly random)

Hash functions

1. preimage resistance (not invertible)

computationally infeasible to find input x' with $h(x') = y$

2. 2nd-preimage resistance

computationally infeasible to find second input mapping to same output

knowing x, find $x' \neq x$ with $h(x') = h(x)$

3. collision resistance

computationally infeasible to find distinct inputs with same output, $h(x') = h(x)$.

One-Way Functions

Easy to compute, hard to invert

One-way permutation: a permutation f such that

1. it is computable in polynomial time
2. $\Pr_{M,w}[M(f(w)) = w] \leq n^{-k}$ for every PPTM M, every k, sufficiently large n, and $w \in \Sigma^n$

One-way function = length-preserving function such that

1. it is computable in polynomial time
2. $\Pr_{M,w}[M(f(w)) = y$ with $f(y) = f(w)] \leq n^{-k}$ for every PPTM M, every k, sufficiently large n, and $w \in \Sigma^n$

We only have *some* y, not $y = w$, since f_i need not be injective. #
Candidates for One-Way Functions

Existence of one-way functions unknown.

Would imply NP $\not\subseteq$ BPP and thus P $\neq$ NP.

Public-Key Cryptosystems

Public key E (encryption key)
publicized, needs to be bound to user (certificate)

Private key D (decryption key)

Encrypt: with public key of recipient.
Only recipient can decrypt, $D(E(m)) = m$

Trapdoor Functions

Indexing function: convert family of functions $\{f_i\}$ with $i \in \Sigma^*$ to a single function $f(i, w) = f_i(w)$, $f : \Sigma^* \times \Sigma^* : \Sigma^*$.

Trapdoor function: easy to invert with extra info, hard without.
$f : \Sigma^* \times \Sigma^* : \Sigma^*$, with additional PPTM G and function $h : \Sigma^* \times \Sigma^* : \Sigma^*$, such that

1. f and h computable in polynomial time
2. $\Pr_{E,w}[E(i, f_i(w)) = y$ with $f_i(y) = f_i(w)] \leq n^{-k}$ for every PPTM E, every k, sufficiently large n, random $w \in \Sigma^n$, and random output $\langle i, t \rangle$ of G on 1^n.
3. For every n, $w \in \Sigma^n$, and every non-zero probability output $\langle i, t \rangle$ of G on some input,
   $h(t, f_i(w)) = y$, where $f_i(y) = f_i(w)$

G generates index i *and* trapdoor t that allows inverting f_i
(note that in (2), the trapdoor t is unavailable)

Trapdoor example: RSA

Key generation:

choose n-bit primes p, q (random numbers, test for primality)

compute $n = pq$ and Euler totient $\phi(n) = (p - 1)(q - 1)$
(count of numbers $< n$ and relatively prime to n)

choose *encryption exponent* $1 < e < \phi(n)$, relatively prime to $\phi(n)$
compute inverse d, $de \equiv 1 \pmod{\phi(n)}$.

public key: (n, e) *private* key: d.
Formally, PPTM G outputs $((n, e), d)$

Encryption: $f_{n,e}(w) = w^e \pmod{n}$
Decryption: $h(d, x) = x^d \pmod{n}$

Reason: $(w^e)^d = w^{de} = w \pmod{n}$ because $de \equiv 1 \pmod{\phi(n)}$
(from Euler's theorem and Chinese Remainder Theorem)