Chapter 2

Techniques (circuits, public-key cryptography)

2.1 Programs as circuits

Every program on inputs $x_1, ..., x_n$ computes a function $f(x_1, ..., x_n)$. Represented by a Turing machines or by a circuit.

2.1.1 Cook-Levin Theorem

> Every problem decided by a non-deterministic Turin machine (an NP-problem) in polynomial time can be formulated as the satisfiability problem (SAT) of a polynomial sized circuit.

From computer architecture, we know that every computation is represented by binary stateful circuits.
Here we formulate all programs as circuits and evaluate the circuits in "encrypted" form.

2.1.2 Circuit for testing equality of numbers

Given two numbers x and y in binary:

$$[x]_2 = x_{n-1}x_{n-2}...x_1x_0$$

$$[y]_2 = y_{n-1}y_{n-2}...y_1y_0$$

Determine if they are equal.

$i \leftarrow n$
while $i \geq 0$ *do*
  $i \leftarrow i - 1$
  if $x_i \neq y_i$ *then*
    return 0
return 1

A circuit must unroll the loop and execute all iterations.

$i \leftarrow n$
$d_n \leftarrow 0$
while $i \geq 0$ *do*
  $i \leftarrow i - 1$
  $d_i \leftarrow d_{i+1} \vee (x_i \oplus y_i)$
return $\neg d_0$ $\quad //d_0 \equiv x$ *differs from* y

2.2 Mathematics of public-key cryptography

- Modular arithmetic
  "computing modulo a number m" $(a \ mod \ m)$
- $\mathbb{Z}_m = 0, ..., m-1$
  with addition mod m
- $\mathbb{Z}_p^\star = 1, ..., p-1$
  with multiplication mod m
- A group $\mathbb{G}$ is cyclic if there exists some g and every element in $\mathbb{G}$ can be obtained by computing g^i, for some $i \geq 0$.
- $\mathbb{G}$ is finite
  $|\mathbb{G}|$ is the number of elements in $\mathbb{G}$
- $g^0, g^1, g^2, ..., g^{|\mathbb{G}|-1} = g^0$
- $\mathbb{G} =< g >$, g is generator of $\mathbb{G}$
- $\mathbb{Z}_m$ integers mod m with addition is cyclic
- $\mathbb{Z}_{11}^\star$:
  $< 3 > = 1, 3, 9, 5, 4 \neq \mathbb{Z}_{11}^\star$
  $< 2 > = 1, 2, 4, 8, 5, 10, 9, 7, 3, 6 = \mathbb{Z}_{11}^\star$
- $\mathbb{Z}_p^\star$ is cyclic for any prime p with $|\mathbb{Z}_p^\star| = p - 1$
- $q|p-1$ with q prime, then there is a cyclic group of order q, defined by multiplication modulo p
  $\Rightarrow$ used in cryptography

2.3 Discrete Logarithm Problem

- Computing logarithm is the inverse of exponentiation
- Given $\mathbb{G} =< g >$, then the discrete logarithmof some $y \in \mathbb{G}$ is the number i s.t. $g^i = y$. E.g. $\mathbb{G} \subset \mathbb{Z}_p^\star$, i s.t. $g^i \equiv y(\ mod\ p)$.

DLP

DLP: Given $y = g^x$ for $x \leftarrow \mathbb{Z}_q$, comput x

CDH

Given x, y s.t.:

$$x = g^a \ for\ a \leftarrow \mathbb{Z}_q$$
$$y = g^b \ for\ b \leftarrow \mathbb{Z}_q$$

compute $g^{a \cdot b}$. (Computational Diffie-Hellman Problem)

DDH

Given either x, y, z computed as:

$$x = g^a \ for\ a \leftarrow \mathbb{Z}_q$$
$$y = g^b \ for\ b \leftarrow \mathbb{Z}_q$$
$$z = g^c \ for\ c \leftarrow \mathbb{Z}_q$$

or given:

$$x = g^a \ for\ a \leftarrow \mathbb{Z}_q$$
$$y = g^b \ for\ b \leftarrow \mathbb{Z}_q$$
$$z = g^{a \cdot b}$$

decide, which is the case.

2.4 Public-Key Encryption

- $\textsc{KeyGen}() \rightarrow (pk, sk)$
- $\textsc{Enc}(pk, m) \rightarrow c$
- $\textsc{Dec}(sk, c) \rightarrow m'$

Completeness

$\forall m : (pk, sk) \leftarrow \textsc{KeyGen}()$
$\textsc{Dec}(sk, \textsc{Enc}(pk, m)) = m$

Security

- An encryption of a message m are indistinguishable from random elements of ciphertext space
- For two messages m_1, m_2, no adversary can distinguish the $\text{Enc}(pk, m_1)$ from $\text{Enc}(pk, m_2)$

2.4.1 ElGamal Public-Key Cryptosystem (Textbook)

$\text{KeyGen}()$
  $x \leftarrow \mathbb{Z}_q$
  $y \leftarrow g^x$
  return (y, x)

$\text{Enc}(y, m)$
  $r \leftarrow \mathbb{Z}_{11}$
  $R \leftarrow g^r$
  $c \leftarrow m \cdot y^r$
  return $((R, c))$

$\text{Dec}(x, (R, c))$
  $m' \leftarrow c/R^x$
  return (m')

Completeness

$$m' = c/R^x = m \cdot y^r/R^x = m \cdot g^{xr}/g^{rx} = m$$

Security

Decide whether, for some m, (y, R, c) is $(g^x, g^r, m \cdot g^{xr})$ or is $(g^x, g^r, m \cdot g^z)$ for $z \leftarrow \mathbb{Z}_q$, corresponds to the decisional DH-problem.

2.4.2 RSA Cryptosystem (Textbook)

Here exponentiation modulo N, where $N = p \cdot q$ and p, q are prime.
In practice, $|p| = |q| \approx 1000$.
$\mathbb{Z}_N^\star$: Set of integers mod N that are coprime with N $(N = p \cdot q)$
$|\mathbb{Z}_N^\star| = \phi(N) = (p-1)(q-1)$

Theorem

For all $a \in \mathbb{Z}_N^\star$ we have:

$$a^{\phi(N)} \equiv 1 \ \textit{(mod N)}$$

KeyGen()
$p, q \leftarrow$ random primes
$N \leftarrow p \cdot q$
e (fixed) exponent, prime
$d \leftarrow e^{-1} \ mod \ (\phi(N) = (p-1)(q-1))$
return $((N, e), d)$

Enc$((N, e), m)$
return $(m^e \ mod \ N)$

Dec(d, c)
return $(c^d \ mod \ N)$

Completeness

Dec$(d,$ Enc$((N, e), m)) =$ Dec$(d, m^e \ mod \ N) = m^{d \cdot e} \ mod \ N = m^{1+k\phi(N)} \ mod \ N = m$

Security

If factoring is easy, then RSA is broken.

2.5 Digital Signatures

- provides authenticity and integrity
- dual to public-key encryption

2.5.1 Digital Signature Scheme

KeyGen() $\rightarrow (pk, sk)$
Sign() $\rightarrow \sigma$
Verify$(pk, m, \sigma) \rightarrow$ false, true

Completeness

$$\forall m : (pk, sk) \leftarrow \text{KeyGen}() \\ \text{Verify}(pk, m, \text{Sign}(sk, m)) \ = \ \text{true}$$

Security

Security means that no adversary can create a message/sign pair, s.t. Verify$(pk, m, \sigma) =$ true, unless σ is output of Sign(sk, m).

2.6 Textbook RSA

Directly uses RSA function to produce the signatures - not secure!

KeyGen()
$p, q \leftarrow$ random primes
$N \leftarrow p \cdot q$
e (fixed) exponent, prime
$d \leftarrow e^{-1} \ mod \ (p-1)(q-1)$
$sk \leftarrow d$
$pk \leftarrow (N, e)$
return (pk, sk)

Sign(sk, m)
$\sigma \leftarrow m^d \ mod \ N$
return (σ)

Verify(pk, m, σ)
return $(\sigma^e \stackrel{?}{\equiv} m \ (mod \ N))$

Not secure because anyone could pick some $\sigma^\star$, compute $m \leftarrow \sigma^{\star e}$. This would pass the verification algorithm.
To make it secure we can use a Hash-Function to compute $\sigma = \mathbb{H}(m)^d \bmod N$, where $\mathbb{H}$ is a full-domain Hash-Function. Also in the verification $\mathbb{H}$ must be used on m.

2.7 Schnorr Signatures

Group $\mathbb{G} =< g >$, cyclic
$\mid \mathbb{G} \mid = q$, q prime
$\mathbb{G} \subset \mathbb{Z}_p$, p prime
$p = m \cdot q + 1$
DLP is hard in $\mathbb{G}$
Hash function $\mathbb{H} : \{0,1\}^\star \rightarrow \mathbb{Z}_p$
KeyGen()
$x \leftarrow \mathbb{Z}_q$
$y \leftarrow g^x$
return (y, x)
Sign(x, m)
$r \leftarrow \mathbb{Z}_{\text{ıı}}$
$t \leftarrow g^r$
$c \leftarrow \mathbb{H}(m \| t)$
$s \leftarrow r - c \cdot x$
return $(((c, s)))$

Verify$(y, m, (c, s))$
return $(c \stackrel{?}{\equiv} \mathbb{H}(m \| g^s \cdot y^c)$ (in $\mathbb{Z}_p$))

2.7.1 Completeness

$$\begin{aligned} \hat{t} &= g^s \cdot y^c \\ &= g^{r-cx} g^{x \cdot c} \\ &= g^r \\ &= t \end{aligned}$$

2.7.2 Security

One can prove that breaking this is equivalent to computing discrete log of y.