6.857 Recitation 08: Public-Key Cryptography

Conner Fromknecht

April 5, 2016

Today

- Sage Demo
- El Gamal Recap
- IND–CCA2
- Cramer-Shoup Cryptosystem
- Elliptic Curve Pedersen Commitments

Sage Demo

If you don't have Sage installed, you can make an account at `https://cloud.sagemath.com` to get free access to a Sage terminal. The following example shows how to setup an Elliptic Curve and demonstrates simple operations on the curve's points.

```
sage: Field = Zmod(p)
sage: Curve = EllipticCurve(Field, [a, b])
sage: G = Curve.point((x, y))
# G = (x : y : z) written in projective form
# Interpret as (x/z, y/z) for z = {0,1}
# So z = 0 is point at infinity
sage: P = 2*G
sage: B = G + 15*P
# B = 31*G
```

1 El Gamal Recap [1, p. 365]

- Defined over a cyclic group $\mathbb{G}$ with order q and generator g.
- `Gen`(1^λ): Construct group $(\mathbb{G}, q, g) = \mathcal{G}(1^\lambda)$. Choose $x \xleftarrow{R} \mathbb{Z}_q$ and compute $h = g^x$. Public key $pk = (\mathbb{G}, q, g, h)$ and private key $sk = (\mathbb{G}, q, g, x)$.
- `Enc`$(pk, m \in \mathcal{G})$: Choose $r \xleftarrow{R} \mathbb{Z}_q$ and output ciphertext $c = (g^r, m \cdot h^r)$.
- `Dec`(sk, c): Let $c = (c_1, c_2)$. Compute $c_2 / c_1^x = m$.

RSA Recap [1, p. 355]

- Defined for $N = pq$ where p and q are large primes.
- $\texttt{Gen}(1^\lambda)$:
  - Run $\texttt{GenRSA}(1^\lambda)$ to obtain N, e, and d
  - Public key $pk = (N, e)$
  - Secret key $sk = (N, d)$
- $\texttt{Enc}(pk, m \in Z_N^*)$: Compute $c = m^e \mod N$
- $\texttt{Dec}(pk, c \in Z_N^*)$: Compute $m = m^d \mod N$

where we define $\texttt{GenRSA}(1^\lambda)$:

- $(N, p, q) \leftarrow \texttt{GenModulus}(1^\lambda)$
- Let $\phi(N) = (p-1)(q-1)$
- Choose e such that $\gcd(e, \phi(N)) = 1$
- Compute $d = e^{-1} \mod \phi(N)$
- Return N, e, d

IND–CCA2

Indistinguishability under *Adaptive* Chosen Ciphertext Attacks is defined as a two phase game between an examiner $\mathcal{E}$ and an adversary $\mathcal{A}$.

- Strongest notion of security for public key encryption.
- Mathematically captures the idea that the adversary can't do better than guessing, even after extensive access to the challenge ciphertext and oracle.

Phase 1: Find

- $\mathcal{E}$ generates (pk, sk) using $\texttt{Gen}(1^\lambda)$
- $\mathcal{E}$ send pk to adversary $\mathcal{A}$
- $\mathcal{A}$ computes for polynomial time in λ, with access to decryption oracle $\texttt{Dec}(sk, \cdot)$
- $\mathcal{A}$ outputs m_0 and m_1 and any state information s. ($|m_0| = |m_1|$ and $m_0 \neq m_1$)

Phase 2: Guess

- $\mathcal{E}$ picks $b \xleftarrow{R} \{0, 1\}$ and computes $c' = \texttt{Enc}(pk, m_b)$
- $\mathcal{E}$ sends (c', s) to adversary
- $\mathcal{A}$ computes for polynomial time in λ, again with access to $\texttt{Dec}(sk, \cdot)$ for any input except c'.
- $\mathcal{A}$ outputs $\hat{b}$, his guess for b.

$\mathcal{A}$ wins if $\hat{b} = b$. Encryption scheme is IND–CCA2 secure if $Pr[\hat{b} = b] \leq \frac{1}{2} + negl(\lambda)$.

Cramer-Shoup Cryptosystem

Problem: El Gamal encryption exhibits multiplicative homomorphism, so an attacker can create valid encryptions of other messages. Given two ciphertexts

$$c_1 = \texttt{Enc}(pk, m_1) = (g^r, m_1 \cdot y^r)$$
$$c_2 = \texttt{Enc}(pk, m_2) = (g^s, m_2 \cdot y^s)$$

we can compute

$$\begin{aligned} c_1 \cdot c_2 &= (g^{r+s}, (m_1 \cdot m_2) \cdot y^{r+s}) \\ &= \texttt{Enc}(pk, m_1 \cdot m_2) \end{aligned}$$

Solution: Cramer-Shoup cryptosystem—solves malleability in El Gamal. Creates IND–CCA2 encryption scheme, defined over group cyclic group $\mathbb{G}$ with prime order q.

$\texttt{Gen}(1^\lambda)$

- Choose $g_1, g_2 \xleftarrow{R} \mathbb{G}$
- Choose secret key $sk = (x_1, x_2, y_1, y_2, z) \xleftarrow{R} \mathbb{Z}_q$
- Hash function $H : \mathbb{G}^3 \to \mathbb{Z}_q$, maps three elements in $\mathbb{G}$ to $\mathbb{Z}_q$.
- $c = g_1^{x_1} g_2^{x_2},\ d = g_1^{y_1} g_2^{y_2},\ h = g_1^z$
- Public key $pk = (g_1, g_2, c, d, h, H)$

$\texttt{Enc}(pk, m \in \mathbb{G})$

- Choose $r \xleftarrow{R} \mathbb{Z}_q$
- $u_1 = g_1^r,\ u_2 = g_2^r,\ e = m \cdot h^r$
- $\alpha = H(u_1, u_2, e)$
- $v = c^r d^{r\alpha}$
- Ciphertext $c = (u_1, u_2, e, v)$

$\texttt{Dec}(sk, c)$

- $\alpha = H(u_1, u_2, e)$
- If $u_1^{x_1 + y_1\alpha} u_2^{x_2 + y_2\alpha} \neq v$, REJECT.
- $m = e / u_1^z$

Elliptic Curve Pedersen Commitments

Similar to El Gamal Pedersen Commitments, provides *perfect hiding* for the committed value.

- $\texttt{Setup}(1^\lambda)$: Construct $(\mathbb{E}_p, q, G) = \mathcal{G}(1^\lambda)$ with prime order q and generator G. Choose secret $a \xleftarrow{R} \mathbb{Z}_q$ and compute public $H = aG$. Output $(\mathbb{E}_p, q, G, H)$.

- $\texttt{Commit}(x \in Z_q)$: Choose $r \xleftarrow{R} \mathbb{Z}_q$. Compute commitment $c = xG + rH$.
- $\texttt{Reveal}(x, r \in \mathbb{Z}_q)$: Check $c = xG + rH$.

Perfect Hiding

Possible for given commit $c = \texttt{Commit}(x)$ to reveal any x'?

$$\begin{aligned} c = xG + rH &= x'G + r'H \\ xG + arH &= x'G + ar'H \\ (x + ar)G &= (x' + ar')H \\ x + ar &\equiv x' + ar' && \mod q \\ r' &\equiv (x - x')/a + r && \mod q \end{aligned}$$

We know $\exists\ a$, since q is prime. In addition $x \neq x'$, so $r \neq r'$.

Computationally Binding

Possible to compute x' and r'?

$$\begin{aligned} xG + rH &= x'G + r'H \\ x + ar &= x' + ar' \\ a &= (x - x')/(r - r') \\ &= \log_G H \mod p. \end{aligned}$$

Would require breaking DLP on $\mathbb{G}$.

Malleability—Additive Homomorphism

$$\begin{aligned} c &= \texttt{Commit}(x) = xG + rH \\ c' &= G + c = (x + 1)G + rH \end{aligned}$$

References

[1] J. Katz and Y. Lindell. Introduction to modern cryptography, 2008.