ElGamal Cryptosystem and variants

BIL 448/548
Internet Security Protocols
Ali Aydin Selçuk

Structure of \( \mathbb{Z}_p^* \)

For a prime \( p \), \( \mathbb{Z}_p^* \) is all non-zero elements of \( \mathbb{Z}_p \).

Fermat’s (Little) Theorem: For all \( x \in \mathbb{Z}_p^* \), we have \( x^{p-1} \equiv 1 \pmod{p} \).

Let \( <g> \) denote the numbers generated by powers of \( g \) in \( \mathbb{Z}_p^* \): \( <g> = \{g, g^2, \ldots, g^{p-1}\} \).

E.g. for \( \mathbb{Z}_5^* \):
- \( <1> = \{1\} \)
- \( <2> = \{2, 4, 3, 1\} \)
- \( <3> = \{3, 4, 2, 1\} \)
- \( <4> = \{4, 1\} \)

- “order” of 1 is one; of 4 is two; of 2 & 3 is four.
- 2 & 3 are “generators” of \( \mathbb{Z}_5^* \) (they have order p-1).
- Fact: For every prime \( p \), \( \mathbb{Z}_p^* \) has a generator.

ElGamal – Encryption

Parameters:
- \( p \), a large prime
- \( g \), a generator of \( \mathbb{Z}_p^* \)
- \( \alpha \in \mathbb{Z}_{p-1} \), \( \beta = g^\alpha \mod p \)
- \( p, g, \beta \) public; \( \alpha \) private

Encryption:
- generate random, secret \( k \in \mathbb{Z}_{p-1} \).
- \( E(x, k) = (r, s) \), where
  \[
  r = g^k \mod p \\
  s = x^\beta^k \mod p
  \]
- \( D(r, s) = s(r^\alpha)^{-1} \mod p = x \cdot g^{\alpha k} g^{-\alpha k} \mod p = x \).

ElGamal – Encryption

- Plaintext \( x \) is masked by a random factor, \( g^{\alpha k} \mod p \).
- DH problem: Given \( g^\alpha, g^k \mod p \), what is \( g^{\alpha k} \mod p \)?
- \( p, g \) can be common. Then \( g^k \mod p \) can be computed in advance.
- Same \( k \) should not be used repeatedly.
- Performance:
  - encryption: two exponentiations
  - decryption: one exponentiation, one inversion
- Size: Ciphertext twice as large as plaintext.
ElGamal – Signature

Parameters: The same as encryption.

Signature:
- generate random, secret $k \in \mathbb{Z}_{p-1}$.
- $S(m, k) = (r, s)$, where
  $$r = g^k \mod p$$
  $$s = (m - r\alpha)k^{-1} \mod (p - 1)$$
  (i.e., $m = r\alpha + sk$)

Verification:
- Is $\beta^r r^s \equiv g^m \pmod p$?
- $\beta^r r^s = g^{ar^k(m - r\alpha)k^{-1}} = g^{ar^k(m - r\alpha)} = g^m \mod p$.

Security:
- Only one who knows $\alpha$ can sign; can be verified by $\beta$.
- Solving $\alpha$ from $\beta$, or $s$ from $r$, $m$, $\beta$, is discrete log.
- Other ways of forgery? Unknown.
- Same $k$ should not be used repeatedly.

Variations:
- Many variants, by changing the "signing equation", $m = r\alpha + sk$.
- E.g., the DSA way:
  $$m = -r\alpha + sk$$
  with verification: $\beta^r g^m \equiv r^s \pmod p$? ($\equiv g^{m + r\alpha}$)

Digital Signature Algorithm (DSA)

• US government standard, designed by NSA.
• Based on ElGamal & Schnorr:
  - patent-free (ElGamal)
  - can’t be used for encryption
• Objections:
  - ElGamal was not analyzed as much as RSA
  - slower verification
  - industry had already invested in RSA
  - closed-door design

Let $q | (p-1)$ be prime, and $g \in \mathbb{Z}_p^*$ be of order $q$.

Schnorr group: The subgroup in $\mathbb{Z}_p^*$ generated by $g$, of prime order $q$.
$$<g> = \{1, g, g^2, \ldots, g^{q-1}\}$$

Fact: $q$ can be much shorter than $p$ (e.g. 160 vs. 1024 bits), and the hardness of DLP in $<g>$ remains the same.
DSA (cont’d)

Parameters: prime p, prime q | (p-1), and \( g \in \mathbb{Z}_p \)

of order q. Hash fnc. \( H: \{0,1\}^* \rightarrow \mathbb{Z}_q \).

Keys: \( \alpha \in \mathbb{Z}_q \) is private; \( \beta = (g^\alpha \mod p) \) is public.

Signature: \((r,s)\) where
- \( v = g^k \mod p \)
- \( r = v \mod q \)
- \( s = (H(M) + r \alpha) k^{-1} \mod q \)

Verification:
- \( v' = g^{H(M)} s^{-1} \beta^{-1} s^{-1} \mod p \)
- \( r = v' \mod q \)

Advantage: Reduced size (\( r, s \) are 160-bit)

Elliptic Curve Cryptosystems

Generalized Discrete Log Problem:
- For any group \((G, \cdot)\), for \( x \in G \), define \( x^n = x \cdot x \cdot \ldots \cdot x \) (n times)
- DLP: For \( y = x^n \), given \( x, y \), what is \( n \)?

Elliptic curves over \( \mathbb{Z}_p \):
- Set of points \((x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p \) that satisfy \( y^2 \equiv x^3 + ax + b \) (mod \( p \))
- Well-defined, provided that \( 4a^3 \neq -27b^2 \) (mod \( p \)).

Elliptic Curve Cryptosystems (cont’d)

EC example over \( \mathbb{R}^2 \):

- \( P (3.35, 1.96) \)
- \( P'(3.89, -5.63) \)
- \( R (3.39, 2.63) \)
- \( Q (4.1, 0.83) \)
- \( R' (3.89, -5.63) \)
- \( P + Q - R = (3.89, -5.63) \).

Elliptic Curve Cryptosystems (cont’d)

• Facts for an EC over a finite field:
  - Exponentiation is efficient.
  - DLP is hard. In fact, harder than in \( \mathbb{Z}_p \) (no sub-exponential algorithm is known)
- Hence, DH, ElGamal, etc. can be used with smaller key sizes over ECs. (160-bit EC ~ 1024-bit RSA)

• Popular for constrained devices (e.g., smart cards)

• Advantages over RSA:
  - smaller key size
  - compact in hardware
  - faster (for private key operations)

• Licensed by NSA.
### ECC vs. RSA

NIST guidelines for key sizes (bits) with eqv. security levels:


<table>
<thead>
<tr>
<th>Symmetric Key</th>
<th>RSA/DH/ElGamal</th>
<th>ECC</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>1024</td>
<td>160</td>
</tr>
<tr>
<td>112</td>
<td>2048</td>
<td>224</td>
</tr>
<tr>
<td>128</td>
<td>3072</td>
<td>256</td>
</tr>
<tr>
<td>192</td>
<td>7680</td>
<td>384</td>
</tr>
<tr>
<td>256</td>
<td>15360</td>
<td>512</td>
</tr>
</tbody>
</table>

(according to our current knowledge of algorithms for factorization, DLP, and EC DLP)