Cryptography, winter 2014/15

ElGamal encryption and RSA

Dr. Daniel Loebenberger

computer
cosec b-it
security

Cryptosystem. ElGamal encryption scheme.

General setup:

Input: 1^n.

1. Choose a finite cyclic group $G = \langle g \rangle$ with d elements, where d is an n-bit integer, and a generating element g. These data, including a description of G, are made public.

Key generation:

Output: $(\mathsf{sk}, \mathsf{pk})$.

2. Choose a secret key $\mathsf{sk} = b \xleftarrow{} \mathbb{Z}_d$ at random. The public key is $\mathsf{pk} = B = g^b \in G$.

CRYPTOSYSTEM. ElGamal encryption scheme.

Encryption:

Input: Plaintext $x \in G$.
Output: Ciphertext $\mathsf{enc}_{\mathsf{pk}}(x) \in G^2$.

1. Choose a secret session key $a \longleftarrow \mathbb{Z}_d$ at random.
2. Public session key $A \longleftarrow g^a \in G$, and common session key $k \leftarrow B^a = g^{ab} = A^b$.
3. $y \leftarrow x \cdot k \in G$, Return $\mathsf{enc}_{\mathsf{pk}}(x) = (y, A)$.

Decryption:

Input: Arbitrary $(y, A) \in G^2$.
Output: Decryption $\mathsf{dec}_{\mathsf{sk}}(y, A) \in G$.

4. Common session key $k \leftarrow A^b$, inverse $k^{-1} \in G$ of the common key.
5. $z \leftarrow y \cdot k^{-1}$, Return $\mathsf{dec}_{\mathsf{sk}}(y, A) = z$.

Let $G = \mathbb{Z}_{2579}^{\times}$ and $g = 2$ again. Bob has published his public key $B = g^b = 949$. Alice wants to encrypt the plaintext `MY SECRET` and writes it in the familiar numerical notation $(12, 24, 18, 4, 2, 17, 4, 19)$, where `A` corresponds to 0, etc. Spaces are often ignored in cryptography. Alice sends two letters b_1, b_2 as one unit by computing $26b_1 + b_2$. She uses different secret session keys to avoid statistical attacks, and sends (y, A) as her message.

plaintext		encryption				decryption		
letters	x	a	$A = g^a$	$k = B^a$	$y = x \cdot k$	$k = B^a$	k^{-1}	$k^{-1}y$
MY	336	2111	1617	2430	1516	2430	1402	336
SE	472	367	734	2474	2020	2474	2186	472
CR	69	351	832	203	1112	203	559	69
ET	123	161	1355	1483	1879	1483	873	123

Theorem

For any group G, breaking the ElGamal cryptosystem and solving the Diffie–Hellman Problem can be reduced to each other in polynomial time. The reductions can be done using at most $O(n^2)$ bit operations, where n is the bit size of G.

Cryptosystem. RSA.

Key Generation key gen.

Input: Security parameter n.
Output: secret key sk and public key pk.

1. Choose two distinct primes p and q at random with $2^{(n-1)/2} < p$, $q < 2^{n/2}$.
2. $N \leftarrow p \cdot q$, $L \leftarrow (p-1)(q-1)$. [N is an n-bit number, and $L = \phi(N)$ is the value of Euler's ϕ function.]
3. Choose $e \xleftarrow{} \{2, \ldots, L-2\}$ at random, coprime to L.
4. Calculate the inverse d of e in $\mathbb{Z}_L$.
5. Publish the public key pk $= (N, e)$ and keep sk $= (N, d)$ as the secret key.

CRYPTOSYSTEM. RSA.

Encryption enc.

Input: $x \in \mathbb{Z}_N$, $\mathsf{pk} = (N, e)$.
Output: $\mathsf{enc}_{\mathsf{pk}}(x) \in \mathbb{Z}_N$.

1. $y \leftarrow x^e$ in $\mathbb{Z}_N$.
2. Return $\mathsf{enc}_{\mathsf{pk}}(x) = y$.

Decryption dec.

Input: $y \in \mathbb{Z}_N$, $\mathsf{sk} = (N, d)$.
Output: $\mathsf{dec}_{\mathsf{sk}}(y) \in \mathbb{Z}_N$.

3. $x^* \leftarrow y^d$ in $\mathbb{Z}_N$.
4. Return $\mathsf{dec}_{\mathsf{sk}}(y) = x^*$.

security parameter n,
distinct random primes p and q of $n/2$ bits,
$N = pq$ of n bits,
$L = \phi(N) = (p-1)(q-1)$,
$e, d \in \mathbb{Z}_L \setminus \pm 1$ with $ed = 1$ in $\mathbb{Z}_L$,
plaintext x, ciphertext y, decryption x^*, all in $\mathbb{Z}_N$,
$y = x^e$, $x^* = y^d$.

Figure: The RSA notation.

Chinese Remainder Theorem

Let $N = q_1 \cdots q_r$ with pairwise coprime integers $q_1, \ldots, q_r$. Then the ring homomorphism

$$\begin{array}{rcl} \mathbb{Z}_N & \to & \mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_r}, \\ x \bmod N & \mapsto & (x \bmod q_1, \ldots, x \bmod q_r) \end{array}$$

is an isomorphism. In other words: given integers $a_1, \ldots, a_r$, there exists an integer $x \in \mathbb{Z}$ that solves the congruences

$$x = a_i \text{ in } \mathbb{Z}_{q_i} \text{ for } 1 \leq i \leq r$$

simultaneously, and two such solutions x and x' differ by a multiple of N, so that $x = x'$ in $\mathbb{Z}_N$.

Theorem

RSA works correctly, that is, for every message x the decrypted encrypted message z equals x.