Introduction to Cryptology

Lecture-12
Public-Key Cryptography
Quadratic Residues and „Rabin Lock“

17.05.2023, v54

Rabin Lock for a Public-Key System is Based on the Square Root Problem in a Finite Ring (1979)

Michael Oser Rabin, 1931, Breslau, Germany

Rabin Crypto-System 1979

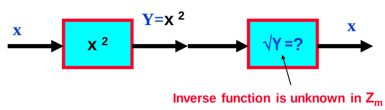
Basic idea: **Squaring in a ring modulo m=pq.**

Claim: Square root computation in the ring Z_m, where m=p•q is not feasible If the factors p&q of the modulus m are not known!

Squaring and Square Roots in Z_m (Rabin Lock)

Claim: the function $Y = X^2$ is one-way in Z_m if m is composite!

Squaring: $Y = x^2 \pmod{m}$



We investigate two cases for computing the square root in Z_m:
1. The modulus m is a prime p that is [in GF(p)]
2. The modulus is non-prime, [in the Ring Z_m, where m is a product of two primes p and q].

First Case : Squaring and Square Roots in GF(p)

Quadratic Residues QR, and Quadratic non-Residues QNR in GF(p)

Squaring in GF(p)

Example: $y = x^2 \pmod{7}$ i.e. in GF(7)

x	1	2	3	4	5	6
$y = x^2$	1	4	2	2	4	1

Quadratic Residues QR

$\sqrt{1}$ = 1 and 6 ⇔ [±1 in GF(7)]
$\sqrt{4}$ = 2 and 5 ⇔ [± 2 in GF(7)]
$\sqrt{2}$ = 3 and 4 ⇔ [±3 in GF(7)]

⇒ 1, 2, 4 are the QR's in GF(7) (Elements having square root)

Quadratic non-Residues QNR

$\sqrt{3}$ = does not exist in GF(7)
$\sqrt{5}$ = does not exist in GF(7)
$\sqrt{6}$ = does not exist in GF(7)

⇒ 3, 5, 6 are the QNR's in GF(7) (Elements having no square root)

Fact: There are (p-1)/2 QR and (p-1)/2 QNR in GF(p)

First Case : Squaring and Square Roots in GF(p)

How to identify Quadratic Residues QR, and Quadratic non-Residues QNR

How to identify QR and QNR in GF(p) :

If $\beta \in GF(p)$ and $\beta \neq 0$ then:

- β is **QR** if $\beta^{(p-1)/2} = 1 \pmod{p} \Rightarrow (\beta^{(p-1)/2} - 1) = 0$
- β is **QNR** if $\beta^{(p-1)/2} = -1 \pmod{p} \Rightarrow (\beta^{(p-1)/2} + 1) = 0$

Proof:
The roots of $x^{(p-1)} - 1 = (x^{(p-1)/2} - 1)(x^{(p-1)/2} + 1)$ are the units of GF(p)

If α is the SQRT of β then $\beta = \alpha^2$
=> $\beta^{(p-1)/2} = \alpha^{p-1} = 1$ (Fermat Theorem) => $(\beta^{(p-1)/2} - 1) = 0$ are the QR's above
the others are the QNR's. The count of each is (p-1)/2

Note: There are no deterministic techniques known to generate QNRs in GF(p) !

Computing Square Roots in GF(p)

How to compute square roots for Quadratic Residues QR?

Case 1 : If (p-1)/2 is odd (that is p+1 is divisible by 4) and β is a QR in GF(p),

then the two square roots of β are:

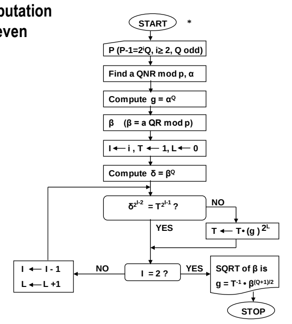
$$\alpha = \beta^{(p+1)/4}$$

and $$-\alpha = p - \alpha$$

Case 2: if (p-1)/2 is even, then see the following Algorithm delivers both roots for quadratic residues in GF(p):

Case 2: A Square-Root Computation in GF(p) for (p-1)/2 even

(Shanks' Algorithm)



* J. L. Massey

Second Case : Squaring and Square Roots in a Ring Z_m

(m = p . q is not a prime)

Squaring in Z_m

Example: m=p. q is a composite of two primes m= 3 x 5

The function $y \equiv x^2 \pmod{15}$ is shown below:

	units								Non-units					
x	1	2	4	7	8	11	13	14	3	5	6	9	10	12
$y = x^2$	1	4	1	4	4	1	4	1	9	10	6	6	10	9

1, 4 — 6, 9, 10

Quadratic Residues

The units : 1, 4 are the QR's in Z^*_{15} ---------------->

The units : 2, 7, 8, 11, 13, 14 are the QNR's in Z^*_{15}

$\sqrt{1}$ = 1 and 14 ⇔ [±1 in Z_{15})]
= 4 and 11 ⇔ [±4 in Z_{15})]

$\sqrt{4}$ = 2 and 13 ⇔ [±2 in Z_{15})]
= 7 and 8 ⇔ [±7 in Z_{15})]

Fact: for m= p . q There are (p -1) (q -1)/4 QR in Z^*_m.
Each QR has 4 distinct square roots

Computing Square Roots in Z_m if m = p . q

No algorithm is known for computing the square roots of any unit element in Z_m if the prime factors of m, p and q are not known

!! There is a Computational Equivalence Between Factoring m= p q and taking Square Roots in Z_m !!!

Fact: If m= p q where p and q, are distinct odd primes and two different SQRT's α and β of some QR in Z_m are known, where $\alpha \neq \beta$ and $\alpha \neq -\beta$, then:

either gcd ($\alpha + \beta$, m) = p

or gcd ($\alpha + \beta$, m) = q

Computing Square Roots in Z_m if m factors p , q are known

four square roots for a QR element **c** modulo m do exist: r_1, r_2, r_3, and r_4

That is: $\sqrt{c} = r_1, r_2, r_3, r_4$

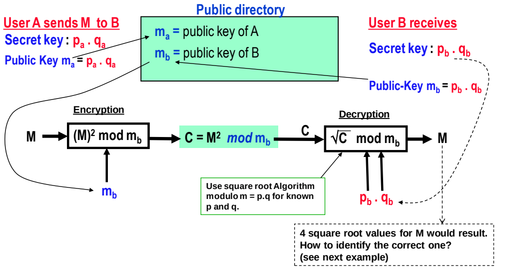
Computing the square roots **if p+1 and q+1, are divisible by 4:**

1. Compute **a** and **b** satisfying **gcd(p,q) = a · p + b · q = 1**, using the extended gcd algorithm.
2. Compute **r** = $c^{(p+1)/4}$ mod p (Square root mod p).
   Compute **s** = $c^{(q+1)/4}$ mod q (Square root mod q) .
3. Apply the Chines Remainder Theorem:
   x = (a · p · s + b · q · r) mod m
   y = (a · p · s - b · q · r) mod m

=> the four-square roots are: $r_1 = x$, $r_2 = -x$, $r_3 = y$, $r_4 = -y$

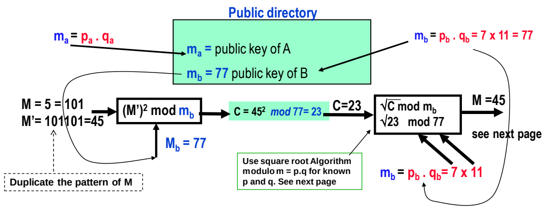
Computing the square roots if p and q mod 4 ≠ 3 (p+1 and q+1 are not divisible by 4) require using Shanks' algorithm in page 8 to compute r and s

Rabin Secrecy-System (1979)



Example: Rabin Secrecy-System

Setup and calculate Cryptogam and decrypt the message M=5 for a user with the public key $m_b = 7 \times 11 = 77$



Solution Cont.: See square root algorithm calculations in Z_m:

Encryption:

Messages must be in the range from **1 to 7**, so this system of redundancy will work. Start with data bits **$M=101_2$ or 5_{10}**. The replication gives **$M' = 101101_2$ or 45_{10}**.

Then **c** = M'^2 mod 77 = **23**.

Decryption:

Take p = 7, q = 11, and n = 77.

Compute gcd(11,7) = (-3)*7 + 2*11 = 1 => that is **a = -3** and **b = 2**.

To compute the square roots of C modulo 77 compute r and s :

r = $c^{(p+1)/4}$ mod p => r = 23^2 mod 7 = 4

s = $c^{(q+1)/4}$ mod q => s = 23^3 mod 11 = 1

Then **x = (a*p*s + b*q*r) mod m** => **x** = ((-3)*7*1 + 2*11*4) mod 77 = **67**

y = (a*p*s - b*q*r) mod m => **y** = ((-3)*7*1 - 2*11*4) mod 77 = **45**

x and **y** are two of the four square roots, and the remaining two are

-x mod 77 = **-67** mod 77 = **10**

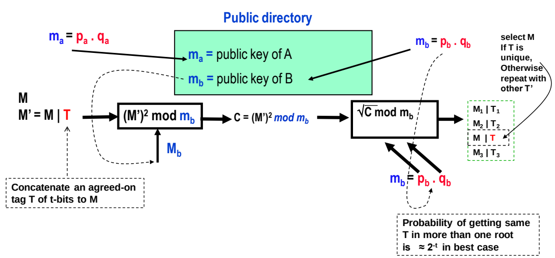
-y mod 77 = **-45** mod 77 = **32**

In binary, the four-square roots are
$67 = 1000011_2$
$45 = 0101101_2$
$10 = 0001010_2$
$32 = 0100000_2$

The only sqare root with two equal blocks delivers the correct result

One of these roots is M'. Only 45 has the required repetition redundancy, so this is the only possible message **M'=45 =** 101101 => **M = 101**.

Alternative constellation for Rabin Secrecy-System



Rabin Signature Scheme Based on Rabin Lock

Setup: n = p .q is public, p and q are two secret primes generated by the signer

Signing: The message hash value *H(m)* is signed, where *m* is the clear message

if $H(m)^{\frac{p-1}{2}} \bmod p = 1$ AND $H(m)^{\frac{q-1}{2}} \bmod q = 1$ *H(m) is* QR in GF(p) and GF(q)

The signature S is computed as:

$$S = \left(\left(p^{q-2} H(m)^{\frac{q+1}{4}} \bmod q\right) p + \left(q^{p-2} H(m)^{\frac{p+1}{4}} \bmod p\right) q\right) \bmod (p \cdot q)$$

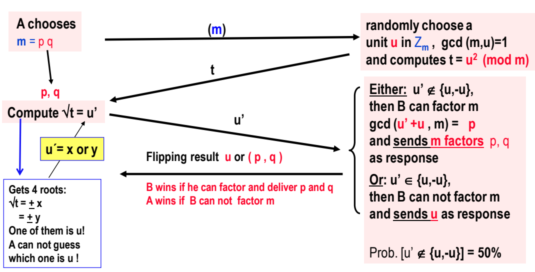
The signed message M is : (M,S)

Verification: Anybody knows *H(s)* and the public key *n* can verify the signature as follows: $H(m) = S^2 \bmod n$

H(x) should be a hash function with high collision resistance!

(Rabin-Lock based application-1)

Fair Coin-Flipping Using a Blind Communication

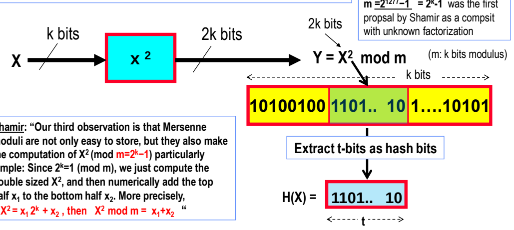


(Rabin-Lock based application-2)

SQUASH Hash Function (Shamir) 2007-2008

Key Idea: Square the input value X in a ring Z_m and take a part of the resulting square vector as a hash value m= is a composite with unknown factorization

$m = 2^{1277}-1 = 2^k-1$ was the first propsal by Shamir as a compsit with unknown factorization



Shamir: "Our third observation is that Mersenne moduli are not only easy to store, but they also make the computation of X^2 (mod $m=2^k-1$) particularly simple: Since $2^k=1$ (mod m), we just compute the double sized X^2, and then numerically add the top half x_1 to the bottom half x_2. More precisely, if $X^2 = x_1 2^k + x_2$, then $X^2 \bmod m = x_1+x_2$ "