An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic
Significance Function

Stephen C. Pohlig and Margin E. Hellman
IEEE Transactions on Information Theory, 1978

2010. 03. 09.
Kanghoon Lee, AIPR Lab., KAIST
What’s the problem?

Pair of Inverse Functions

\[ y \equiv \alpha^x \quad (\text{mod } p) \]
\[ x \equiv \log_{\alpha} y \quad \text{over } GF(p) \]

where \( p \) is prime, \( \alpha \) is a fixed primitive element of \( GF(p) \)

① \( y \equiv \alpha^x \quad (\text{mod } p) \) : \( O(\log_2 p) \) time complexity  
   (ex. \( \alpha^{18} = (((\alpha^2)^2)^2)^2 \alpha^2 \))

② \( x \equiv \log_{\alpha} y \quad \text{over } GF(p) \) : Previously, \( O(\sqrt{p}) \) time & space complexity

**One-way Function** : Original problem – easy
Inverse problem – difficult

Really?
\textbf{p-1 Must Have Large Prime Factor!}

\[ x \equiv \log_a y \text{ over } GF(p) \]

✓ Can we solve this problem faster than \( O(\sqrt{p}) \) time??

✓ Over GF(p), when \textbf{p-1 has only small prime factors},
  the logarithm problem can be solved \( O(\log^2 p) \)

✓ To make one-way function,
  \textbf{p-1 must have at least one large prime factor}
Use in Cryptography

✓ For plain-text M, key K, cipher-text C with the restrictions
\[ 1 \leq M, C \leq p - 1, \quad 1 \leq K \leq p - 2, \quad \text{GCD}(K, p - 1) = 1 \]
\[
C \equiv M^K \pmod{p}
\]

✓ For deciphering operation,
\[
M \equiv C^D \pmod{p}, \quad \text{where} \quad D \equiv K^{-1} \pmod{p - 1}
\]
(D is uniquely determined because \( \text{GCD}(K, p-1) = 1 \))

✓ Finding the key K is equivalent to computing
\[
K \equiv \log_M C \quad \text{over } GF(p)
\]
Background – Finite Field (Algebra)

✓ **GF(p)**: Galois Field (a.k.a. Finite Field)

⇒ A field that contains only finitely many elements

✓ Computations over GF(p)

ex. When \( p = 5 \) (i.e. \( GF(5) \))

\[
3 + 4 \equiv 2 \pmod{5}, \quad 3 - 4 \equiv 4 \pmod{5}
\]

\[
3^2 = 9 \equiv 4 \pmod{5}, \quad \log_3 4 \equiv 2 \pmod{5}
\]

✓ **Primitive Element**: A generator of the multiplicative group of the field

ex. \( 3^1 \equiv 3, \quad 3^2 \equiv 4, \quad 3^3 \equiv 2, \quad 3^4 \equiv 1 \pmod{5} \)

So, \( 3 \) is a primitive element of \( GF(5) \)
Euler’s \( \varphi \)-function (a.k.a. Euler’s totient function)

\[
\varphi(p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}) = (p_1 - 1)p_1^{e_1} (p_2 - 1)p_2^{e_2} \cdots (p_k - 1)p_k^{e_k}, \text{ where } p_i \text{'s are prime}
\]

\[
= p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k} \prod_{p_i} \left(1 - \frac{1}{p_i}\right)
\]

The fraction \( \rho \)

\[
\rho = \frac{\varphi(p-1)}{p-1} = \prod_{p_i|(p-1)} \left(1 - \frac{1}{p_i}\right) \quad \left(\forall p < 1.6 \times 10^{103} \Rightarrow p > 0.1\right)
\]

For primes of the form \( p = 2p'+1 \), with \( p' \) prime, \( \rho = \frac{1}{2} \left(1 - \frac{1}{p'}\right) \approx \frac{1}{2} \)
**Background - Number Theory (2)**

- **Fermat’s Little Theorem**
  
  \[ z^{p-1} \equiv 1 \pmod{p}, \quad 1 \leq z \leq p-1 \]

- From the theorem
  
  \[ z^x \equiv z^{x \pmod{p-1}} \pmod{p} \]

- **Chinese Remainder Theorem**
  
  Suppose that \( n_1, n_2, \ldots, n_k \) are positive integers which are pairwise coprime.

  For any integers \( a_1, a_2, \ldots, a_k \), there exist an integer \( x \pmod{n_1n_2\ldots n_k} \) satisfying

  \[ x \equiv a_1 \pmod{n_1}, \quad x \equiv a_2 \pmod{n_2}, \quad \ldots, \quad x \equiv a_k \pmod{n_k} \]

- **c.f. Euler’s Theorem**
  
  For any positive integer \( n, z \) ( \( \text{GCD}(n,z) = 1 \) )

  \[ z^{\phi(n)} \equiv 1 \pmod{n} \]
An Algorithm for $p = 2^n + 1$ (1)

✓ Given $\alpha, p, y$, ($\alpha$ is a primitive element of $GF(p)$)

Must find $x$ such that $y \equiv \alpha^x \pmod{p}$

✓ Let $x = \sum_{i=0}^{n-1} b_i 2^i$,

✓ Then, $b_0$ is determined by

$$y^{(p-1)/2} \pmod{p} = \begin{cases} +1, & \text{if } b_0 = 0 \\ -1, & \text{if } b_0 = 1 \end{cases}$$

(∵) Since $\alpha$ is primitive, $\alpha^{(p-1)/2} \equiv -1 \pmod{p}$

Therefore, $y^{(p-1)/2} = (\alpha^x)^{(p-1)/2} \equiv (\alpha^{(p-1)/2})^x \pmod{p}$
An Algorithm for $p = 2^n + 1$ (2)

- Now, $b_1$ is determined by letting

$$z \equiv y\alpha^{-b_0} \equiv \alpha^{x_1} \pmod{p}, \quad \text{where } x_1 = \sum_{i=1}^{n-1} b_i 2^i$$

- Then,

$$z^{(p-1)/4} \pmod{p} \equiv (\alpha^{x_1})^{(p-1)/4} \equiv (\alpha^{(p-1)/2})^{x_{1/2}} \equiv (-1)^{x_{1/2}}$$

$$\equiv \begin{cases} +1, & b_1 = 0 \\ -1, & b_1 = 1 \end{cases}$$

- Remaining bit of $x$

$$m \equiv \frac{p-1}{2^{i+1}}$$

$$z \equiv \alpha^{x_i} \pmod{p}, \quad \text{where } x_i = \sum_{j=i}^{n-1} b_j 2^j$$

$$z^m \pmod{p} \equiv \begin{cases} +1, & b_i = 0 \\ -1, & b_i = 1 \end{cases}$$
Flowchart for $p = 2^n + 1$

START

$Z \leftarrow Y$
$\beta \leftarrow \alpha^{-1} \pmod{p}$
$m \leftarrow (p-1)/2$
$i \leftarrow 0$

$W \leftarrow Z^m \pmod{p}$

TEST $W$

$b_i \leftarrow 0$

$\beta \leftarrow \beta^2 \pmod{p}$
$m \leftarrow m/2$
$i \leftarrow i+1$

$W \leftarrow Z^m \pmod{p}$

W=1

FALSE

W=-1

TRUE

HALT

i=n
An Algorithm for Arbitrary Primes (1)

✓ Generalize the algorithm to arbitrary primes $p$
  • $2^{16} + 1$ is the largest known prime of the form $2^n + 1$

✓ Let $p - 1 = p_1^{n_1} p_2^{n_2} \cdots p_k^{n_k}$, $p_i < p_{i+1}$
  where the $p_i$ are distinct primes and the $n_i$ are positive integers

✓ By Chinese Remainder Theorem,
  if the value of $x \pmod{p_i^{n_i}}$ is determined for all $i$, then
  $x \pmod{\prod_{i=1}^{k} p_i^{n_i}} = x \pmod{p - 1} = x$
Consider the following expansion of \( x \pmod{p_i^{n_i}} \)

\[
x \pmod{p_i^{n_i}} = \sum_{j=0}^{n_i-1} b_j p_i^j , \quad \text{where } 0 \leq b_j \leq p_i - 1
\]

Then,

\[
\gamma^{(p-1)/p_i} \equiv \alpha^{(p-1)x/p_i} \equiv \gamma_i^x \equiv \gamma_i^{b_0} \pmod{p}
\]

where \( \gamma_i = \alpha^{(p-1)/p_i} \)

\( \Rightarrow \) The resultant value uniquely determines \( b_0 \)
An Algorithm for Arbitrary Primes (3)

✓ The function $g_i(w)$ is defined by

$$
γ_i^{g_i(w)} \equiv w \pmod{p}, \quad 0 \leq g_i(w) \leq p_i - 1
$$

✓ The resultant value, $y^{(p_i-1)/p_i} \equiv γ_i^{b_i} \pmod{p}$ determines $b_i$ by $g_i(w)$

✓ So, dominant computational requirement: computing $g_i(w)$
Flowchart for arbitrary primes

START

Z ← Y
β ← α⁻¹ (mod p)
n ← (p-1)/p
j ← 0

W ← Z^n (mod p)
b_j ← g_i(W)
Z ← Z β^b_j (mod p)
β ← β^p_i (mod p)
N ← n/p_i
j ← j + 1

j = n_i

TRUE

HALT

FALSE
Time & Space Complexity

Theorem

Let

\[ p - 1 = p_1^{n_1} p_2^{n_2} \cdots p_k^{n_k}, \quad p_i < p_{i+1} \]

be the prime factorization of \( p - 1 \), where \( p \) is prime, the \( p_i \) are distinct primes, and the \( n_i \) are positive integers.

Then, for any \( \{r_i\}_{i=1}^k \) with all \( 0 \leq r_i \leq 1 \), logarithms over \( GF(p) \) can be computed in \( O \left( \sum_{i=1}^k n_i \left( \log_2 p + p_i^{1-r_i} (1 + \log_2 p_i^{r_i}) \right) \right) \) operations with \( O \left( \log_2 p \cdot \sum_{i=1}^k (1 + p_i^{r_i}) \right) \) bits of memory.

Proof. [1]
Discussion

✓ $p = 2^{448} \cdot 5^2 + 1$ is a prime that $p-1$ has only small prime factors (i.e., 2, 5)
  → $2 + 448 = 450$ iterations of the loop in the flowchart
  → Dominant computational requirement: 450 exponentiations $mod\ p$

✓ When $p = 2p' + 1$, (p' is also prime)
  (ex. $p' = 2^{121} \cdot 5^2 \cdot 7^2 \cdot 11^2 \cdot 13 \cdot 17 \cdot 19 \cdot 23 \cdot 29 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 47 \cdot 53 \cdot 59 + 1$)
  → Dominant computational requirement: $g(w)$
  → computing $g(w)$: more than $10^{30}$ operations & $10^{30}$ bits of memory ($r=1/2$)

Not one-way function

One-way function
References