Discrete Log

Discrete Logarithm

- Discrete log problem:
- Given p, g and g^a (mod p), determine a
  - This would break Diffie-Hellman and ElGamal
- Discrete log algorithms analogous to factoring, except no sieving
  - This makes discrete log harder to solve
  - Implies smaller numbers can be used for equivalent security, compared to factoring

Discrete Log Algorithms

- We discuss three methods
- **Trial multiplication**
  - Analogous to trial division for factoring
- **Baby-step giant-step**
  - TMTO for trial multiplication
- **Index calculus**
  - Analogous to Dixon's algorithm

Trial Multiplication

- The most obvious thing to do…
- We know p, g and $g^a \pmod{p}$
- To find a, compute

  $g^2 \pmod{p}$, $g^3 \pmod{p}$, $g^4 \pmod{p}$, …
- Until one matches $g^a \pmod{p}$
- Expected work is about p/2

Baby Step Giant Step

- Speed up to trial multiplication
- Again, know p, g and $x = g^a \pmod p$
  - We want to find exponent a
- First, let $m = \lceil \mathrm{sqrt}(p - 1) \rceil$
- Then $a = im + j$, some $i,j \in \{0,1,\ldots,m-1\}$
- How does this help? Next slide...

Baby Step Giant Step

- Have $x = g^a \pmod{p} = g^{im+j} \pmod{p}$
- Therefore, $g^j = xg^{-im} \pmod{p}$
- If we find i and j so that this holds, then we have found exponent a
  - Since $a = im + j$
- How to find such i and j ?

Baby Step Giant Step

- Algorithm: Given $x = g^a \pmod{p}$
- **Giant steps:** Compute and store in a table, $xg^{-im} \pmod{p}$ for $i = 0,1,\ldots m-1$
- **Baby steps:** Compute $g^j \pmod{p}$ for $j = 0,1,\ldots$ until a match with table — obtain $a = im + j$
- Expected work: sqrt(p) to compute table, sqrt(p)/2 to find j, for total of 1.5 sqrt(p)
- Storage: sqrt(p) required

Baby Step Giant Step Example

- Spse $g = 3$, $p = 101$ and $x = g^a \pmod{p} = 37$
- Then let $m = 10$ and compute giant steps:

giant step i	0	1	2	3	4	5	6	7	8	9
$3^{-10i} \pmod{101}$	1	14	95	17	36	100	87	6	84	65
$37 \cdot 3^{-10i} \pmod{101}$	37	13	81	23	19	64	88	20	78	82

- Next, compute $3^j \pmod{101}$ until match found with last row
- In this case, find $3^4 = 37 \cdot 3^{-20} \pmod{101}$
- And we have found $a = 24$

Index Calculus

- Given p, g, $x = g^a \pmod p$, determine a
- Analogous to Dixon's algorithm
  - Except linear algebra phase comes first
- Choose bound B and factor base
  - Suppose $p_0, p_1, \ldots, p_{n-1}$ are primes in factor base
- Precompute discrete logs: $\log_g p_i$ for each i
  - Can be done efficiently
  - Corresponds to linear algebra phase in Dixon's

Index Calculus

- Next, randomly select $k \in \{0,1,2,\ldots,p-2\}$ and compute $y = x \cdot g^k \pmod{p}$ until find y that factors completely over factor base
- Then $y = x \cdot g^k = p_0^{d_0} \cdot p_1^{d_1} \cdot p_2^{d_2} \cdots p_{n-1}^{d_{n-1}} \pmod{p}$
- Take $\log_g$ and simplify to obtain

$$a = \log_g x = (d_0 \log_g p_0 + d_0 \log_g p_0 + \ldots + d_0 \log_g p_0 - k) \pmod{(p-1)}$$

- And we have determined a
  - Note p – 1 follows from Fermat's Little Thm

Index Calculus Example

- Spse: $g = 3$, $p = 101$, $x = 3^a = 94 \pmod{p}$
- We choose factor base 2,3,5,7
- Compute discrete logs: $\log_3 2 = 29$, $\log_3 3 = 1$, $\log_3 5 = 96$, $\log_3 7 = 61$
- Select random k, compute $y = x \cdot g^k \pmod{p}$ until y factors over factor base
- For $k = 10$, find $y = 50 = 2 \cdot 5^2 \bmod (101)$

Index Calculus Example

- For $k = 10$, have $y = 50 = 2 \cdot 5^2 \bmod (101)$
- Then

  $a = (\log_3 2 + 2 \log_3 5 - 10) \pmod{100}$

  $= 29 + 2 \cdot 96 - 10 = 11 \pmod{100}$
- Easy to verify $3^{11} = 94 \pmod{101}$
- Work is same as Dixon's algorithm
  - In particular, work is subexponential

Conclusions

- Many parallels between factoring and discrete log algorithms
  - For example, Dixon's and index calculus
- For discrete log, not able to sieve
  - Therefore, no analog of quadratic sieve
- For elliptic curve cryptosystems (ECC)
  - No analog of Dixon's or index calculus…
  - …since no concept of a factor base
  - So ECC is secure with smaller parameters