FUJITSU

2010年6月11日（金）15:15 – 15:45
「暗号理論の安全性を支える数論アルゴリズムとその応用」
グローバルCOEプログラム　マス・フォア・インダストリ主催

On the security evaluation of elliptic curve cryptography

Masaya Yasuda
(FUJITSU LABORATORIES LTD.)

shaping tomorrow with you

九州大学　伊都キャンパス　数理学研究教育棟B1　大会議室111

Background（1）

- Elliptic Curve Cryptography (ECC)
  - In 1985, N. Koblitz and V. Miller independently proposed using elliptic curves to design public-key cryptographic schemes.
    - Ex. elliptic curve-based signature, public-key encryption, etc…
  - Advantages are the higher cryptographic strength per bit in comparison with RSA and the higher speed in implementations.

- The security of ECC ≒ The hardness of the ECDLP
  - ECDLP = Elliptic Curve Discrete Logarithm Problem
    - This is somewhat mathematical problem.
  - No efficient algorithm for the ECDLP is known except special cases.
    - Special cases = supersingular case, anomalous case, etc…

Background（2）

- Experimental reports of solving the ECDLP
  - Certicom ECC Challenge (1997~)

Table: Status of the Certicom ECC Challenge

Year	ECC2 (in bits)	ECC2K (in bits)	ECCp (in bits)
1997	79		79
1998	97	95	97
1999			
2000		108	
2001			
2002			109
2003			
2004	109		
・・・			
2010		130?	

ECC2 = Elliptic curves over binary fields
ECC2K = Koblitz curves
ECCp = Elliptic curves over prime fields

10,000 PCs for 18months

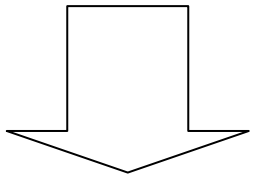
2,600 PCs for 17months

In progress (October 2009 ~)
http://ecc-challenge.info/

※112-bit prime ECDLP solved (not in Certicom ECC Challenge) ← current record
200 PS3 for 6 months (January 2009 ~ July 2009)

Our motivation

- The problem of past experimental reports to evaluate the security of ECC:
  - Due to variance calculation by volunteers all over the world, detail data for solving the ECDLP is uncertain.
  - Past experiments were not implemented under an union environment.



We cannot evaluate the security of ECC accurately from past experimental reports.

Our work

- Evaluate the security of ECC
  - We extract detail data for solving the ECDLP from experiments under an union environment.
  - We evaluate the security strength balance between types of elliptic curves.
    - Types of elliptic curves: ECC2, ECC2K, ECCp.
- Compare the security strength between ECC and RSA
  - Using our past data of the security of RSA
    - In the past, we evaluated the security of RSA with our factoring device.

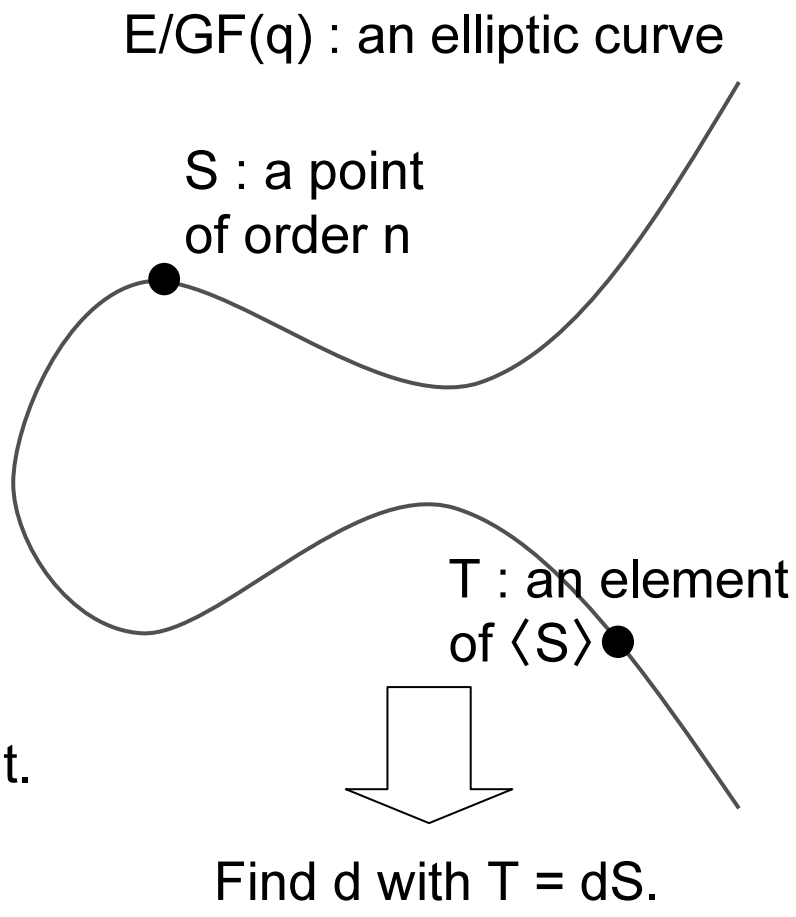
Our factoring device

In the next page, we explain our work in detail.

ECDLP and Attacks

ECDLP

- (The hardness of the ECDLP) ≒ (The security of ECC)
- ECDLP over GF(q):
  - Given (E, q, n, S, T)
  - Find the integer d with T = dS
- Remark
  - For a large prime number n (160bit), we cannot solve the ECDLP in realistic time at present.



※We fix this notation.

Pollard's rho method（1）

- The strongest known attack for the ECDLP (generic curves) is Pollard's rho method.
- The basic idea:
  - Search for the two distinct pairs (c_i, d_i), (c_j, d_j) with

$$c_i \cdot S + d_i \cdot T = c_j \cdot S + d_j \cdot T$$

  - Then we obtain the solution d with T = dS:

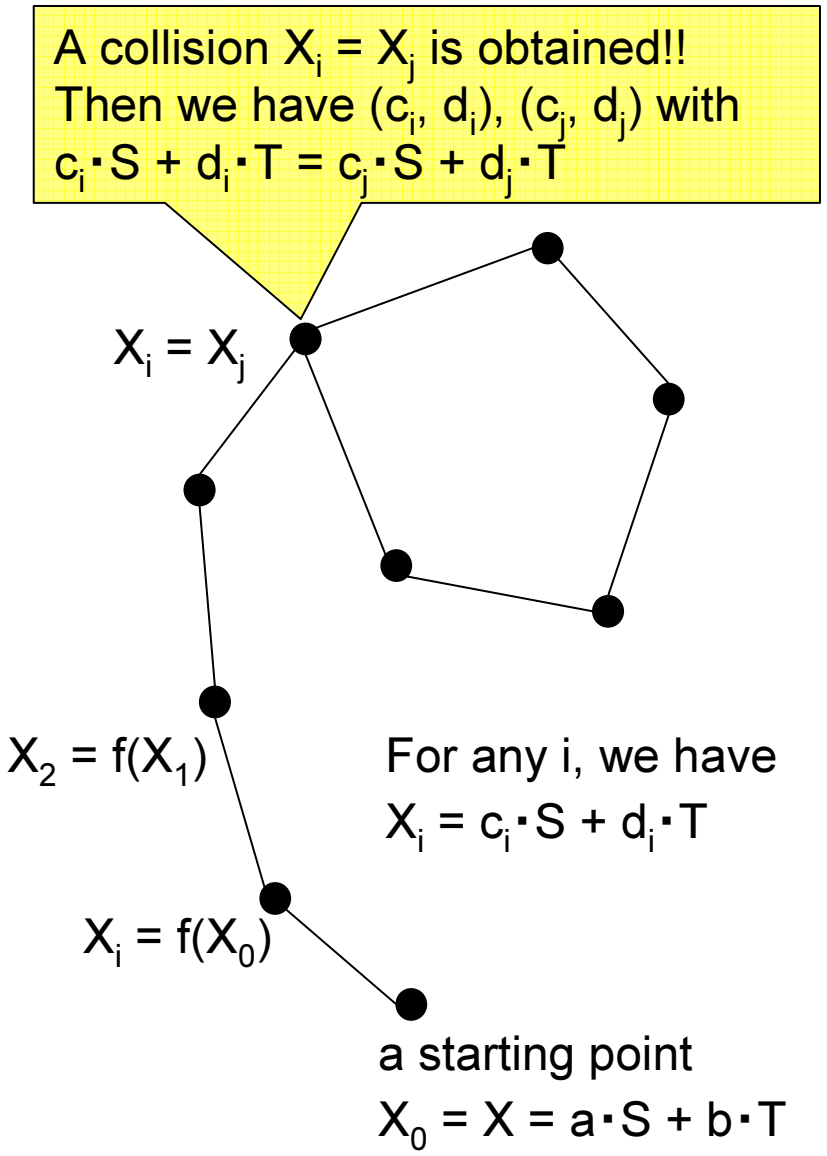
$$(c_i - c_j) \cdot S = (d_j - d_i) \cdot T = (d_j - d_i) \cdot dS$$

$$d = (c_i - c_j) \cdot (d_j - d_i)^{-1} \bmod n$$

We explain the method to search (c_i, d_i), (c_j, d_j) in the next page.

Pollard's rho method (2)

- Choose an iteration function f: ⟨S⟩→ ⟨S⟩
  - It is easy compute a, b with f(X) = a·S + b·T
  - f : quasi-random
- For X = a·S + b·T, define X_{i+1} = f(X_i) with X_0 = X.
  - Then we can compute c_i, d_i with $X_i = c_i \cdot S + d_i \cdot T$
- Improvement:
  - Parallel version
  - Distinguished points
    - It is a collision detection technique



Remarks on Pollard's rho method

- The running time = #(iterations before a collision is obtained) × t(f)
  - t(f) = the running time of an iteration function f
  - #(iterations before a collision is obtained) ~ $(\pi n/2)^{1/2}$ if f : random
    - It is heavily dependent on the choice of an iteration function f. (see below for example)

Table: Performance of iteration functions on elliptic curves over prime fields [Te]

Iteration functions	f_P	f_{PG}	$f_{TA[20]}$	$f_{TM[16:4]}$
Av. of iterations / $(\pi n/2)^{1/2}$	1.28	1.29	1.03	1.04

f_P : Pollard's original iteration function
f_{PG} : Pollard's generalized iteration function
$f_{TA[20]}$: Teske's L-addiding walk with L =20
$f_{TM[16:4]}$: Teske's mixed-walk with 16 mult. and 4 sqr.

These iteration functions are suitable for solving the ECDLP.

[Te] E. Teske, "On random walks for Pollard's rho method", Math. Comp. 70 (2001).

How about on Koblitz curves?

Pollard's rho method for the ECDLP on Koblitz curves

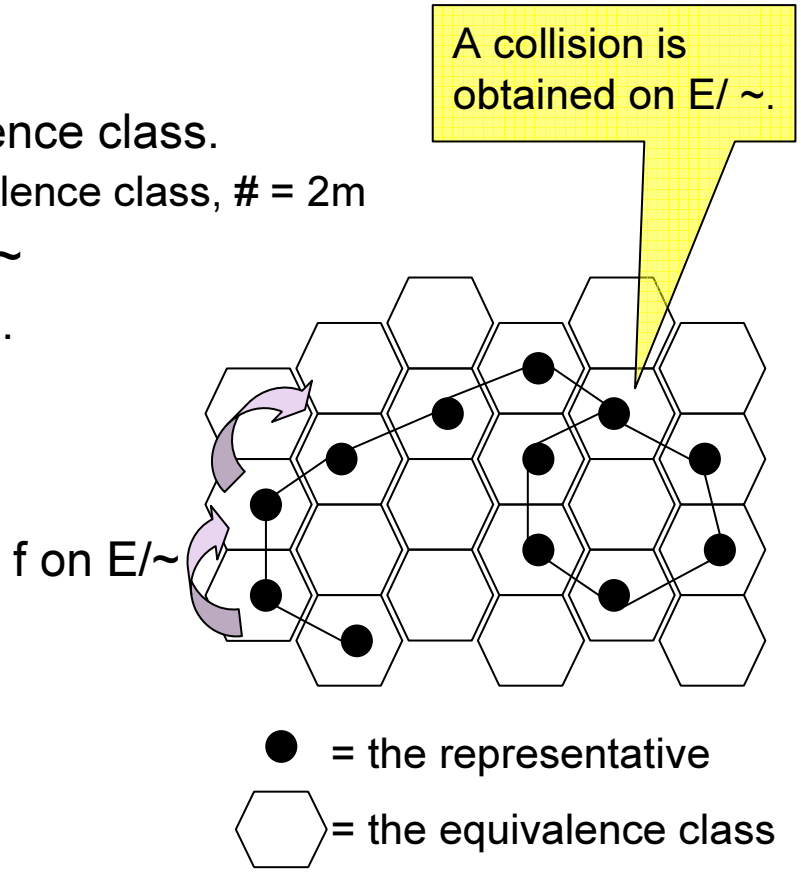
Review on Koblitz curves

- Koblitz curves were first suggested for use in cryptography by Koblitz.
  - The defining equation E: $y^2 + xy = x^3 + ax^2 + b$
    - a, b : elements of GF(2) with $b \neq 0$.
  - The advantage of these curves is that point multiplication algorithms can be devised.

- Frobenius map ϕ : $E(GF(2^m)) \rightarrow E(GF(2^m))$
  - ϕ : $(x, y) \rightarrow (x^2, y^2)$
    - It is a group homomorphism of order m
    - It can be efficiently computed since squaring in $GF(2^m)$ is relatively inexpensive.

Speeding Pollard's rho method for Koblitz curves

- Using the Frobenius map ϕ, the rho method for Koblitz curves can be sped up.
- The basic idea:
  - Define an equivalence relation:
    - $P \sim Q \Leftrightarrow P = \pm \phi^j(Q)$ for some j.
    - [P] = the representative of the equivalence class.
      - $\{\pm P, \pm \phi(P), \pm \phi^2(P), \cdots\}$: equivalence class, # = 2m
  - Consider an iteration function f on E / ~
    - E/ ~ : the set of the representatives [P].
- Expected #(iterations):
  - $1/2 \cdot (\pi n/m)^{1/2}$ if f: random on E / ~
    - Speed up by $(2m)^{1/2}$
    - #(E/~) = n/(2m)



What is a suitable f on E / ~ ?

Our iteration function on E/~

- We define an iteration function on E/~:
  - j = $\mathrm{hash}_m(L(P))$, L : labeling function, $0 \leqq s \leqq m$.
  - For s = 0, our iteration function is the same as that proposed by [GLV].

$$f_s(P) = \begin{cases} 2P & \text{if } j < s \\ P + \phi^j(P) & \text{otherwise} \end{cases}$$

- Properties:
  - It is a well-defined map on E/~.
  - We can compute f_s with high speed as the parameter s becomes large.
    - A point doubling on elliptic curves is in general faster than a point addition.

[GLV] R. Gallant, R. Lambert and S. Vanstone, “Improving the Paralleized Pollard Lambda Search on Binary Anomalous Curves”, Mathematics of Computation 69, pp. 1045-1062 (2000).

Performance of our iteraiton function on Koblitz curves (Experimental investigation)

Description of experiments

- To analyze the performance of our iteration function, we attacked the ECDLP on Koblitz curves.
  - Parallelized Pollard's rho method with 10 processors.
    - Distinguished points (collision detection)
  - We attacked the ECDLP on Koblitz curves of relatively small parameters for 100 times with randomly chosen starting points.
    - The parameters are ECC2K-41, 53, 83, 89. (Koblitz curves over GF(2^m) with m = 41, 53, 83, 89)

Our union environment for attacking the ECDLP

Experimental results

- We summarize the performance of f_s with s=0, m/5, m/3, m/2.

Table: Performance of our iteration function f_s.

Our iteration function f_s	s=0	s=m/5	s = m/3	s = m/2
ECC2K-41	1.06	1.14	1.29	1.17
ECC2K-53	1.10	0.96	1.12	1.26
ECC2K-83	1.03	0.99	1.25	1.16
ECC2K-89	1.01	1.20	1.08	1.29
Av. of iterations / Exp.	1.05	1.07	1.18	1.22

Exp. = 1/2 $(\pi n/m)^{1/2}$: the expected number of iterations before a collision is obtained.

- Investigation:
  - f_s with s = 0 is suitable for solving the ECDLP on Koblitz curves.
    - f_s with s= 0 has a performance almost same as the random function on E/~

Our results on the security of ECC

Our estimation of computating power required to break ECC（1）

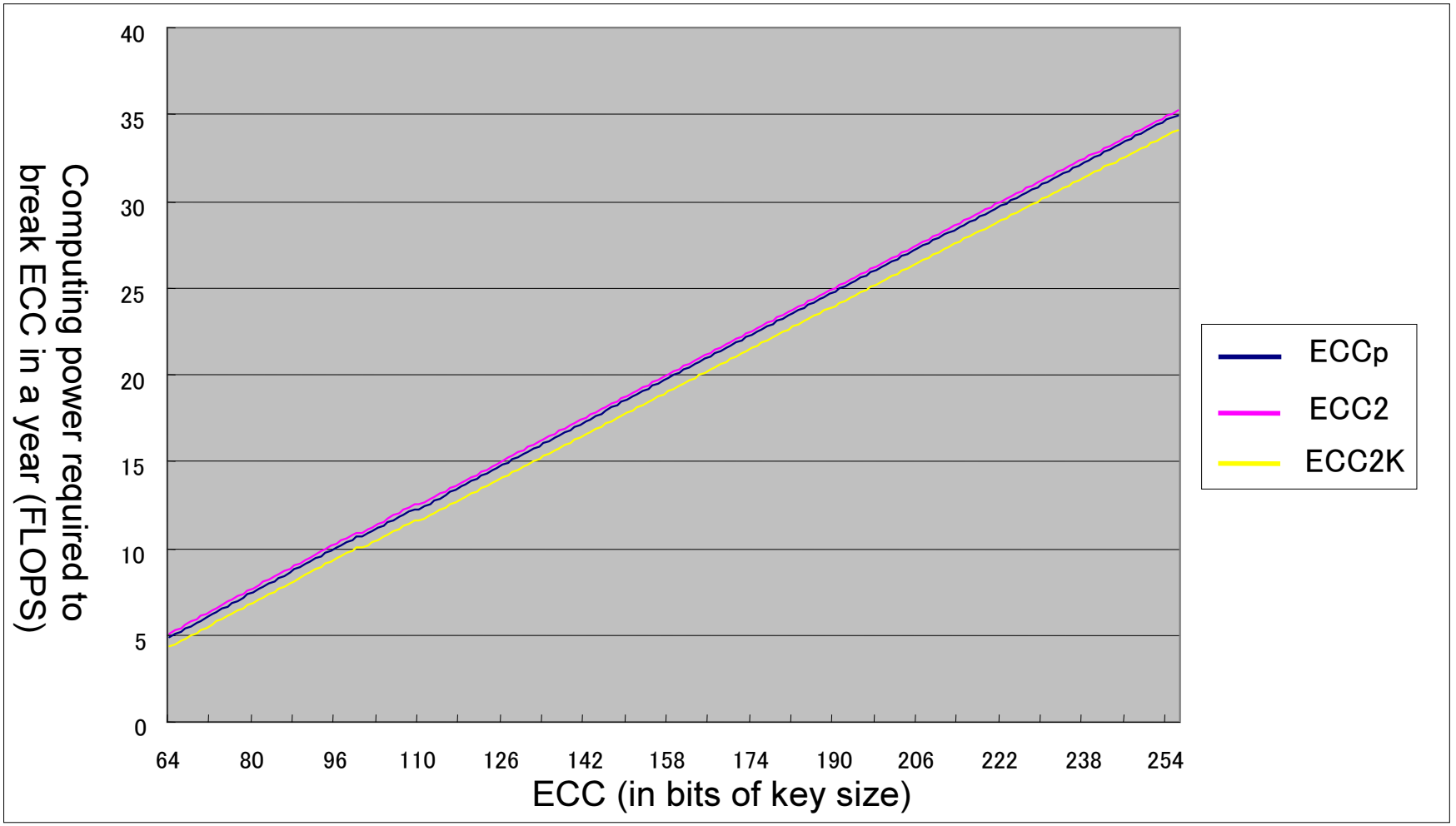
- We estimate computing power required to break ECC in a year (FLOPS) using Pollard's rho method.
  - Review: the running time = #(iterations) × t(f)
    - t(f) : the computational speed of an iteration function f.
- The method of our estimation:
  - For #(iterations), we use our experimental results under our union environment.
    - In this talk, we only explain the performance on Koblitz curves.
  - For t(f), we use the latest data:
    - ECCp : 1772cycle / iteration (224bit) [1]
    - ECC2 : 1047cycle / iteration (131bit) [2]

[1] Bernstein, "Curves25519 : new Diffie-Hellman speed records", PKC 2006.

[2] Bailley, et al. "The Certicom Challenge ECC2-X", SHARCS 2009.

Our estimation of computating power required to break ECC（2）

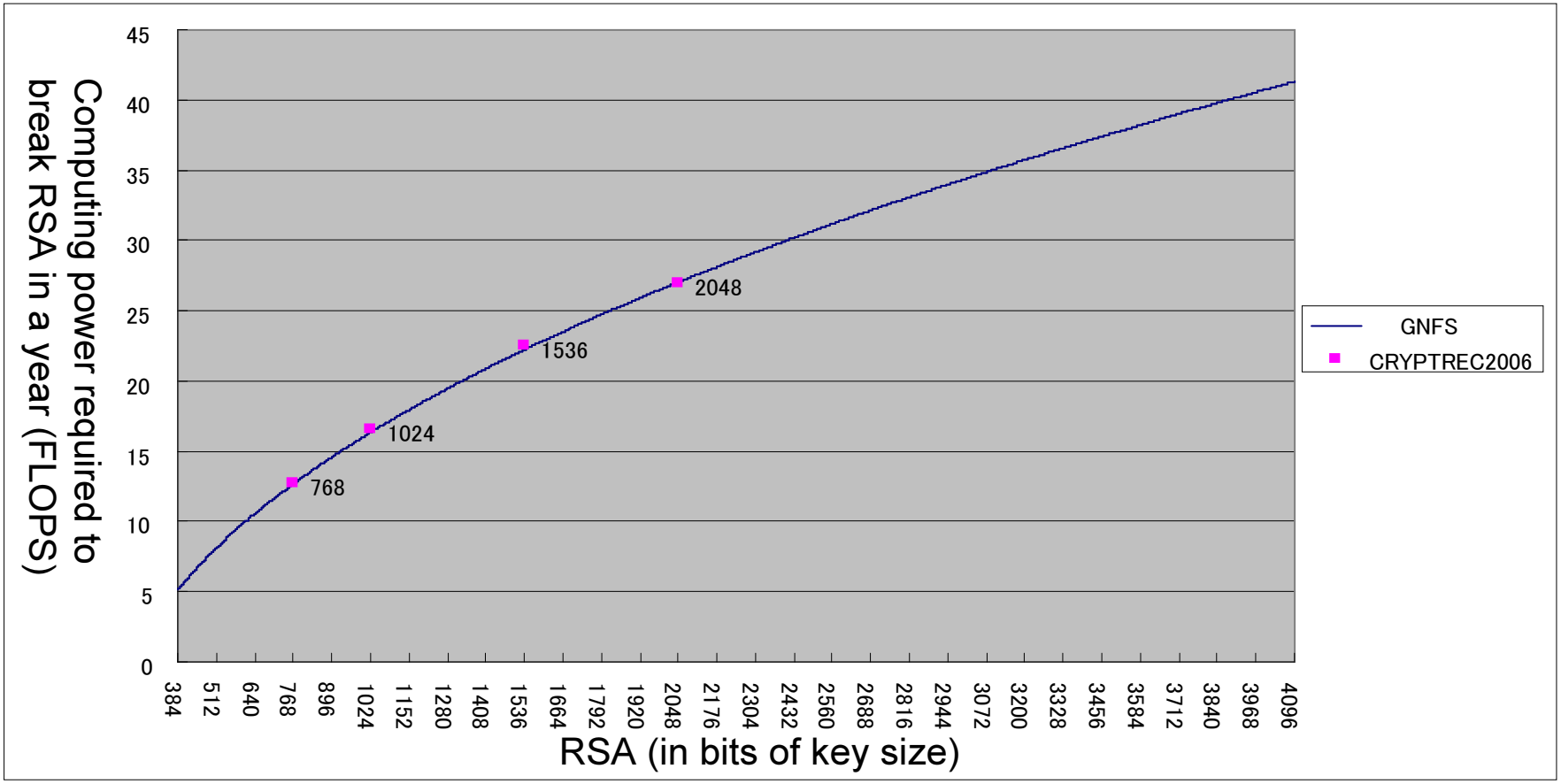
Table: Computing power required to break ECC in a year using Pollard's rho method. (FLOPS, exponent of 10)



Our estimation of computing power required to break RSA (our previous work)

- In the past, we evaluated the security of RSA cryptosystem by breaking RSA with our factoring devise.

Table: Computing power required to break RSA in a year using GNFS (FLOPS, exponent of 10)



Comparision of the security strength between ECC and RSA

RSA	ECCp	ECC2	ECC2K
696	106	105	110
768	114	112	117
850	122	120	125
1024	138	137	142
1219	152	151	156
1536	177	175	181
2048	206	204	210
2206	214	213	219
2832	245	244	250
6281	371	370	376
11393	497	496	503

in bits of key size

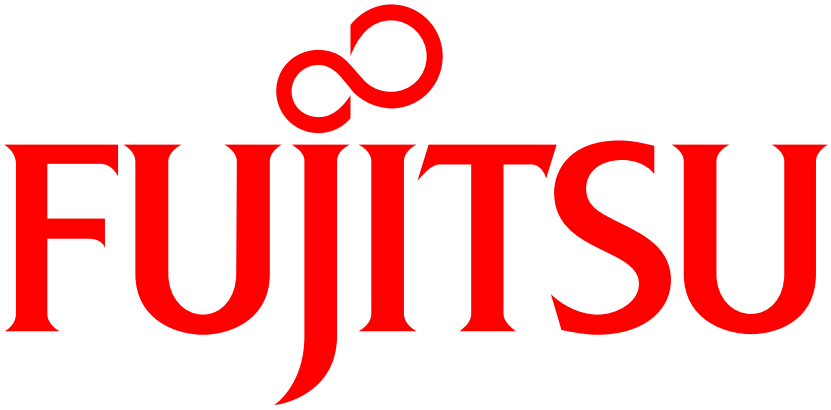
Current breakable security level !!
- RSA-768 solved (2010)
- ECCp-112 solved (2009)

Our estimation:
RSA-1024
≒ECCp-138, ECC2-137, ECC2K-142

Conclusion

Conclusion

- We extracted detail data for solving the ECDLP from experiments under our union environment.
  - Parallelized Pollard's rho method with 10 processors
  - We analyzed the performance of many iteration functions on elliptic curves
    - In this talk, we only explained the performance of our iteration function on Koblitz curves.
- We evaluated the security of ECC and compared the security strength between ECC and RSA.
  - The security balance between types of elliptic curves.
  - RSA-1024 ≒ ECCp-138, ECC2-137, ECC2K-142
    - cf. RSA-1024 ≒ ECC-160~223 (NIST SP800-57)



shaping tomorrow with you