SHARCS 2012
(Special-Purpose Hardware for Attacking Cryptographic Systems)
Date: 17-18 March 2012
Place: Washington, DC, USA

FUJITSU
shaping tomorrow with you

On the strength comparison of ECC and RSA

Masaya Yasuda, Takeshi Shimoyama, Tetsuya Izu, Jun Kogure

(FUJITSU LABORATORIES LTD.)

A part of this research is financially supported by the National Institute of Information and Communications Technology (NICT), Japan.

Outline

- Background and our motivation
- The hardness of the ECDLP
  - ECDLP = Elliptic Curve Discrete Logarithm Problem
- The hardness of the IFP
  - IFP = Integer Factorization Problem
- Comparison of the hardness of the ECDLP and the IFP
- Conclusions and future work

Background and our motivation (1/2)

FUJITSU

Table: The comparison of security strengths of ECC and RSA with 80-bit security

Report	ECC	RSA
NIST	160	1024
Lenstra-Verheul	160	1300
RSA Labs.	160	760
NESSIE	160	1536
IETF	-	1228
ECRYPT II	160	1248

- There are a lot of previous works that analyze the security of ECC and RSA. But the comparison of strengths varies depending on analysis.

⇩

Our motivation is once again to compare the security strengths, considering state-of-the-art of theory and experiments.

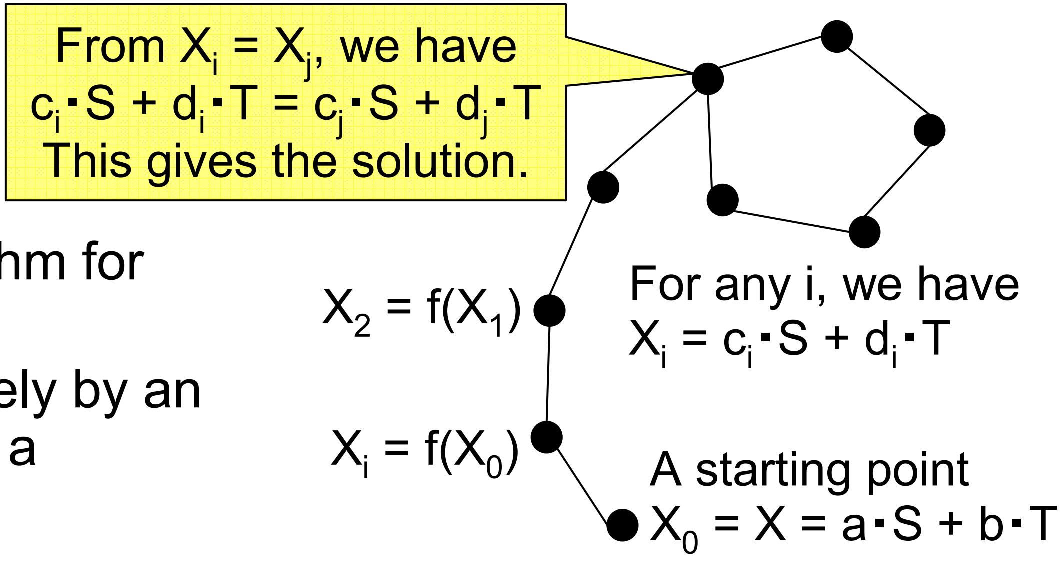
Background and our motivation（2/2）

- To compare the strengths of ECC and RSA, we estimate the computing power required to solve the ECDLP and the IFP in a year, respectively.
  - The security of ECC and RSA ≒ the hardness of the ECDLP and the IFP, respectively.

- In this talk, we focus on the hardness of the ECDLP and the IFP only from the view point of software implementation.
  - Using special-purpose hardware for solving the ECDLP or the IFP is a theme of great interest at this conference.
  - But these platforms and architectures vary, and it is difficult to make an analysis on the cost performance. ⇒ This is a future work.

[ECDLP] Pollard's rho method

- **Definition of the ECDLP:**
  - Given an elliptic curve E/GF(q) and two points S, T of large prime order n, find k with T = kS.



From $X_i = X_j$, we have
$c_i \cdot S + d_i \cdot T = c_j \cdot S + d_j \cdot T$
This gives the solution.
$X_2 = f(X_1)$
For any i, we have
$X_i = c_i \cdot S + d_i \cdot T$
$X_i = f(X_0)$
A starting point
$X_0 = X = a \cdot S + b \cdot T$

- **Pollard's rho method:**
  - It is the fastest known algorithm for solving the ECDLP.
  - First generate $\{X_i\}$ successively by an iteration function f. Then find a collision $X_i = X_j$.

- **Our policy to estimate the complexity of the rho method:**
  - Since the rho method is probabilistic, we estimate the complexity required to solve the ECDLP **with 99% probability** in order to unify the success probability of solving the IFP
  - Furthermore, we focus on three types in the ECDLP:
    - prime fields, binary fields, and Koblitz curves types.

Suitable iterations for the rho method (1/2)

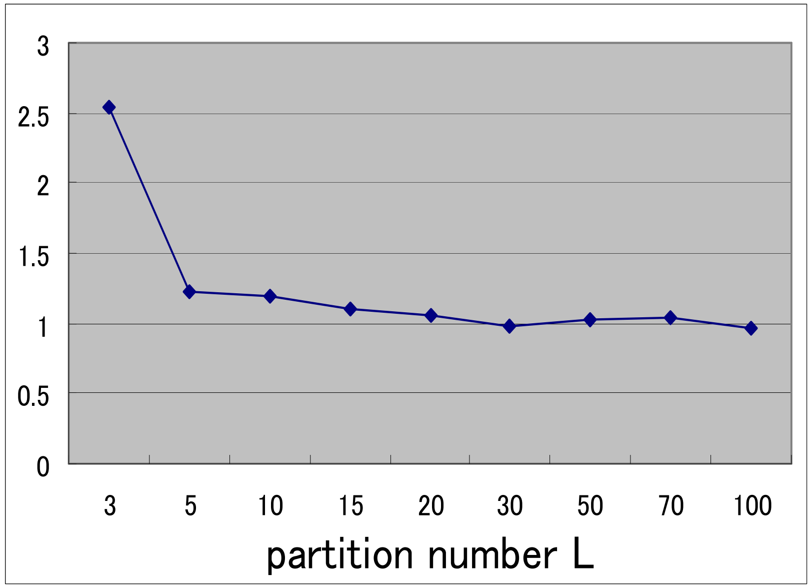
- **Choice of iterations**
  - The complexity heavily depends on choice of iteration functions.
    - An iteration function f is the most suitable if f is a random walk.
  - To analyze the randomness of f, we define

δ := the average of #{ iterations with f } / Exp.

('Exp' = the expected number of iterations with random walks)

If δ is close to 1, f has the enough randomness.

- **Prime and binary fields cases:**
  - Teske's L-adding walk: $f_{TA}(X) = X + M_j$.
    - M_i: pre-computed points ($i = 1, \ldots, L$).
    - L : partition number, which affects the randomness.
  - **From experiments, f_{TA} with $L \geqq 20$ has enough randomness <u>on average</u>.**



Our experimental result on δ with f_{TA}

Suitable iterations for the rho method (2/2)

- **Koblitz curves case:**
  - Gallant et al.'s function: $f_{GLV}([X]) = [g(X)]$, $g(X) = X + \phi^j(X)$
    - ϕ: the Frobenius map, $j = \mathrm{hash}_m([X])$
    - $[X]$: the equivalent class containing X
  - This function is well-defined on E/~ and suitable for the speed-up with ϕ and the negation map.
    - E/~: the set of the equivalent classes.
  - Our experimental result on δ with f_{GLV} on Koblitz curves $E(GF(2^m))$:

	m = 41	m =53	m = 83	m = 89
δ	1.06	1.10	1.03	1.01

  - **From experiments, f_{GLV} has enough randomness <u>on average</u>.**

Our method to estimate the complexity

FUJITSU

To estimate the complexity, we fix f_{TA} with L≧20 and f_{GLV}, which have enough randomness on average.

- The complexity of the rho method is determined by

  ① the number of iterations, and

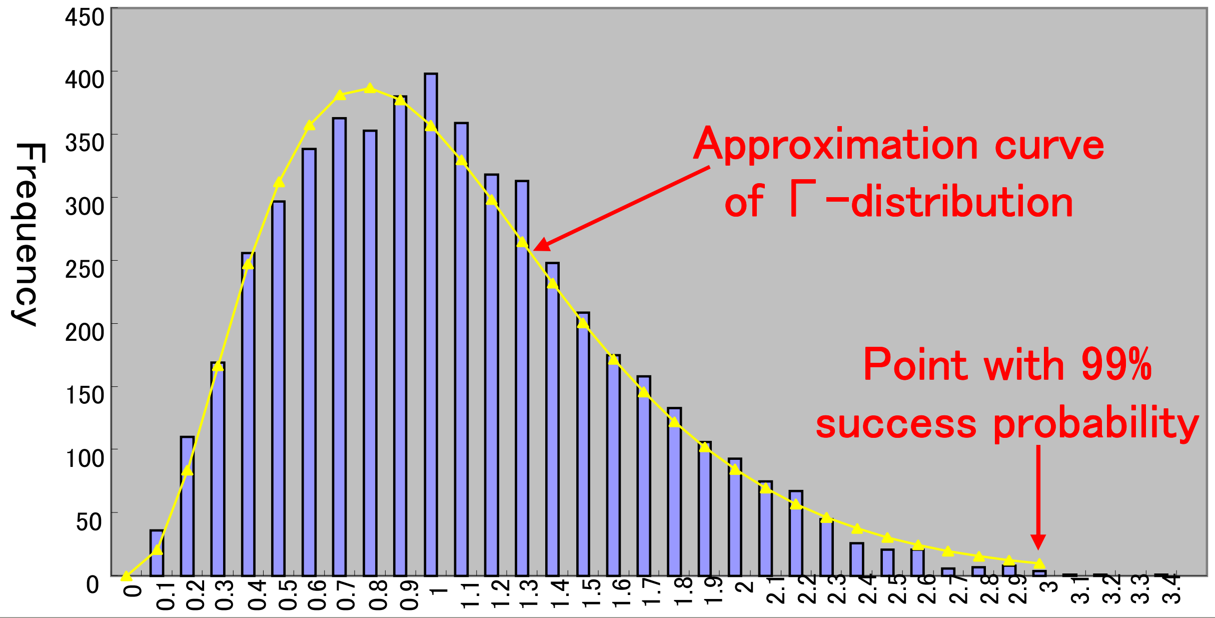
  We estimate the number of iterations required to solve the ECDLP with 99% probability, by considering the distribution of the frequencies.

  ② the processing performance of iterations.

  We estimate the processing performance of iterations by using the previously known best records.

① The number of iterations

- **Distribution of the frequencies of the number of iterations:**
  - We solved the ECDLP over prime and binary fields of 40-bit for 10,000 times. We also used f_{TA} with L = 20.
    - We did not use the speed-up with the negation map.
  - As the numbers of iterations can be modeled as waiting times, it is reasonable to approximate the graph by Γ-distribution.
  - **With this approximation, we estimate that we can solve the ECDLP with 99% probability if we compute iterations by 3 times of the expected number.**



※The Horizontal axis shows the number of iterations (set 1 as the expected number)

② The processing performance of iterations

- We use the latest data:
  - Prime fields: 306.08 cycles / iteration (112-bit) [1]
  - Binary fields: 1047 cycles / iteration (131-bit) [2]
  - Koblitz curves: 1.62 × 1047 cycles / iteration (131-bit)
    - 1.62 : overhead of transforming bases estimated by our implementation.

- t(f): the processing performance on an elliptic curve of N-bit.
- **We assume that t(f) scales proportionally to $N^{1.585}$, due to the Karatsuba method.**

$$t(f) = \begin{cases} 306.08 \times (N/112)^{1.585} \ (prime\ \ fields\ case), \\ 1047 \times (N/131)^{1.585} \ (binary\ \ fields\ case), \\ 1.62 \times 1047 \times (N/131)^{1.585} \ (Koblitz\ \ curves\ case). \end{cases}$$

[1] D. Bernstein, T. Lange and P. Schwabe, “On the Correct Use of the Negation map in the Pollard rho Method”, PKC 2011.
[2] D. Bailey et al., “The Certicom Challenge ECC2-X”, SHARCS 2009.

Estimation formula

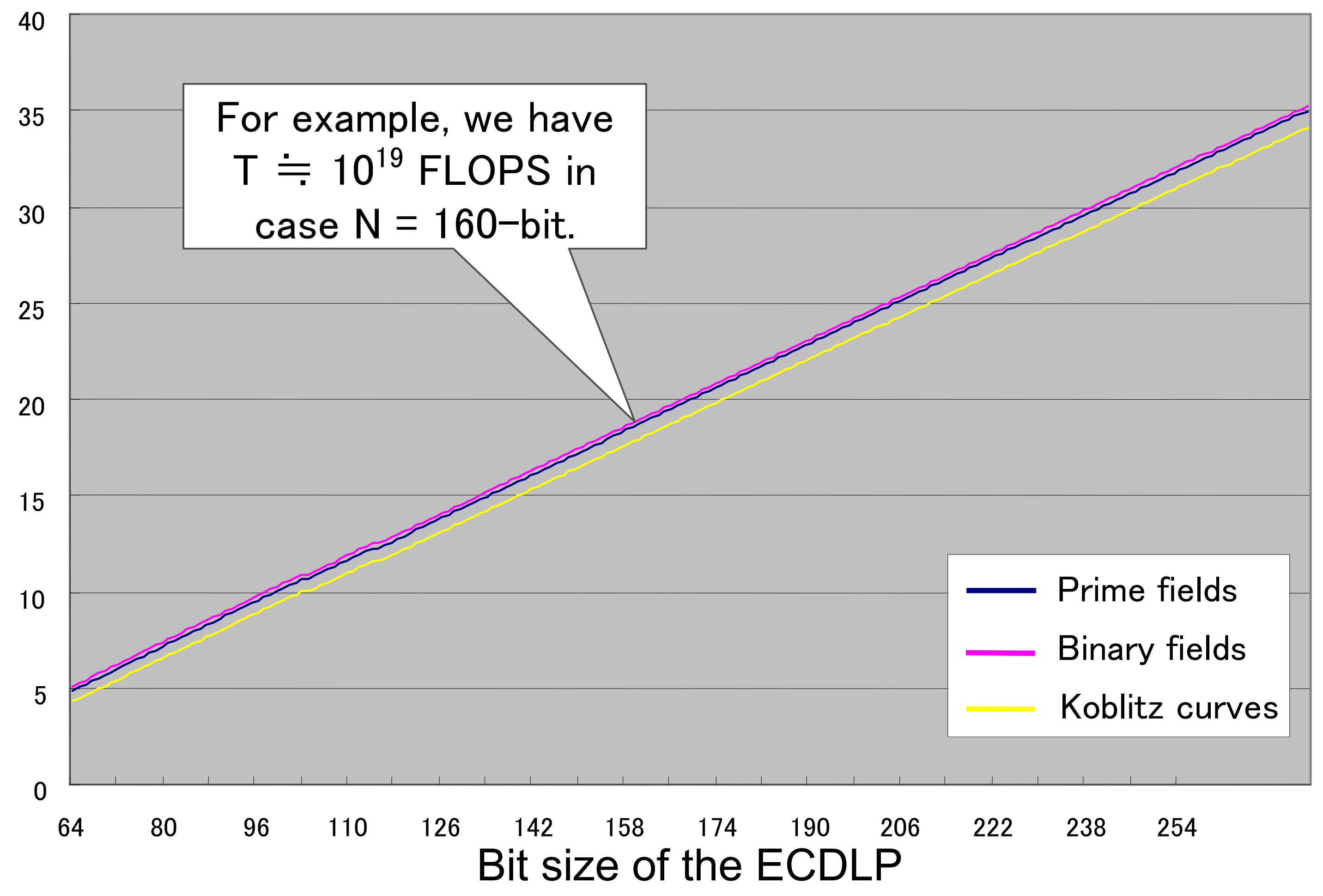
The complexity of the rho method = #{iterations} × t(f).

- **Estimation formula of the complexity of the rho method:**
  - T = the computing power to solve the ECDLP of N-bit with 99% probability
    - FLOPS is the unit of T.
    - Consider 2^N as the order of the point S. Set Y = 365･24･60･60 (seconds)

$$T = \begin{cases} 3 \cdot \sqrt{\pi 2^N} / 2 \times 306.08 \times (N/112)^{1.585} / Y \ (\mathit{prime\ \ fields\ case}), \\ 3 \cdot \sqrt{\pi 2^N} / 2 \times 1047 \times (N/131)^{1.585} / Y \ (\mathit{binary\ \ fields\ case}), \\ \underbrace{3 \cdot \sqrt{\pi 2^N / N} / 2}_{\text{The number of iterations with 99\% success probability. Assume the efficiency of the speed-up with automorphisms.}} \times \underbrace{1.62 \times 1047 \times (N/131)^{1.585}}_{\text{The processing performance of iterations. Assume that t(f) scales proportionally to } N^{1.585}.} / Y \ (\mathit{Koblitz\ curves\ case}). \end{cases}$$

Computing power to solve the ECDLP

- By our estimation formula, we estimate the computing power required to solve the ECDLP of N-bit in a year as follows:
  - The vertical axis shows $\log_{10}(T)$.



[IFP] GNFS

- GNFS (= general number field sieve)
  - It is the most efficient known algorithm for solving the IFP of large composite integers.
  - Heuristically, the complexity of the GNFS for factoring N is given by

$$L_N\left(\frac{1}{3}, \sqrt[3]{\frac{64}{9}}\right) \quad as \;\; N \to \infty$$

where

$$L_N(s,c) = \exp((c+o(1))(\log N)^s(\log\log N)^{1-s}).$$

- Our method to estimate the complexity of the GNFS:

① We analyze the o(1)-value of $L_N(s, c)$ experimentally.

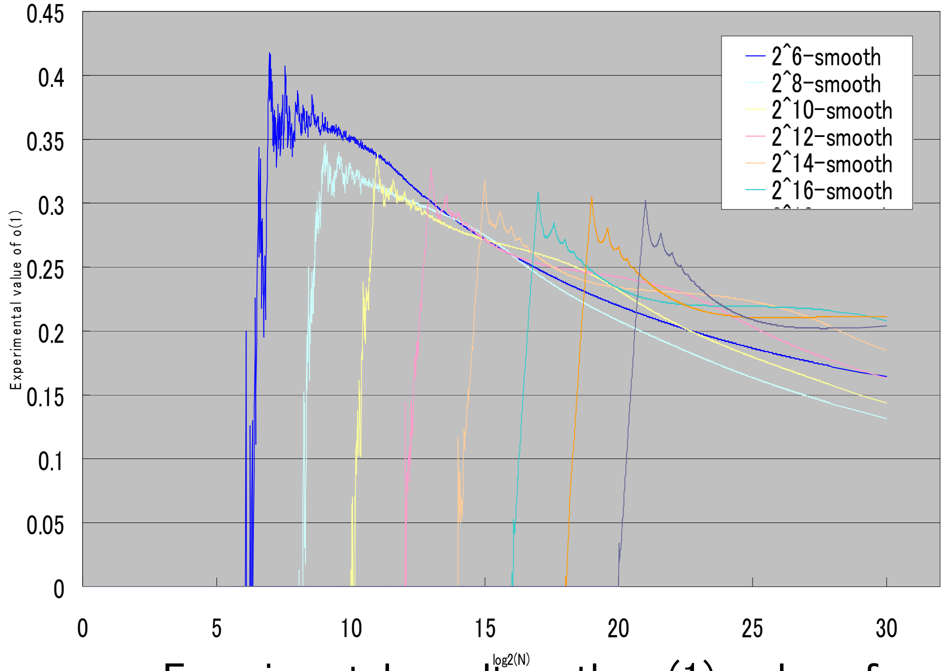
② By using the previously known result, we set

- the o(1)-value of $L_N(s, c)$, and
- the leading coefficient ℓ of $L_N(s, c)$.

① Analysis of the o(1)-value of $L_N(s, c)$

- The o(1)-value of $L_N(s,c)$ is closely related to the o(1)-value of Φ(x, z).
  - For two integers x and z, let Φ(x, z) denote the probability that an arbitrary integer in [1, x] is z-smooth.
  - It is known that we have $\Phi(x,z) = (\log_z x)^{(-1+o(1))\log_z x}$ *as* $x \to \infty$.

- Experimental investigation:
  - We expect that the o(1)-value of Φ(x, z) is included in [0, 0.2] as x→∞.
  - Since Φ(x, z) is used twice for deriving $L_N(s, c)$, we expect that **<u>the o(1)-value of $L_N(s,c)$ is included in [0, 0.4] as x→∞.</u>**



Experimental result on the o(1)-value of Φ(x, z) with x≦32-bit.

② Setting of o(1) and ℓ

FUJITSU

Previously known results:

- The result in [3] gives the computing power required to solve the IFP of N-bit in a year as follows:
  - Assume that the memory requirement is 2G Bytes.
  - T = the computing power (FLOPS is the unit of T)

N	768	1024	1536	2048
$\log_{10} T$	12.6879	16.3395	22.2319	27.0413

o(1) and ℓ:

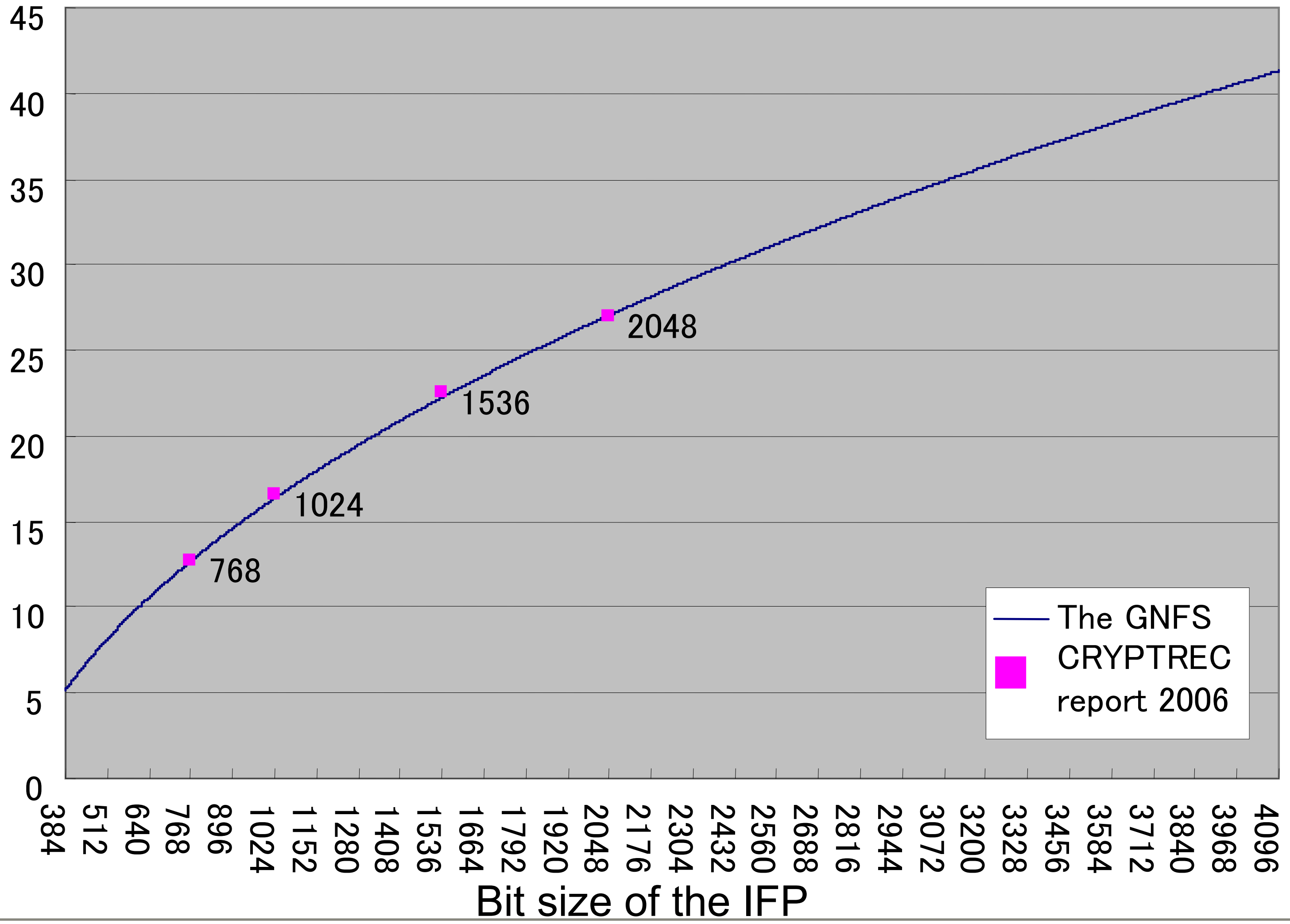
- The complexity of the GNFS is very close to each data of the above result if we set
  - o(1) = 0.348172, and（Note that o(1) is in [0, 0.4]）
  - $\ell = 1.0373 \times 10^7$ as the leading coefficient of $L_N(s, c)$.
- **Estimation formula**：

$$T = \ell \cdot L_N\left(\frac{1}{3}, \sqrt[3]{\frac{64}{9}}\right)$$ with o(1) and ℓ as above

[3] CRYPTREC Report 2006, available at http://www.cryptrec.go.jp/report/c06_wat_final.pdf.

Computing power to solve the IFP

- By our estimation formula, we estimate the computing power required to solve the IFP of N-bit in a year as follows:
  - The vertical axis shows $\log_{10}(T)$.



The ECDLP vs the IFP (1/2)

- By comparing the computing power required to solve the ECDLP and the IFP, we estimate the bit sizes that provide the same level security:

Bit size of the IFP	Bit size of the ECDLP		
	Prime fields	Binary fields	Koblitz curves
512	86	84	88
696	107	105	110
768	115	112	117
851	123	120	125
1024	138	136	141
1219	154	151	157
1536	176	173	179
2048	207	205	211
2839	247	245	251

※If we use ordinary multiplication for elliptic operations, the ECDLP bit length can be reduced at most two bits.

The ECDLP vs the IFP (2/2)

- In particular, we have
  - **768-bit IFP ≒ 115-bit ECDLP over prime field, 112-bit over binary field, or 117-bit on Koblitz curves.**
    - The world records of 2011 for solving RSA and ECC are 768-bit RSA in 2010 [4] and 112-bit ECC over prime field in 2009 [5], respectively.
    - Since the times of these two records are close, we consider that these records indicate reasonability of our estimation.
  - **1024-bit IFP ≒ 138-bit ECDLP over prime field, 136-bit over binary field, or 141-bit on Koblitz cuves.**
    - Although it is often said that 1024-bit RSA corresponds to 160-bit ECC, our estimation indicates that shorter ECC key sizes provide the same level of security.

[4] T. Kleinjung et al. "Factorization of a 768-bit RSA modulus", CRYPTO 2010.
[5] EPFL IC LACAL, "PlayStation 3 computing breaks 260 barrier 112-bit prime ECDLP solved", http://lacal.epfl.ch/112bit_prime.

Conclusions and future work

- **Conclusions:**
  - ECDLP (the rho method):
    - By considering the distribution of the frequencies, we estimate #{iterations} with 99% success probability.
    - We also estimate t(f) by using the previously known best records.
  - IFP (the GNFS):
    - To estimate the complexity of the GNFS, we determine o(1) and ℓ of $L_N(s, c)$ by using our analysis of Φ(x, z) and CRYPTREC result [3].
  - Our estimation showed that 1024-bit IFP ≒ 140-bit ECDLP.
    - If we say 160-bit ECC has 80-bit security, our estimation indicates that 1024-bit RSA does not reach this security.
- **Future work:**
  - We need to analyze the complexity of the rho method with the iteration function proposed by [6], which seems to be the best known function on Koblitz curves.
  - Furthermore, we need to check if the processing performance of iterations scales proportionally to $N^{1.585}$.

[6] D. Bailey et al., "Breaking ECC2K-130", http://eprint.iacr.org/2009/541.pdf.

FUJITSU

shaping tomorrow with you