CONJUGATION AS PUBLIC KEY AGREEMENT PROTOCOL IN MOBILE CRYPTOGRAPHY

Vittorio Ottaviani
Dipartimento di Informatica Sistemi e Produzione, Università di Roma Tor Vergata, Via del Politecnico 1, 00133 Rome, Italy

Alberto Zanoni, Massimo Regoli
Centro "Vito Volterra", Università di Roma Tor Vergata, Via Columbia 2, 00133 Rome, Italy

Keywords: Key agreement, Conjugation, Mobile Device.

Abstract: We analyze a key agreement algorithm realization, not using Diffie-Hellman approach, but using matrix powers and conjugation. Introduced in a theoretical frame by Sakalauskas et. al. in 2007, it is here implemented in J2ME on mobile devices (Nokia N70 equipped with Symbian S60 operating system). We study its applicability and performances and compare them with Elliptic Curve and standard Diffie-Hellman Bouncy Castle implementation, freely available on the web.

1 INTRODUCTION

Mobile devices diffusion increasing looks like a never ending story; according to RBC analysts previsions (Snol, 2009a), 2011 shipments of smartphone devices will approach 400 M units equalizing PC sales. Figure 1(a) resumes the trend 2005–2011, "A" indicates actual values, "E" indicates estimated values.

Between vendors NOKIA still owns the biggest part of the mobile device market, but other vendors, e.g., Apple and RIM, are gaining considerable part of the customers; Figure 1(b) shows the 2008 smartphone's market situation (Snol, 2009b).

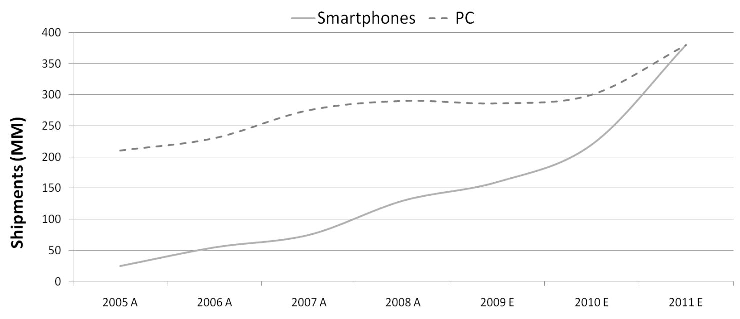
Smartphone OS's diffusion, obviously, reflects the smartphone device diffusion; Symbian leads the market with 52% followed by RIM 17%, Windows Mobile 12%, iPhone 8%, Palm 2%, Android 1% and others 9% (Dignan, 2009).

As more and more powerful smartphones and the like are becoming everyday matter, many software applications are appearing on the market. Multimedia management, games, navigators, chat programs and message exchange are only some examples of the possibilities that these devices can be used for.

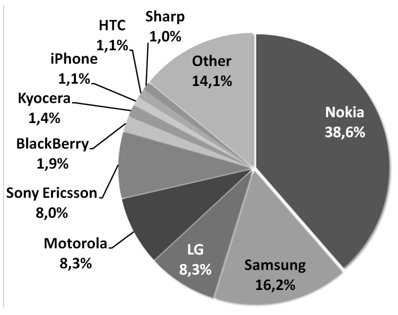
For more delicate matters (e.g., mobile banking, payments and other operations that require privacy and security for the user, like mobile voting systems), cryptographic protocols and algorithms enter the game. In particular, in many cases a key agreement is needed to send/exchange private data/information by coding them with a specific algorithm. Some mobile cryptography use examples are (Ahmad et al., 2009), in which elliptic curves are efficiently used, and (Grillo et al., 2008a), (Grillo et al., 2008b), concerning trusted text messaging. All these works focus more on coding/signing part than on key agreement, but of course a key agreement phase is needed before encrypting or signing.

In this paper we present a JavaME implementation of a new key agreement protocol – a particular case of a class recently proposed in (Sakalauskas et al., 2007) – and compare our implementation performance (Ottaviani et al., 2009) against standard and Elliptic Curve Diffie-Hellman protocol (Diffie and Hellman, 1976).

In the next section we explain the mathematical problem to be solved to exploit the key agreement, and some consideration upon possible attacks and why these attacks are not effective on such algorithm. In section 3 the implementation choices are presented, analyzing why they do not affect security, optimizing performances. In section 4 we analyze the testing methodology explaining each step of the testing phase. Section 5 shows the testing phase results. Section 6 analyzes with more detail the section 5 data. Section 7 resumes all results proposing possible improvements and applications of the algorithm.



(a)



(b)

Figure 1: (a) Smartphone and PC sales prevision, (b) 2008 mobile handset market share.

2 MATHEMATICAL SETTING: KEY AGREEMENT PROTOCOL

We consider $GL(d,\mathbb{Z}_p) = \mathcal{M}$, where p is a prime number. Fix $G \in \mathcal{M}$ and let φ be the conjugation isomorphism associated to G

$$\varphi_G : \mathcal{M} \ni M \mapsto \varphi_G(M) = GMG^{-1} \in \mathcal{M}$$

The following public key agreement between Alice (**A**) and Bob (**B**) – see (Sakalauskas et al., 2007) for a more general setting – exploits the property $[\varphi_G(A)]^n = \varphi_G(A^n)$.

1. **A** and **B** share $Q, S \in \mathcal{M}$, with $SQ \neq QS$ and $\det(Q) = |Q| = 1$,
2. **A** chooses two numbers $x_A, n_A \in \mathbb{N}$.
3. **A** computes $M_A = S^{n_A} Q^{x_A} S^{-n_A}$ and sends it to **B**.
4. **B** receives from A the matrices M_A.
5. **B** chooses two numbers $x_B, n_B \in \mathbb{N}$, computes $M_B = S^{n_B} Q^{x_B} S^{-n_B}$ and sends M_B to **A**.
6. **A** computes $M_{AB} = S^{n_A} M_B^{x_A} S^{-n_A} = S^{n_A}(S^{n_B} Q^{x_B x_A} S^{-n_B}) S^{-n_A}$
7. **B** computes $M_{BA} = S^{n_B} M_A^{x_B} S^{-n_B} = S^{n_B}(S^{n_A} Q^{x_A x_B} S^{-n_A}) S^{-n_B}$

At the end **A** and **B** share the common matrix $M_{AB} = M_{BA}$, which represents the Secret Shared Key (SSK). Infact,

$$\begin{aligned} M_{AB} &= S^{n_A+n_B} Q^{x_B x_A} S^{-(n_A+n_B)} \\ &= S^{n_B+n_A} Q^{x_A x_B} S^{-(n_B+n_A)} = S^{n_B} M_A^{x_B} S^{-n_B} = M_{BA} \end{aligned}$$

Note that if $|Q| \neq 1$, a possible eavesdropper Eve (**E**) could set up a discrete logarithm problem by considering the determinantal equation (Bodrato, 2009)

$$\begin{aligned} |M_A| &= |S^{n_A} Q^{x_A} S^{-n_A}| = |S^{n_A}||Q^{x_A}||S^{-n_A}| \\ &= |S|^{n_A} |Q|^{x_A} |S|^{-n_A} = |Q|^{x_A} \end{aligned}$$

with $\det(Q)$ known, if **E** can solve this scalar discrete logarithm problem, thus recovering x_A, then she can easily find, by solving a linear problem, and adjusting the free parameters entering in the solution, a polynomial X in the matrix S of degree $\leq d$, with coefficients in $\mathbb{Z}_p$ such that $M_A X = X Q^{x_A}$.

Using this, **E** can compute

$$\begin{aligned} X M_B^{x_A} X^{-1} &= (X S^{n_B} X^{-1})(X Q^{x_A} X^{-1})^{x_B}(X S^{-n_B} X^{-1}) \\ &= S^{n_B} M_A^{x_B} S^{-n_B} = M_{AB} \end{aligned}$$

because X commutes with S. In conclusion: if $\det(Q) \neq 1$, then, the breaking complexity of the algorithm is essentially equivalent to the breaking complexity of a (discrete) logarithm in $\mathbb{Z}_p$, i.e., to that of (scalar) Diffie-Hellman. With $\det(Q) = 1$ (see step 1 of agreement process), this “attack” cannot be performed.

Figure 2 shows the agreement process performed by the algorithm. **E** could intercept S, Q, d, p, M_A and M_B. In order to recover the private keys (e.g., n_A and x_A), she could set up the following equation

$$M_A = S^{n_A} Q^{x_A} S^{-n_A} = (S^{n_A} Q S^{-n_A})^{x_A}$$

but this is much more difficult than a usual matrix discrete logarithm problem (DLP), as the base matrix is unknown. Other identities, such as

$$M_A S^{n_A} = S^{n_A} Q^{x_A}$$

are difficult to exploit because both S^{n_A} and Q^{x_A} are not known separately.

We have that $^{\#}\mathcal{M} = \prod_{i=0}^{d-1}(p^d - p^i)$. Let $o(M)$ be the order of a matrix $M \in \mathcal{M}$, i.e., the smallest integer such that $M^{o(M)} = 1$. In order to avoid useless computations, it is sufficient to choose $n_A, n_B < o(S)$ (resp. $x_A, x_B < o(Q)$).

The order of a matrix $M \in \mathcal{M}$ is in general difficult to compute, but an upper bound for it can be found as follows. For each $M \in \mathcal{M}$ let $p_M(x) = \prod_{i=1}^{k} f_i(x)^{d_i}$ be its characteristic polynomial factorized in $\mathbb{Z}[x]$, with

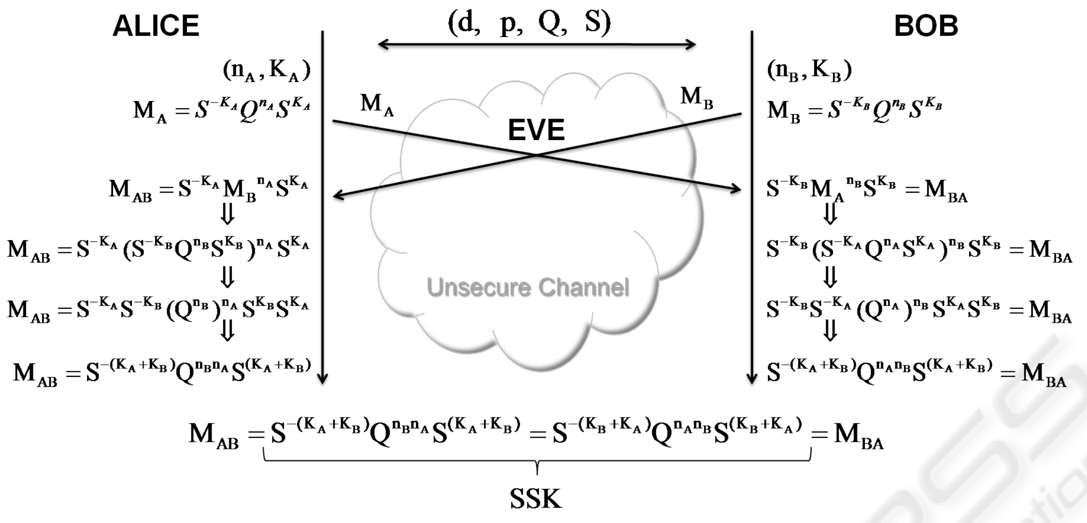

ALICE
(d, p, Q, S)
BOB
(n_A, K_A)
$M_A = S^{-K_A}Q^{n_A}S^{K_A}$
M_A
EVE
M_B
(n_B, K_B)
$M_B = S^{-K_B}Q^{n_B}S^{K_B}$
$M_{AB} = S^{-K_A}M_B^{n_A}S^{K_A}$
$S^{-K_B}M_A^{n_B}S^{K_B} = M_{BA}$
$M_{AB} = S^{-K_A}(S^{-K_B}Q^{n_B}S^{K_B})^{n_A}S^{K_A}$
$S^{-K_B}(S^{-K_A}Q^{n_A}S^{K_A})^{n_B}S^{K_B} = M_{BA}$
Unsecure Channel
$M_{AB} = S^{-K_A}S^{-K_B}(Q^{n_B})^{n_A}S^{K_B}S^{K_A}$
$S^{-K_B}S^{-K_A}(Q^{n_A})^{n_B}S^{K_A}S^{K_B} = M_{BA}$
$M_{AB} = S^{-(K_A+K_B)}Q^{n_Bn_A}S^{(K_A+K_B)}$
$S^{-(K_A+K_B)}Q^{n_An_B}S^{(K_A+K_B)} = M_{BA}$
$M_{AB} = \underbrace{S^{-(K_A+K_B)}Q^{n_Bn_A}S^{(K_A+K_B)} = S^{-(K_B+K_A)}Q^{n_An_B}S^{(K_B+K_A)}}_{SSK} = M_{BA}$

Figure 2: Key Agreement process using conjugate.

$\alpha = \max\{d_i \mid i = 1, \dots k\}$. An upper bound (multiple) $m(M)$ for its multiplicative order $o(M)$ is given by the following formula (Celler and Leedham-Green, 1995)

$$m(M) = \operatorname{lcm}(p^{d_1} - 1, \dots, p^{d_k} - 1) \cdot p^{\lceil \log_p(\alpha) \rceil}$$

3 J2ME IMPLEMENTATION

The previously described operations to perform key agreement have been developed in Java Micro Edition (J2ME). We chose to implement in such programming language because we need a suite that can run on different hardware platforms and operating systems. Moreover we noticed that a good performance evaluation can be obtained, comparing our implementation of the key agreement algorithm with Bouncy Castle's implementation of Elliptic Curve and standard Diffie-Hellman key agreement algorithm.

Bouncy castle provide a plethora of API performing different cryptographic operations implemented in JAVA, J2ME and C#, we used the Elliptic Curve Diffie-Hellmen (ECDH) and the standard Diffie-Hellman (DH) key agreement J2ME implementation to perform the comparison. The first step to implement the algorithm described in section 2, is to implement the modular operations on matrices (e.g., modular matrix multiplication, power, inversion, conjugate and other ancillary operations).

It is very important, in a mobile environment, to optimize every step of every operation with respect to resource consumption: in small capacity devices every waste of resources implies a delay, larger than the delay, in more performing devices corresponding to the same waste: because of the shortage of RAM, CPU and storage capacity, operations need to be optimized as much as possible.

To perform the operations described in section 2 we use a 32 bit unsigned integer data structure. Unfortunately in JAVA and J2ME there is no unsigned integer data structure; to solve this problem there are two possible approaches:

1. use bigger data structures, such as 64 bit *signed long integer* simulating a 32 bit size applying modulus when the value exceeds 2^{32},
2. use available 32 bit *signed integer* combining it with arithmetical operations modulus 2^{31}.

We have chosen the latter solution, i.e., to develop the modular matrix as a integer array (`int[]`) with modulo 2^{31}. This data structure is, in our opinion, the best compromise between RAM wasting and CPU usage due to operations needed to perform a task. Security of the key agreement is not affected using 31 bit integers, while performances are compromised, if one uses the 64 bit signed integer to simulate 32 bit unsigned integer. Using long integers the RAM consumption doubles and the system's performances, in our opinion, degrade too much to justify the slight improvement in security.

4 PERFORMANCE TESTING METHODOLOGY

In this section we report our performance tests of Matrix Conjugation Based Key Agreement versus Elliptic Curve and standard Diffie-Hellman on a Nokia N70 platform.

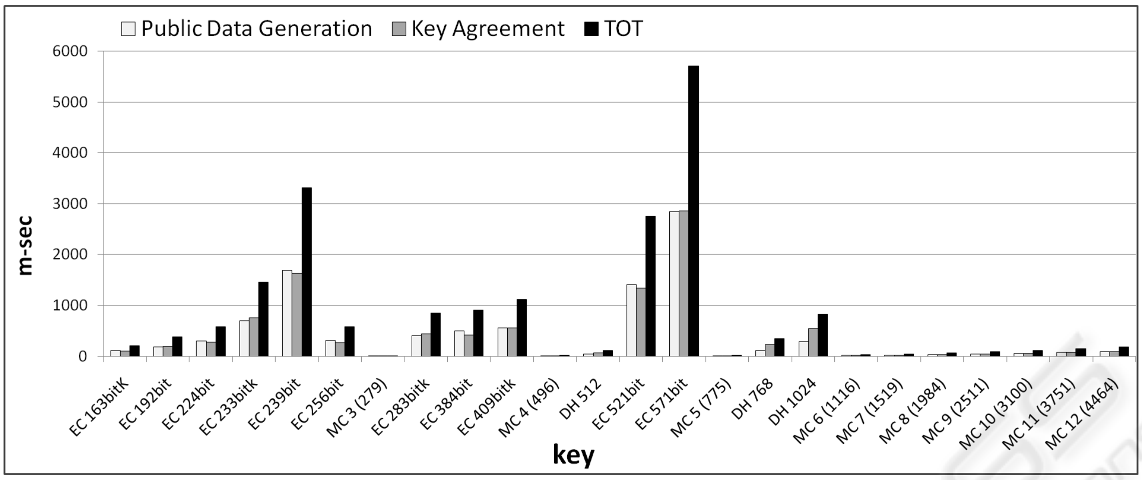


Figure 3: Public data and Key Agreement generation time: all tests. EC . . . bit: Elliptic Curve Diffie-Hellman with a . . . bit key. EC . . . bitK: Koblitz Elliptic Curve Diffie-Hellman with a . . . bit key. MC d (. . .): Matrix Conjugation at dimension d with a . . . bit key. DH . . . : Diffie-Hellman with a . . . bit key

The NOKIA N70 is a multimedia smartphone launched in Q3 2005. In 2007, it was the second most popular cellular phone, with 8% of all sales at Rampal Cellular Stockmarket(PRNewswire, 2007). Our experiments show similar results with other mobile devices. NOKIA N70 is equipped with:

- **CPU** : Texas Instruments OMAP 1710 (ARM architecture 926TEJ v5) 220 MHz processor
- **RAM** : 55 MB
- **FLASH** : 19.9 MB
- **MMC** : 2 GB
- **SCREEN** : 176×208 TFT Matrix, 256K colours
- **BATTERY** : BL-5C (970 mAh)
- **OS** : BB5 / Symbian OS v8.1a, S60 Platform Second Edition, Feature Pack 3 operating system
- **JAVA** : MIDP 2.0 midlets

In a mobile device, in general, and using J2ME, in particular, there are several problems in measuring the time required for a given task, because the accuracy of the `System.currentTimeMillis()` function is not sufficient.

We will use, as an estimate of the time length of a given task, the avarage of the time lengths, measured on several repetitions of the same task. More precisely:

Definition 1. *Let n be the number of iterations of one task, and let θ_i denote the time needed to perform the i^{th} task measured using the* `System.currentTimeMillis()`. *The actual time that the device needs to perform such task will be measured as follows:*

$$\Theta_n = \frac{1}{n}\sum_{i=1}^{n}\theta_i \qquad (1)$$

It is an empirical fact that Θ_n becomes approximately independent from n, for "large" n. The size on n depends on the task is and usually smaller for longer tasks (i.e., larger Θ_i) , see Table 1 below.

For each algorithm tested, we performed the above described operation for the most used instances of the algorithms; e.g., for the ECDH case we tested all the curves recommended by the NIST (NIST, 1999). For what concerns standard Diffie-Helman and Matrix Conjugation Based Key Agreement analysis, we considered instances with comparable private key length, in order to have an idea of brute force attack complexity with respect to performances.

Next section shows the experimental results of the comparison of various performances of different key agreement algorithms.

5 PERFORMANCE EVALUATION

Here we show the results of all the tests performed on standard key agreement algorithms and protocols and on Matrix Conjugation Based Key Agreement.

We compared the performance of Matrix Conjugation Based Key Agreement to other reference algorithms, such as Diffie-Hellman key agreement (DH) (Rescorla, 1999) and Elliptic Curve Diffie-Hellman key agreement (ECDH) (Barker et al., 2007). We remark that these algorithms are the most used to perform key agreement operations in desktop and mobile environments. Among the NIST suggested Elliptic Curves (Research, 2000), we select both Koblitz

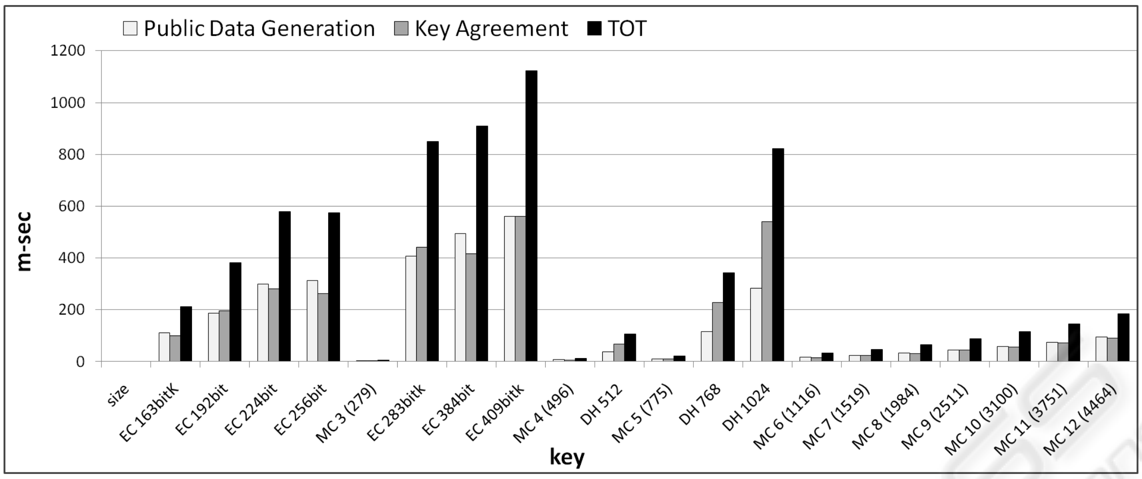


Figure 4: Public data and Key Agreement generation time: results with an upper bound of 1 sec. EC ... bit: Elliptic Curve Diffie-Hellman with a ... bit key. EC ... bitK: Koblitz Elliptic Curve Diffie-Hellman with a ... bit key. MC d (...): Matrix Conjugation at dimension d with a ... bit key. DH ...: Diffie-Hellman with a ... bit key.

curves (ending with a K in Figure 3 and Figure 4) and pseudo-random curves over $GF(p)$.

In Figure 3 the time comparison between Matrix Conjugation Based Key Agreement (MC in Figure 3 and Figure 4), standard and Elliptic Curve Diffie-Hellman is shown. We can note that conjugation based key agreement generates the public data and the SSK faster than the other algorithms. Since in Figure 3 the difference in generation times for the secret and the key agreement is not really significant, we illustrate in Figure 4 a closer look to show better the differences in time.

While a key agreement using Elliptic Curve with a 571-bits key takes 5706.3 milliseconds, a key agreement using conjugation based key agreement with a 5×5 matrix (775-bits key) takes only 20.63 milliseconds. This difference is significant even considering that the SSK generated by Matrix Conjugation Based Key Agreement is 50% larger than the Elliptic Curve SSK. Even when considering the case of standard Diffie-Hellman, the differences in mobile environment look quite impressive; for example, a Diffie-Hellman 768-bits SSK is agreed in 343.44 milliseconds while a Matrix Conjugation Based Key Agreement 775-bits SSK takes only 20.63 milliseconds. These differences are illustrated in Figure 4.

6 EXPERIMENTAL RESULTS

Table 1 summarizes all the results obtained in the performance testing for the different classes of algorithms. *Parameters* field indicates:

- In the ECDH case, the type of curve that is used to generate the agreement (K indicates a Koblitz curve) and the size of the generated SSK;
- In the DH case, the size of the generated SSK;
- In the Matrix Conjugation Based Key Agreement, the matrix dimension and the bit size of the matrix generated as key.

Public Data Generation (Pub. Data) field indicates the time to generate the exchanged data to agree a SSK. The field *Key Agreement* (Key Agr.) shows time needed to generate the SSK by means of exchanged and private data.

In *Total* field the sum of times used to generate exchanged data and SSK is shown. The last field, *Iterations* (Iter.), indicates how many times the agreement has been performed. This field is useful to understand the accuracy of the values in the *Public Data Generation*, *Key Agreement* and *Total* fields. In all cases but ECDH we did 100 iterations; in ECDH cases we decided to use just 10 iterations because times were more than one order of magnitude bigger than in the other cases, so that keeping the same accuracy was not necessary.

7 CONCLUSIONS

In this paper we compared a custom key agreement algorithm based on matrix conjugation with standard Diffie-Hellman and Elliptic Curve Diffie-Hellman key agreement. Our experiments have been performed using one of the most popular smartphone in the world. Experimental results showed that the key agreement based on matrix conjugation results to

Table 1: Time used from algorithms to generate the secret to agree a SSK.

Param.	Pub. Data	Key Agr.	Total	Iter.
Elliptic Curve Diffie-Hellman				
163bit K	110,90	100,00	210,90	10
192bit	185,90	195,30	381,20	10
224bit	298,50	281,20	579,70	10
233bit K	696,90	759,30	1456,20	10
239bit	1684,40	1626,60	3311,00	10
256bit	312,50	262,50	575,00	10
283bit K	407,80	442,20	850,00	10
384bit	493,70	415,70	909,40	10
409bit K	561,00	560,90	1121,90	10
521bit	1404,60	1342,20	2746,80	10
571bit	2845,30	2861,00	5706,30	10
Diffie-Hellman				
512	37,51	68,58	106,09	100
768	116,25	227,19	343,44	100
1024	282,98	539,83	822,81	100
Matrix Conjugation Based Key Agreement				
3 (279)	3,27	3,13	6,40	100
4 (496)	6,72	5,94	12,66	100
5 (775)	10,32	10,31	20,63	100
6 (1116)	16,72	15,47	32,19	100
7 (1519)	23,76	22,96	46,72	100
8 (1984)	33,90	31,41	65,31	100
9 (2511)	44,35	44,85	89,20	100
10 (3100)	57,97	56,87	114,84	100
11 (3751)	74,21	71,26	145,47	100
12 (4464)	93,91	89,53	183,44	100

be from 8 to 450 times faster than the two DH.

Providing the users new services on their mobile device enlarges the need of security to protect the information exchanged; such information can contain data about bank accounts, credit card numbers, pins or simply passwords.

Currently existing cryptographic methods affect too much usability of applications, charging the system with resource consumption due to cryptographic operations. Considering the growing business opportunity around the mobile world and, at the same time, the need of new more performing applications that can run on small capacity devices, as smartphones or netbooks, this work's results open the possibility to apply such cryptographic methodology to many scenarios in mobile devices use and to continue working in the development of innovative cryptographic methods based on new ideas.

REFERENCES

Ahmad, T., Hu, J., and Han, S. (2009). An efficient mobile voting system security scheme based on elliptic curve cryptography. *Network and System Security, International Conference on*, 0:474–479.

Barker, E., Johnson, D., and Smid, M. (2007). *NIST SP 800-56A - Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*. NIST.

Bodrato, M. (2009). Personal communication.

Celler, F. and Leedham-Green, C. R. (1995). Calculating the order of an invertible matrix. In *In Groups and Computation II*, pages 55–60. American Mathematical Society.

Diffie, W. and Hellman, M. E. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654.

Dignan, L. (2009). Smartphone operating systems: The market share, usage disconnect.

Grillo, A., Lentini, A., Me, G., and Italiano, G. F. (2008a). Transaction oriented text messaging with trusted-sms. In *ACSAC*, pages 485–494. IEEE Computer Society.

Grillo, A., Lentini, A., Me, G., and Rulli, G. (2008b). Trusted sms - a novel framework for non-repudiable sms-based processes. In Azevedo, L. and Londral, A. R., editors, *HEALTHINF (1)*, pages 43–50. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.

NIST (1999). Recommended elliptic curves for federal government use. Technical report, NIST.

Ottaviani, V., Italiano, G. F., Grillo, A., and Lentini, A. (2009). Benchmarking for the qp cryptographic suite. Technical report, University of Rome "Tor Vergata", dept. of Informatics, Systems and Production.

PRNewswire (2007). Rcs announces 2007 January-June trading data for the global cellular phone open market. Note.

Rescorla, E. (1999). Rfc 2631 - Diffie-Hellman key agreement method. Technical report, RTFM Inc.

Research, C. (2000). Standards for efficient cryptography - SEC 1: Elliptic curve cryptography. Technical Report 20, Certicom Corp., secg-talk@lists.certicom.com.

Sakalauskas, E., Tvarijonas, P., and Raulynaitis, A. (2007). Key agreement protocol (kap) using conjugacy and discrete logarithm problems in group representation level. *Informatica*, 18(1):115–124.

Snol, L. (2009a). More smartphones than PCs by 2011. PC Advisor, August 2009.

Snol, L. (2009b). Smartphones racing past boring mobiles. PC Advisor, August 2009.