Pitfalls in public key cryptosystems based on free partially commutative monoids and groups

María Isabel González Vasco[1] and Rainer Steinwandt[2]

[1] Área de Matemática Aplicada, Universidad Rey Juan Carlos
C/ Tulipán s/n. 28933, Móstoles, Madrid, Spain
migonzalez@escet.urjc.es

[2]IAKS, Arbeitsgruppe Systemsicherheit, Prof. Dr. Th. Beth
Fakultät für Informatik, Universität Karlsruhe,
Am Fasanengarten 5, 76 131 Karlsruhe, Germany
steinwan@ira.uka.de

Abstract

At INDOCRYPT 2003 Abisha, Thomas, and Subramanian proposed two public key schemes based on word problems in free partially commutative monoids and groups. We show that both proposals are vulnerable to chosen ciphertext attacks, and thus in the present form must be considered as insecure.

Keywords: word problem, finitely presented group, public key cryptography

1 Introduction

The identification of mathematical problems that can serve as sound foundation for the construction of public key schemes is a rather active area of research. It has turned out to be quite hard to come up with practical and secure proposals that are not variants of proposals based on factoring large integers or computing discrete logarithms in suitable represented finite cyclic groups. One line of research in this context focuses on the use of word problems originating in group or language theory (see [4] for an introduction).

Unfortunately, some proposals in this direction turn out to be susceptible to annoyingly simple attacks that circumvent the underlying (difficult) theoretical problem (cf. [2, 3]). Up until now, it remains an interesting challenge to build practical cryptographic schemes originating in word problems.

At INDOCRYPT 2003 Abisha, Thomas, and Subramanian proposed two public key cryptosystems based on *free partially commutative monoids and groups* [1]. In this contribution we show that in the present form these proposals do not offer acceptable cryptographic security, as they succumb to quite efficient chosen ciphertext attacks. Recall that *chosen ciphertext attacks* are those carried out with "restricted" access to the decryption device: that is, the adversary gains knowledge about the target ciphertext (or the secret key) by selecting different ciphertexts for which he is given the corresponding plaintexts. The strongest type of cryptanalysis consists of *ciphertext only attacks*, where the adversary's only source of information is ciphertext (and the public key). We will see that against one of the examples presented in [1], a very efficient ciphertext only attack can be mounted that enables an attacker to decrypt arbitrary ciphertexts.

2 A proposal based on free partially commutative monoids

In this section we shortly recall the basic set-up of the first public key cryptosystem proposed by Abisha et al. at INDOCRYPT 2003—for further details we refer to the original paper [1]. We denote by Σ some (finite) alphabet, and $\theta \subseteq \Sigma \times \Sigma$ specifies a so-called *concurrency relation*, i. e., $(a, b) \in \theta$ means that each occurrence of ab in a word $u \in \Sigma^*$ can be replaced by ba and vice-versa. If $v \in \Sigma^*$ is derived from a word $u \in \Sigma^*$ by repeatedly applying such replacements, we write $u \equiv_\theta v$. In particular, $\equiv_\theta$ is a congruence relation, and it is pointed out in [1] that the word problem in the *free partially commutative monoid* $\Sigma^*/\equiv_\theta$ can be decided efficiently.

Let Δ denote a finite alphabet whose cardinality is "sufficiently greater than that of Σ" ([1] provides no further details here).

The secret data consists of Σ, θ along with two words $x_0, x_1 \in \Sigma^*$ such that $x_0 \not\equiv_\theta x_1$. Further on, the secret key contains a monoid homomorphism $g : \Delta^* \longrightarrow \Sigma^*$ which obeys the following conditions:

- For $\delta \in \Delta$ we either have $g(\delta) = \lambda$ (the empty word) or $g(\delta) \in \Sigma$.
- At least for one letter $\delta \in \Delta$ we have $g(\delta) \neq \lambda$.

The public data consists of two words $y_0 \in g^{-1}(x_0)$ and $y_1 \in g^{-1}(x_1)$. Further on, a Thue system $T \subseteq \Delta^* \times \Delta^*$ is specified such that for $(u, v) \in T$ we either have $(g(u), g(v)) \in \{(ab, ba), (ba, ab)\}$ with $(a, b) \in \theta$ or we have $g(u) = g(v)$. Thus, repeatedly applying rules in T to y_i yields another element in $g^{-1}(x_i)$ ($i \in \{0, 1\}$).

Here applying a rule $(u, v) \in T$ to a word $w \in \Delta^*$ is to be understood as replacing an occurrence of u in w with v (or an occurrence of v in w with u).

To encrypt a bit $b \in \{0, 1\}$ we start with the corresponding public word y_b and repeatedly apply rewrite rules specified in T (no details of this process are specified in [1]). The resulting word $c \in \Delta^*$ forms the ciphertext.

To decrypt a ciphertext $c \in \Delta^*$ the word $g(c) \in \Sigma^*$ is computed. In case of $g(c) \equiv_\theta x_0$ the plaintext is 0, otherwise it is 1. (Note that according to this specification a ciphertext is never considered as invalid.)

3 Security problems in the proposal based on free partially commutative monoids

In the proposed form, the above scheme does not address several issues that are crucial for the security of a practical scheme. In particular, it is unclear how *exactly* the parameters are to be chosen and how the encryption process is to be performed: E. g., how do we decide which rule is to be applied next, and how many "rounds" of rewriting are to be performed? Moreover, even when deciding equivalence of words in Δ^* with respect to T is hard, there can be annoyingly trivial ways for an attacker to bypass this problem. As a (drastic) demonstration of the relevance of this issue we can use the simple example from [1]:

Example 1 *In the simple example put forward in [1], the public Thue system over the alphabet* $\Delta = \{d_1, \ldots, d_9\}$ *reads*

$$\begin{aligned} T = \{&(d_1d_3, d_3d_1), (d_1d_4, d_4d_1), (d_2d_3, d_3d_2), (d_2d_4, d_4d_2), (d_5d_3d_1, d_3d_5), \\ &(d_5d_4d_2, d_4d_5), (d_3d_3d_1, d_3d_3d_1d_5), (d_6d_7, d_7d_6), (d_6d_8, d_8d_6), \\ &(d_7d_9, d_9d_7), (d_8d_9, d_9d_8)\}, \end{aligned}$$

and the public words used for encrypting 0 *and* 1*, respectively, are*

$$\begin{aligned} y_0 &= d_1d_2d_2d_4d_3d_3d_1d_6d_7d_5d_7d_9d_1d_2d_8d_3d_4d_8d_3d_9 \\ y_1 &= d_1d_2d_2d_4d_6d_3d_4d_6d_7d_5d_1d_5d_8d_4d_3d_8d_9. \end{aligned}$$

This Thue system T is designed to have an undecidable word problem. Nevertheless an attacker can easily decrypt arbitrary plaintexts encrypted under such a public key: All rewrite rules in T leave the number of occurrences of the letter d_9 invariant. Consequently, each encryption of 0 *contains the same number of d_9's* *as y_0 does (namely 2), whereas each encryption of 1 results in a ciphertext with a single d_9. In exactly the same way, unauthorized decryption of a bit can be carried out by counting the number of d_3's, d_4's, d_6's, or d_7's in the ciphertext, as the number of occurrences of these letters in y_0 and y_1 is different and is not altered by the rewriting rules.*

Unfortunately, even when the system parameters are chosen in such a way that ciphertext only attacks can be excluded, the following chosen ciphertext attack can still apply:

1. For each $\delta \in \Delta$ (more precisely, for each letter δ occurring in the public data), the attacker sends the concatenation $y_0\delta$ to the owner of the secret key. If the resulting ciphertext does not decrypt to 0, we know that $g(\delta) \neq \lambda$. On the other hand, if the resulting plaintext is 0, we may assume $g(\delta) = \lambda$ (to increase the plausibility of this assumption, one may send another ciphertext formed by inserting δ at several randomly chosen positions in y_0.)

   Thus, assuming a realistic public key size, we must assume that an attacker can determine the set $\{\delta \in \Delta : g(\delta) \neq \lambda\}$. Let Δ', T', y_0', y_1' denote the values obtained from Δ, T, y_0, y_1 through removal of these "superfluous" letters.

2. Next, we want to check which letters have identical images under g: Let η_0 be the first letter of y_0', and replace some occurrence(s) of η_0 in y_0' by a letter $\xi_0 \in \Delta' \setminus \{\eta_0\}$. If the obtained word no longer decrypts to 0, we know that $g(\eta_0) \neq g(\xi_0)$; on the other hand if the ciphertext obtained by such a replacement still decrypts to 0, it is plausible to assume $g(\eta_0) = g(\xi_0)$. After knowing (or at least having plausible candidates for) the elements $\xi \in \Delta'$ with $g(\xi) = g(\eta_0)$, we can proceed in the same manner with another letter of y_0'. In this way, we learn which letters in y_0' (probably) have identical images under g.

   Next, we apply the same technique to some encryption of y_0' under T' in order to get information about letters not contained in y_0'. Note that—provided the decryption procedure does not detect invalid ciphertexts (which in the original specification from [1] is the case)—we are limited to those rules in T' that can be applied when encrypting y_0: As invalid ciphertexts always decrypt to 1, modifying encryptions of 1 is not that helpful. Nevertheless, we must assume that the described approach allows to reveal much information on "redundant" letters in Δ by means of $\mathrm{O}(\Delta'^2)$ (fake) chosen ciphertexts.

3. After the previous step we can select a subset $\Delta'' \subseteq \Delta'$ which contains exactly one letter of many (possibly all) preimages $g^{-1}(\sigma)$ $(\sigma \in \Sigma)$. Let T'',

y''_0, y''_1 be the variants of T', y'_0, y'_1 obtained by replacing each letter with its representative in Δ''. In order to learn which letters in $g(\Delta'')$ commute, we proceed analogously as in the previous step: By applying rewrite rules in T'' to y''_0, we try for each pair $(\delta, \pi) \in \Delta'' \times \Delta''$ $(\delta \neq \pi)$ to find encryptions of 0 which contain the letter sequence $\delta\pi$ or $\pi\delta$. Then we replace $\delta\pi$ with $\pi\delta$ (resp. $\pi\delta$ with $\delta\pi$), and check whether this "partially commuted" ciphertext still decrypts to 0.

Thus, with $\mathrm{O}(\Delta''^2)$ (fake) ciphertexts we can get a plausible candidate for the set $\{(\delta, \pi) \in \Delta'' \times \Delta'' : g(\delta)g(\pi) = g(\pi)g(\delta)\}$.

After having completed these steps (requiring $\mathrm{O}(\Delta^2)$ chosen ciphertexts), an attacker is in a situation comparable to the legitimate owner of the secret key: Given a ciphertext, letters $\delta \in \Delta$ with $g(\delta) = \lambda$ can be removed, and different representatives of the same $\sigma \in \Sigma$ can be replaced with a unique representative in Δ''. Further on, due to Step 3 above, we know (or at least have a plausible guess for) which pairs $(g(\delta), g(\pi))$ belong to the secret concurrency relation θ, so recognizing ciphertexts c with $g(c) \equiv_\theta g(y_0)$ can be considered as feasible. As our attack above always began with an encryption of 0, it may well happen that sometimes we fail in checking the condition $g(c) \equiv_\theta g(y_1)$—e. g., such a ciphertext c could involve a letter $\delta \in \Delta$ which did not occur in any encryption of 0 that we used for our attack. But this is not really a concern: We have good chances to correctly identify all ciphertexts c with $g(c) \equiv_\theta g(y_0)$ and all ciphertexts not satisfying this condition decrypt to 1 anyway. Thus, in summary an attacker has good chances to successfully decrypt a non-negligible part (possibly all) ciphertexts encrypted under the public key.

4 Security problems in the proposal based on free partially commutative groups

The authors of [1] put forward another public key scheme which is essentially a particular case of the one already discussed, where the free partially commutative monoid is actually a group. Adapting the above attack to this proposal is straightforward, and we omit a detailed description of the scheme. One issue which is different from the above setting, and which simplifies the attack, is the following: The second proposal of Abisha et al. makes use of the word problem in finitely presented groups. A consequence of this is the fact, that for each letter $\delta \in \Delta$ a "formal inverse" $\delta^{-1} (\in \Delta^{-1})$ is available whose image under g is determined by $g(\delta)$ already.

By making use of these formal inverses we can easily form (fake) ciphertexts that help to check for arbitrary $u, v \in (\Delta^{\pm 1})^*$ whether $g(u)$ and $g(v)$ represent equivalent words in the secret finitely presented group. For doing so we start with a word u_0 encrypting 0 and insert uv^{-1} at random positions in u_0. By construction of the scheme, g maps all ciphertexts encrypting 0 to the empty word, and if after insertion of uv^{-1} we still obtain a decryption of 0, it is plausible to assume that $g(uv^{-1})$ maps to λ, too. In other words we may assume that $g(u)$ and $g(v)$ represent equivalent words in the secret group. Similarly as in the first scheme, the secret finitely presented group is determined by commutativity relations, and by making use of the formal inverses as just sketched, one can check comparatively easy which generators of the secret group (probably) commute.

In summary, this second scheme is vulnerable to an attack which is quite similar to the one above (and to the reaction attack described in [3]).

5 Conclusion

The above discussion illustrates that in the present form both public key encryption schemes proposed in [1] do not offer acceptable cryptographic security. Besides leaving open crucial details of the key generation and encryption procedure, a chosen ciphertext attack can enable an attacker to decrypt a non-negligible part (possible all) of the ciphertexts.

References

[1] P.J. Abisha, D.G. Thomas, and K.G. Subramanian. PUBLIC KEY CRYPTOSYSTEMS BASED on FREE PARTIALLY COMMUTATIVE MONOIDS and GROUPS. In *INDOCRYPT 2003*, Lecture Notes in Computer Science. Springer, 2003.

[2] M.I. González Vasco and R. Steinwandt. Clouds over a Public Key Cryptosystem Based on Lyndon Words. *Information Processing Letters*, 80:239–242, 2001.

[3] M.I. González Vasco and R. Steinwandt. A Reaction Attack on a Public Key Cryptosystem Based on the Word Problem. *Applicable Algebra in Engineering, Communication and Computing*, to appear. See also Cryptology ePrint Archive: Report 2002/139.

[4] A. Salomaa. *Public-Key Cryptography*, volume 23 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1990.