

# Completion Attacks and Weak Keys of Oleshchuk's Public Key Cryptosystem

Heiko Stamer

University of Kassel,  
Department of Mathematics and Computer Science,  
Heinrich-Plett-Straße 40, D-34132 Kassel, Germany  
stamer@theory.informatik.uni-kassel.de

**Abstract.** This paper revisits a public key cryptosystem which is based on finite string-rewriting systems. We consider a new approach for cryptanalysis of such proposals—the so-called completion attack. If a particular kind of weak key is generated, then a passive adversary is able to retrieve secret messages with a significant probability. Our idea can be applied to other rewriting based cryptosystems as well. Finally we discuss issues concerning the practical usage and present some experimental results. The described vulnerabilities lead to the conclusion that at least the key generation of Oleshchuk's cryptosystem has to be revised.

**Keywords:** Cryptanalysis, completion attack, string-rewriting systems, Church-Rosser property, Knuth-Bendix completion, weak keys.

## 1 Introduction

The security of almost every public key cryptosystem relies on the intractability of only a few number-theoretic problems, e.g., factoring large integers or computing discrete logarithms in finite groups. Unfortunately, no proof of hardness (from a complexity theoretical point of view) is known for these assumptions. Therefore it sounds reasonable to look for other possible trapdoor functions in different areas of mathematics or computer science. Further there is the hope that such proposals [11, 13, 14, 9] will provide some kind of “provable security”, because their underlying questions (e.g. the word problem for finitely presented groups) are undecidable in general. Beside other difficulties a primary problem in the design of such cryptosystems remains: The gap between the average and the worst case hardness of the instances, and hence the possibility of weak keys.

Vladimir A. Oleshchuk [1] proposed a public key cryptosystem that relies on the undecidability of the word problem in semigroups. A basic ingredient of his approach are string-rewriting systems. Each system is represented by a rule set containing ordered pairs of strings over a finite alphabet. Bidirectional rewriting on a string is performed through replacing (non-deterministically chosen) occurrences of the left-hand side by the right-hand side of a rule or vice versa. This operation induces an equivalence relation and we say that two strings are

congruent, if they can be rewritten to each other in finitely many steps. The uniform word problem is the question of whether two given strings are congruent modulo a given rewriting system. This problem is undecidable in general, i.e. there exists no algorithm which terminates for all instances with the correct answer. However, if the set of rules is finite and the string rewriting system has the Church-Rosser property, then the word problem can be solved in polynomial time. This fact has been used by Oleshchuk [1, 2] to construct a trapdoor function and later the so-called Church-Rosser codes.

We consider a quite straightforward technique to attack such kind of cryptosystems. The so-called completion attack employs the well-known Knuth-Bendix algorithm on an improperly chosen public key and constructs an equivalent string-rewriting system which has the Church-Rosser property. We show that it is possible to recover secret messages with significant probability, if the previous step was successful. Due to the undecidable termination of the completion procedure the (in)security of Oleshchuk's cryptosystem with respect to our attack remains somewhat unsettled.

The paper is organized as follows: The next section provides some notations and preliminaries. Then we briefly repeat the definition of Oleshchuk's public key cryptosystem and present an obvious variant. The fourth section is devoted to ideas for cryptanalysis, and in particular to the completion attack. Finally we consider practical issues and discuss experimental results with the attack.

## 2 Preliminaries

Let  $\Sigma$  be a finite alphabet.  $\Sigma^*$  denotes the set of all strings over this alphabet including the empty word  $\epsilon$ . The concatenation of two strings  $x$  and  $y$  is simply written as  $xy$ . Further,  $|x|$  denotes the *length* of a string  $x$ , where  $|\epsilon| = 0$ ,  $|a| = 1$  for  $a \in \Sigma$ , and  $|xa| = |x| + 1$  for  $x \in \Sigma^*$ ,  $a \in \Sigma$ . If  $A, B \subseteq \Sigma^*$  are languages, then their product is defined to be  $AB = \{xy \mid x \in A, y \in B\}$ .

A *string-rewriting system*  $R$  on  $\Sigma$  is a subset of  $\Sigma^* \times \Sigma^*$ . Each pair  $(\ell, r) \in R$  is called *rewrite rule*. The word  $\ell \in \Sigma^*$  is the left-hand side and the word  $r \in \Sigma^*$  the right-hand side of such a rule. Here we will only be dealing with finite string-rewriting systems, i.e.  $R$  is finite and its size is given by  $\|R\| = \sum_{(\ell, r) \in R} |\ell| + |r|$ . Each string-rewriting system induces a *reduction relation*  $\rightarrow_R^*$  on  $\Sigma^*$ , which is the reflexive transitive closure of the single-step reduction relation  $\rightarrow_R = \{(x\ell y, xry) \mid x, y \in \Sigma^* \text{ and } (\ell, r) \in R\}$ . If there is no  $v \in \Sigma^*$  such that  $u \rightarrow_R v$  holds, then the string  $u$  is called *irreducible modulo*  $R$ . We denote the set of all irreducible words modulo  $R$  by  $\text{IRR}(R)$ . For finite string-rewriting systems  $\text{IRR}(R)$  is regular, i.e. a finite-state acceptor recognizing  $\text{IRR}(R)$  can be effectively constructed from the rules of  $R$ . The reflexive, symmetric, and transitive closure of  $\rightarrow_R$  is the *Thue congruence*  $\leftrightarrow_R^*$ . We define the *congruence class* of a word  $u \in \Sigma^*$  as  $[u]_R = \{v \in \Sigma^* \mid v \leftrightarrow_R^* u\}$ . This notation is expandable to  $A \subseteq \Sigma^*$  by  $[A]_R = \{v \in \Sigma^* \mid \exists u \in A : v \leftrightarrow_R^* u\}$ .

An arbitrary string-rewriting system  $R$  is called

- *noetherian* if there exists no infinite sequence of reductions w.r.t.  $R$ ,
- *confluent* if, for all  $u, v, w \in \Sigma^*$ ,  $u \rightarrow_R^* v$  and  $u \rightarrow_R^* w$  imply that  $v$  and  $w$  have a common descendant w.r.t.  $R$  (i.e.  $\exists z \in \Sigma^* : v \rightarrow_R^* z$  and  $w \rightarrow_R^* z$ ),
- *convergent* if  $R$  is noetherian and confluent,
- *length-reducing* if  $|\ell| > |r|$  holds for each rewrite rule  $(\ell, r) \in R$ .

A string-rewriting system  $R$  is *Church-Rosser* (i.e.  $R$  has the Church-Rosser property) if, for all  $x, y \in \Sigma^*$  with  $x \leftrightarrow_R^* y$ , there exists a word  $z \in \Sigma^*$  such that  $x \rightarrow_R^* z$  and  $y \rightarrow_R^* z$ . Hence,  $R$  is Church-Rosser if and only if  $R$  is confluent [3]. For finite length-reducing systems this property is decidable in polynomial time [7]. If a finite length-reducing system  $R$  is Church-Rosser, then each congruence class has a unique irreducible element (modulo  $R$ ) called *normal form*. Thus the corresponding word problem

**Instance:** Two arbitrary strings  $x, y \in \Sigma^*$ .

**Question:** Are  $x$  and  $y$  congruent modulo  $R$ , i.e. does  $x \leftrightarrow_R^* y$  holds?

can be solved in linear time by comparing the normal forms [4].

Let  $R_1$  and  $R_2$  be string-rewriting systems on  $\Sigma$ .  $R_1$  *refines*  $R_2$ , if for all  $u, v \in \Sigma^*$  the congruence  $u \leftrightarrow_{R_1}^* v$  implies  $u \leftrightarrow_{R_2}^* v$ . If  $R_1$  refines  $R_2$  and  $R_2$  refines  $R_1$ , then they generate the same Thue congruence and are called *equivalent*. It is straightforward that  $R_1$  refines  $R_2$ , if and only if the congruence  $\ell \leftrightarrow_{R_2}^* r$  holds for each rewrite rule  $(\ell, r) \in R_1$ .

Let  $>$  be a strict partial ordering on  $\Sigma^*$ . This ordering is called *admissible* if  $u > v$  implies that  $xvy > xvy$  holds for all  $x, y \in \Sigma^*$ , and it is called *well-founded* if there is no infinite strictly descending sequence  $u_1 > \dots > u_i > u_{i+1} > \dots$ . A string-rewriting system  $R$  on  $\Sigma$  is *compatible* with a given ordering  $>$ , if  $\ell > r$  holds for each rewrite rule  $(\ell, r) \in R$ .

A nonempty set  $C \subseteq \Sigma^*$  is called a *code*, if for all words  $u_{i_1}, \dots, u_{i_n} \in C$ ,  $u_{j_1}, \dots, u_{j_m} \in C$ , the equality of  $u_{i_1} \dots u_{i_n} = u_{j_1} \dots u_{j_m}$  implies  $u_{i_1} = u_{j_1}$ . By induction we deduce  $n = m$  and  $u_{i_k} = u_{j_k}$  for all  $1 \leq k \leq n$ . If  $C$  is a code then any word from  $C^*$  has a unique factorization over  $C$ .

### 3 Oleshchuk's Public Key Cryptosystem

We briefly repeat the original definition of Oleshchuk's public key cryptosystem [1]. Afterwards a subsection appends a necessary requirement for unique decryption, which was established later in [2]. Finally we slightly vary the discussed cryptosystem by introducing an additional secret morphism.

Let  $\Sigma$  be the plaintext alphabet of possible messages  $\mathcal{M} = \{w \mid w \in \Sigma^*\}$ . Without loss of generality, we consider only the binary case  $\Sigma = \{x_0, x_1\}$ . The ciphertext alphabet  $\Delta$  should be larger than  $\Sigma$ , i.e.  $|\Delta| > |\Sigma|$ .

*Key Generation.* Let  $T$  be a finite and length-reducing Church-Rosser string-rewriting system on  $\Delta$ . We choose  $u_1, u_2, \dots, u_t \in \text{IRR}(T)$  such that, for all  $i, j = 1, \dots, t$ , the word  $u_i u_j$  is irreducible modulo  $T$  and the set  $\{u_1, u_2, \dots, u_t\}$  is a code. Now let  $R_0, R_1 \subset \{u_1, u_2, \dots, u_t\}$  be two nonempty disjoint sets,

i.e.  $R_0 \cap R_1 = \emptyset$ . Further let  $L_0 \subseteq [R_0]_T$  and  $L_1 \subseteq [R_1]_T$  be two nonempty regular languages. Note that by construction  $L_0 \cap L_1 = \emptyset$  because  $T$  is confluent and  $R_0, R_1$  are disjoint. The next step of key generation picks a finite string-rewriting system  $S$  on  $\Delta$  such that  $\ell \leftrightarrow_T^+ r$ , for all rewrite rules  $(\ell, r) \in S$ , i.e.  $S$  refines  $T$ . This property can be tested easily because  $T$  is Church-Rosser and thus the corresponding word problem is decidable in linear time [4]. However, it is not clear how to construct  $S$  efficiently without leaking information about  $T$ .

The finite string-rewriting system  $S$  and the languages  $L_0, L_1$  form the public key  $K_{\text{pub}}$ . The finite Church-Rosser system  $T$  and the sets  $R_0, R_1$  should be kept secret, because they represent the private key  $K_{\text{sec}}$ .

*Encryption.* The encryption of the letter  $x_i$  is a random word  $y \in [L_i]_S$ . Therefore the non-deterministic function  $\text{Encrypt} : \mathcal{M} \rightarrow \mathcal{C}$  maps a possible message  $m = x_{i_1}x_{i_2} \cdots x_{i_n}$ , where  $x_{i_k} \in \Sigma$ ,  $k = 1, \dots, n$ , to a random ciphertext  $c \in [L_{i_1}L_{i_2} \cdots L_{i_n}]_S$ . In practice one will do the following two steps:

1. Encode the plaintext  $m = x_{i_1}x_{i_2} \cdots x_{i_n}$  into  $\hat{m} = \hat{x}_{i_1}\hat{x}_{i_2} \cdots \hat{x}_{i_n}$ , where each string  $\hat{x}_{i_k}$  is randomly chosen from the corresponding language  $L_{i_k}$ .
2. Rewrite the  $\hat{m}$  randomly and uniformly according to the rules of  $S$ .

*Decryption.* For the decryption of a secret message  $c \in \mathcal{C}$  one has to find a word  $\hat{m} \in (L_0 \cup L_1)^*$  such that  $c \leftrightarrow_S^* \hat{m}$  holds. In general the finite string-rewriting system  $S$  may have an undecidable word problem [4] and even decidability does not guarantee that the problem is computationally feasible [6].

With the secret key  $K_{\text{sec}} = (T, R_0, R_1)$  decryption becomes easy, because  $T$  has the Church-Rosser property and thus there exists a uniquely defined word  $\tilde{m} \in \text{IRR}(T)$  such that  $c \rightarrow_T^* \tilde{m}$ . This normal form can be found in linear time [4] and its factorization  $\tilde{m} = u_{i_1}u_{i_2} \cdots u_{i_n}$  (where  $u_{i_k} \in R_{i_k}$ ) obviously reveals the plaintext  $m = x_{i_1}x_{i_2} \cdots x_{i_n}$  of the encrypted message.

### 3.1 Necessary Requirement for Unique Decryption

It was observed [2] that the condition  $u_i u_j \in \text{IRR}(T)$  for all  $i, j = 1, \dots, t$  is not sufficient for a unique decoding. In fact one has to ensure (during the process of key generation) that  $(R_0 \cup R_1)^* \subseteq \text{IRR}(T)$  holds. However, this generalization can be effectively tested, since both sides are regular languages.

Let  $\psi_T = \max\{|\ell_1|, \dots, |\ell_{|T|}|\}$  for all rewrite rules  $(\ell_i, r_i) \in T$  and let  $\psi_R = \max\{|u_1|, \dots, |u_t|\}$  for all words  $u_i \in (R_0 \cup R_1)$ . Now it is sufficient to check whether the inclusion  $(R_0 \cup R_1)^{\leq 2 \max\{\psi_T, \psi_R\}} \subseteq \text{IRR}(T)$  holds. For example, this can be performed by generating all concatenations of length less than or equal to  $2 \max\{\psi_T, \psi_R\}$  and verifying the irreducibility for each of them.

### 3.2 Morphism Strengthened Variant

The following idea is based on a well-known technique [12, 9, 19] to hide the structure of the rewriting steps by an additional morphism. Let  $\Gamma$  be a new ciphertext alphabet of much greater cardinality than  $\Delta$  and let  $g : \Gamma^* \rightarrow \Delta^*$  be a monoid homomorphism which meets the following conditions:

1. For each letter  $\gamma \in \Gamma$  we either have  $g(\gamma) = \epsilon$  or  $g(\gamma) \in \Delta$ . In the first case we will call  $\gamma$  a *dummy symbol*.
2. At least for each letter  $\delta \in \Delta$  there exists a  $\gamma \in \Gamma$  such that  $g(\gamma) = \delta$ .

In the strengthened variant of Oleshchuk's cryptosystem such a morphism  $g$  is part of the secret key, i.e.  $K_{\text{sec}} = (T, R_0, R_1, g)$ . The modified public key  $K_{\text{pub}} = (S, L_0, L_1)$  is obtained from the original  $(S, L_0, L_1)$  as follows:

$$S = \{(g^{-1}(\ell), g^{-1}(r)) \mid (\ell, r) \in S\} \cup D$$

The finite set  $D$  contains *dummy rules*  $(\ell, r)$  whose words  $\ell, r \in \Gamma^*$  satisfy  $g(\ell) = g(r) = \epsilon$ . Obviously, the refinement property of  $S$  remains valid under the images of  $g$ , i.e. the congruence  $g(\ell) \leftrightarrow_T^* g(r)$  holds for all  $(\ell, r) \in S$ .

The generation of  $L_0$  respectively  $L_1$  depends on their representation: If  $L_i$  is a finite set one can easily compute  $L_i = \{g^{-1}(w) \mid w \in L_i\}$ . Otherwise, if they are represented by a grammar  $G_{L_i}$  one might apply  $g^{-1}$  to each terminal symbol  $\delta \in \Delta$  in all derivation rules of  $G_{L_i}$ .

Finally the decryption has to be changed slightly: In a new initial step the legal recipient of an encrypted message  $c \in \mathcal{C}$  can apply the morphism  $g$  in linear time. Afterwards he proceed as usual, i.e. reducing the result  $g(c)$  modulo  $T$ .

## 4 Cryptanalysis

Like similar cryptosystems [15, 16] the discussed approach is vulnerable to particular cryptanalytic attacks, if weak keys are chosen during the key generation. Specifically, here we are concerned about "bad string-rewriting systems".

### 4.1 Completion Attack on $S$

Oleshchuk [1] already noticed that it is not necessary to find the exact secret system  $T$  generated by the legal key owner. Any Church-Rosser system  $T'$  where all of the conditions

1.  $S$  refines  $T'$
2.  $[L_0]_{T'} \cap [L_1]_{T'} = \emptyset$
3.  $([L_0]_{T'} \cup [L_1]_{T'}) \cap \text{IRR}(T')$  is a code
4.  $([L_0]_{T'} \cup [L_1]_{T'})^* \subseteq \text{IRR}(T')$  (reformulated according to [2])

apply, can be used to decrypt messages. In general, there is no algorithm to decide whether a finite string-rewriting system is equivalent to any finite Church-Rosser system [5]. But a cryptanalyst can try techniques known as *completion procedures* to construct such a convergent system  $T'$  from  $S$ .

The Knuth-Bendix completion [17] takes as input a finite string-rewriting system  $S$  on  $\Delta$  and an admissible well-founded strict partial ordering  $>$  on  $\Delta^*$ . Based on this ordering the system  $S$  is turned into an equivalent system  $T'$  that is compatible with  $>$  by orienting each rule with respect to this ordering. Thus  $T'$  will be noetherian. Then the critical pairs of  $T'$  are computed, and for

each critical pair that does not resolve a new rule is introduced. Unfortunately, each new rule can lead to new unresolvable critical pairs, and hence, this process may not terminate. Moreover, the termination highly depends on the used ordering  $>$ . A detailed description of the Knuth-Bendix completion is omitted here. Interested readers are referred to the existing literature [10, 17, 3].

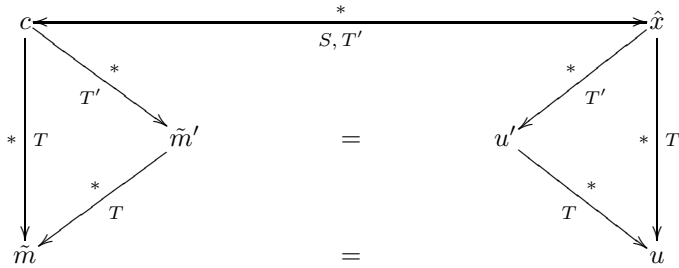
Let  $(S, L_0, L_1)$  be a *KBC-weak* public key, i.e. the Knuth-Bendix completion procedure terminates and outputs a length-reducing system  $T'$  which is equivalent to  $S$ . Thus a cryptanalyst can reduce an observed ciphertext  $c \in \mathcal{C}$  to a normal form  $\tilde{m}' \in \text{IRR}(T')$  in linear time [4], i.e.  $c \rightarrow_{T'}^* \tilde{m}'$ . Further let us assume that the regular languages  $L_0$  and  $L_1$  are finite. By reducing all words of the  $L_i$ 's to their normal forms modulo  $T'$  the adversary will get the finite languages  $R'_i = \{u' \in \text{IRR}(T') \mid \exists \hat{x} \in L_i : \hat{x} \rightarrow_{T'}^* u'\}$  efficiently. Moreover, these sets are of the same size as the  $L_i$ 's.

First, consider the restricted case of one-bit messages ( $n = 1$ ): We can reduce the hard problem of deciding whether  $c \in [L_0]_S$  or  $c \in [L_1]_S$  to a much easier question modulo  $T'$ .

**Lemma 1 (One-bit Messages).**  $c \in [L_i]_S$  if and only if  $\tilde{m}' \in R'_i$ , for  $i = 0, 1$ .

*Proof.* We prove both cases  $i = 0$  and  $i = 1$  in common:

$\Rightarrow$  For  $c \in [L_i]_S$  there exists a word  $\hat{x} \in L_i$  such that  $c \leftrightarrow_{T'}^* \hat{x} \leftrightarrow_{T'}^* \tilde{m}'$  since  $S$  and  $T'$  are equivalent. Further, by construction each  $\hat{x}$  has an irreducible word  $u' \in R'_i$  modulo  $T'$  where  $\hat{x} \rightarrow_{T'}^* u'$ . Finally  $u' = \tilde{m}'$ , because  $T'$  is confluent and consequently the normal forms are unique.



$\Leftarrow$  The other direction follows by similar arguments. □

Now we turn to longer messages, i.e.  $n > 1$ . Here appears the problem that not necessarily  $(R'_0 \cup R'_1)^* \subseteq \text{IRR}(T')$  and thus the decoding may be ambiguous.

Let  $\bar{L}'_0 \subseteq L_0$  respectively  $\bar{L}'_1 \subseteq L_1$  be the finite set of all strings  $\hat{x}_{i_j} \in L_{i_j}$  used during the encryption of a fixed ciphertext  $c$ , i.e.  $c \in [\bar{L}'_{i_1} \bar{L}'_{i_2} \dots \bar{L}'_{i_n}]_S$ . Further denote by  $\bar{R}'_i = \{u' \in \text{IRR}(T') \mid \exists \hat{x} \in \bar{L}'_i : \hat{x} \rightarrow_{T'}^* u'\}$  the corresponding normal forms modulo  $T'$ . Under three additional conditions, namely that  $T'$  is code preserving w.r.t.  $\bar{L}'_i$ , we can generalize the previous Lemma 1.

**Lemma 2 (Arbitrary Messages).** If  $[\bar{L}'_0]_{T'} \cap [\bar{L}'_1]_{T'} = \emptyset$ ,  $([\bar{L}'_0]_{T'} \cup [\bar{L}'_1]_{T'})$  is a code, and  $([\bar{L}'_0]_{T'} \cup [\bar{L}'_1]_{T'})^* \subseteq \text{IRR}(T')$ , then

1.  $(\bar{R}'_0 \cup \bar{R}'_1)^* \subseteq \text{IRR}(T')$  and  $(\bar{R}'_0 \cup \bar{R}'_1)$  is a code, and
2.  $c \in [\bar{L}'_{i_1} \bar{L}'_{i_2} \cdots \bar{L}'_{i_n}]_S$  if and only if  $\tilde{m}' \in \bar{R}'_{i_1} \bar{R}'_{i_2} \cdots \bar{R}'_{i_n}$ . (for all  $n \in \mathbb{N}$ )

*Proof.* The first claim is obvious, because  $\bar{R}'_0 \subseteq [\bar{L}'_0]_{T'}$  and  $\bar{R}'_1 \subseteq [\bar{L}'_1]_{T'}$ . The remaining part follows by an inductive application of Lemma 1 using the fact, that  $(\bar{R}'_0 \cup \bar{R}'_1)$  is a code and that all possible concatenations of the included words are irreducible modulo  $T'$ .  $\square$

**Proposition 1.** *The congruence classes  $[L_0]_{T'}$  and  $[L_1]_{T'}$  are disjoint.*

*Proof.* Assume the contrary.  $T'$  and  $S$  are equivalent, i.e.  $\leftrightarrow_S^* = \leftrightarrow_{T'}^*$ , and thus for each  $w \in ([L_0]_{T'} \cap [L_1]_{T'})$  the congruence  $\hat{w}_0 \leftrightarrow_S^* w \leftrightarrow_S^* \hat{w}_1$  holds for at least two words  $\hat{w}_0 \in L_0, \hat{w}_1 \in L_1$ . However, since  $S$  refines  $T$  and the string-rewriting system  $T$  is Church-Rosser there exists a unique normal form  $u \in \text{IRR}(T)$  such that  $\hat{w}_0 \rightarrow_T^* u$  and  $\hat{w}_1 \rightarrow_T^* u$ . Hence it either violates the condition  $L_0 \cap L_1 = \emptyset$  or contradicts the unique decryption of messages.  $\square$

**Theorem 1.** *If the public key is KBC-weak and if  $L_0, L_1$  are finite sets, then a passive adversary can retrieve the plaintext of an encrypted message with significant probability in linear time.*

*Proof.* The adversary applies Lemma 2. In the case of a KBC-weak key (completion procedure terminates) the first condition is always fulfilled, because  $[L_0]_{T'} \supseteq [\bar{L}'_0]_{T'}$  and  $[L_1]_{T'} \supseteq [\bar{L}'_1]_{T'}$  are already disjoint (see Proposition 1). Moreover, the remaining conditions are often satisfied for short messages or sparse sets  $\bar{L}'_0, \bar{L}'_1$  (including the trivial case of one-bit messages). As  $(\bar{R}'_0 \cup \bar{R}'_1)$  is a code the adversary can easily determine the corresponding factorization of  $\tilde{m}'$  and hence the plaintext. Further, if one of the necessary conditions does not hold, a passive adversary is probably still able to retrieve partial information about the corresponding plaintext  $m$ . Note that for a successful attack it may be sufficient to distinguish some subwords of  $\tilde{m}'$  between  $R'_0$  and  $R'_1$ .  $\square$

*Example 1 (Knuth-Bendix Completion Attack).*

$$T = \{(cb, c), (aa, a), (ab, a)\}, S = \{(ab, aab), (cba, ca), (baa, ba)\}$$

$$R_0 = \{cacac\}, R_1 = \{aca\}, L_0 = \{caacbab\}, L_1 = \{abacbaab\}$$

On input of  $S$  and  $>_{\text{lex}}$  (length-lexicographical ordering) the Knuth-Bendix completion terminates with the length-reducing Church-Rosser system

$$T' = \{(aab, ab), (cba, ca), (baa, ba), (caa, ca)\}.$$

By reducing  $L_0, L_1$  modulo  $T'$  we get  $R'_0 = \{cacabc\}$  and  $R'_1 = \{abacab\}$ .

$$\begin{array}{l} caacbab \xrightarrow{(4)}_{T'} cacbab \xrightarrow{(2)}_{T'} cacabc \\ abacbaab \xrightarrow{(2)}_{T'} abacaab \xrightarrow{(4)}_{T'} abacab \end{array}$$

Now suppose that the cryptanalyst observes the ciphertext  $c = cbaacabc$ , which can be reduced in two steps to  $\tilde{m}' = cacabc \in [R'_0]_{T'}$ .

$$c = cbaacabc \xrightarrow{(2)}_{T'} caacabc \xrightarrow{(4)}_{T'} cacabc = \tilde{m}' \in [R'_0]_{T'}$$

Hence the corresponding plaintext is the single letter  $x_0$ .

As consequence we have to avoid the KBC-weakness during the key generation. Unfortunately, such a property is undecidable [5] in general. Hence the security of Oleshchuk's cryptosystem with respect to completion attacks remains undecidable. Furthermore, even the morphism strengthened variant does not really protect against this kind of attack, because it leaves the distinguish ability between  $R'_0$  and  $R'_1$  unchanged. In such a case, however, the effort of the adversary increases since he has to deal with the included dummy rules.

The straightforward idea to prevent our attack might be the usage of infinite regular languages  $L_i$ , e.g., represented by grammars in the public key. Nonetheless, the  $\bar{L}'_i$ 's are still finite, because the sender can only apply finitely many rewrite steps during the encryption. But the point is that hopefully these sets are large enough such that the computation of the distinguishing normal forms  $\bar{R}'_i$  is infeasible for a bounded adversary.

We stress that completion attacks are applicable to other rewriting based schemes, e.g., the public key cryptosystem of Samuel et al. [19]. In their proposal (Church-Rosser) tree replacement systems [20, 21] are used for building the trapdoor. Thus it seems to be very likely to mount successful *tree completion procedures* and retrieve parts of secret messages, if KBC-weak keys are chosen as well.

#### 4.2 Guessing $T$ by Pre- or Suffix Properties of $S$

Other properties of weak keys are exploitable: A pitfall stems from the fact that  $S$  refines  $T$ . Of course, if  $S = \{(xly, xry) \mid (\ell, r) \in T, x, y \in \Delta^*\}$  the congruence  $xly \leftrightarrow_T^* xry$  holds for each rewrite rule in  $S$ .

A cryptanalyst can guess the secret Church-Rosser system  $T$ , if  $S$  was simply chosen like above, i.e. for some  $(\ell, r) \in T$  with  $|\ell| > |r|$  there exists a prefix  $z \in \Delta^*$  such that  $(z\ell, zr) \in S$  or  $(zr, z\ell) \in S$ .

#### 4.3 Further Ciphertext-Only Attacks

If the string-rewriting system  $S$  and the sets  $L_0, L_1$  are not carefully generated, then (analogously to [16]) additional information about the corresponding plaintext  $m \in \mathcal{M}$  is leaked by a given ciphertext  $c \in \mathcal{C}$ .

For example, assume that a letter of the ciphertext alphabet  $\Delta$  appears only in words either from  $L_0$  or  $L_1$ . Assume further that this relation is preserved by the rules of  $S$ . Now the adversary counts the occurrences of such a letter in  $c$  and hence obtains information about the number and positions of the  $x_0$ 's resp.  $x_1$ 's in the plaintext  $m$ . One can generalize this kind of attack to other "measures", e.g., if  $S$  preserves some unique subword or a characteristic length. Again, the point is that a cryptanalyst only has to distinguish between  $L_0$  and  $L_1$ .

## 5 Practical Issues

This section sketches some questions that arose during our implementation of Oleshchuk's cryptosystem. The programming was done as *proof of concept* in approximately 1 600 lines of C++ code. Thus features and documentation are very limited: The program constructs a random key pair  $(K_{\text{pub}} = (S, L_0, L_1), K_{\text{sec}} = (T, R_0, R_1))$  and performs some simple encryption/decryption operations on one-bit and larger messages. Finally, if possible, the completion attack (see Section 4.1) is mounted. We provide the source code [23] under the *GNU General Public License*, if the reader would like to verify the results of the next section or if he want to use parts of our work in further cryptanalysis.

*Choosing "Cryptographically Good" Parameter Settings.* This seems to be a serious question since many possible parameters may have influence on the security of the entire cryptosystem, e.g., the size of the ciphertext alphabet  $\Delta$ , the sizes of the string-rewriting systems  $T$  and  $S$ , the sizes of the sets  $R_i$ , and, if finite sets  $L_i$  are used, the number of applied rules during their generation. In the original paper there were only few hints how to choose these parameters appropriately.

Concerning the ciphertext alphabet  $\Delta$  we can say that in the unary case the word problem becomes decidable for finite string-rewriting systems [3]. On the other hand, if we consider a bounded number of rewrite rules over an arbitrary finite alphabet, then there exists a string-rewriting system with only three rules which has an undecidable word problem [18]. It is a well-known open question [22] of whether or not this problem becomes decidable if we consider only one-rule rewriting systems. Hence  $S$  should have at least three rules.

*Generating a String-rewriting System  $S$  that Refines  $T$ .* Up to now we don't have any other method than to randomly guess  $S$  and check whether  $\ell \leftrightarrow_T^+ r$  holds for all rewrite rules  $(\ell, r) \in S$ . Each obvious strategy to perform this in a more efficient way will probably introduce new vulnerabilities, e.g., by the inherent structure of the derived public key space. Beside the other concerns, this issue is the most crucial problem since it makes the generation algorithm really impractical for reasonable key sizes.

*Encrypting Messages.* Here we have to ensure that a cryptanalyst cannot handle the word problem by a brute-force search in the Thue congruence. Therefore the number of nodes in the derivation tree of  $c \leftrightarrow_S^* \hat{m}$  should grow exponentially in the number of performed rewrite steps during the encryption.

*Ciphertext Blow-up.* From a practical point of view the enormous ciphertext blow-up of the encryption function is undesirable. Of course, we can try to balance the application of length-increasing and length-reducing rewrite rules, but such a strategy could be another starting point for successful attacks.

## 6 Experimental Results

Now we want to estimate how likely the proposed completion attack works in practice. We stress again that in general it is undecidable whether a public key is

**Table 1.** Experimental results with the completion attack (three independent runs)

	$ S  = 3$	$ S  = 4$	$ S  = 5$
$ R_0 \cup R_1  = 5$	$23 : 12, 24 : 20, 23 : 14$ 23.3% vs. 15.3%	$13 : 4, 8 : 7, 15 : 12$ 12% vs. 7.6%	$5 : 3, 4 : 1, 4 : 3$ 4.3% vs. 2.3%
$ R_0 \cup R_1  = 10$	$17 : 13, 19 : 10, 23 : 11$ 19.6% vs. 11.3%	$13 : 8, 9 : 5, 11 : 2$ 11% vs. 5%	$4 : 2, 5 : 1, 3 : 2$ 4% vs. 1.6%
$ R_0 \cup R_1  = 25$	$22 : 10, 23 : 13, 26 : 10$ 23.6% vs. 11%	$12 : 4, 15 : 8, 9 : 6$ 12% vs. 6%	$3 : 1, 4 : 1, 5 : 2$ 4% vs. 1.3%

KBC-weak. Thus our experimental results can only give an very rough approximation. Further there are practical limitations, in particular the generation of  $S$ , while performing such an analysis. Hence we restrict our investigation to “small keys” which may have less cryptographic relevance. Table 1 shows the number of successful Knuth-Bendix completions on  $S$  (using the length-lexicographical ordering) versus the number of successful completion attacks (Theorem 1) itself. The attack was performed on a fixed 112-bit message and we have count only a proper decryption as successful. The completion procedure was aborted either after three iterations or if more than 250 critical pairs occurred. For each single run our program [23] has generated (randomly and uniformly) one hundred instances of Oleshchuk’s cryptosystem with  $|\Delta| = 3$ ,  $|T| = 3$ , and  $6 \leq ||T|| \leq 12$ . These parameters have been chosen to achieve a reasonable time-frame for the experiment. The above results show that our attack works well in the small parameter scenario, if the Knuth-Bendix completion itself is successful.

## 7 Conclusion

We have shown that the discussed cryptosystem is principally vulnerable to the presented completion attack. Even the strengthened variant does not hide the distinguish ability of the encoding languages properly. Thus, at least in our opinion, it does not offer an acceptable level of cryptographic security. On the other hand, the KBC-weakness of public keys is undecidable in general. Hence it is very hard to estimate how likely our attack works in practice. We leave it as an open question whether Oleshchuk’s cryptosystem can be repaired to withstand the proposed attacks. However, the practical issues have shown clearly that all further effort can be only of theoretical interest.

*Acknowledgment.* The author thanks Friedrich Otto, Tomasz Jurdziński, and Andreas Conz for the interesting discussions regarding this work. Additionally, he thanks the anonymous referees for their helpful comments.

## References

1. Vladimir A. Oleshchuk. On Public-Key Cryptosystem Based on Church-Rosser String-Rewriting Systems. Proceedings of COCOON'95, Lecture Notes in Computer Science 959, pp. 264–269, 1995.
2. Vladimir A. Oleshchuk. Church-Rosser Codes. Proceedings of 5th IMA Conference, Lecture Notes in Computer Science 1025, pp. 199–204, 1996.
3. Ronald V. Book and Friedrich Otto. String-Rewriting Systems. Texts and Monographs in Computer Science, Springer, New-York, 1993.
4. Ronald V. Book. Confluent and other types of Thue systems. *Journal of the ACM* 29:171–183, 1982.
5. Colm O'Dunlaing. Undecidable questions related to Church-Rosser Thue systems. *Theoretical Computer Science* 23:339–345, 1983.
6. Günther Bauer and Friedrich Otto. Finite Complete Rewriting Systems and the Complexity of the Word Problem. *Acta Informatica* 21:521–540, 1984.
7. Deepak Kapur, Mukkai S. Krishnamoorthy, Robert McNaughton, and Paliath Narendran. An  $O(|T|^3)$  algorithm for testing the Church-Rosser property of Thue systems. *Theoretical Computer Science* 35:109–114, 1985.
8. Robert McNaughton, Paliath Narendran, and Friedrich Otto. Church-Rosser Thue systems and formal languages. *Journal of the ACM* 35:324–344, 1988.
9. Valtteri Niemi. Cryptology: Language-Theoretic Aspects. In G. Rozenberg and A. Salomaa (Eds.): Handbook of Formal Languages, Springer, Berlin, 1997.
10. Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite Systems. In J. van Leeuwen (Ed.): Handbook of Theoretical Computer Science, Volume B, Elsevier Science Publishers, Amsterdam, 1990.
11. Neal R. Wagner and Marianne R. Magyarik. A Public-Key Cryptosystem Based on the Word Problem. In *Advances in Cryptology: Proceedings of CRYPTO'84*, Lecture Notes in Computer Science 196, pp. 19–36, 1985.
12. Arto Salomaa. On a public-key cryptosystem based on language theory. *Computers and Security* 7:83–87, 1988.
13. Rani Siromoney and Lisa Mathew. A Public Key Cryptosystem Based on Lyndon Words. *Information Processing Letters* 35:33–36, 1990.
14. Akihiro Yamamura. Public-Key Cryptosystems Using the Modular Group. 1st International Workshop on Practice and Theory in Public Key Cryptography (PKC'98), Lecture Notes in Computer Science 1431, pp. 203–216, 1998.
15. Maria I.G. Vasco and Rainer Steinwandt. Pitfalls in public key cryptosystems based on free partially commutative monoids and groups. Cryptology ePrint Archive: Report 2004/012, 2004.
16. David P. Garcia and Maria G. Vasco. Attacking a Public Key Cryptosystem Based on Tree Replacement. Cryptology ePrint Archive: Report 2004/098, 2004.
17. Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In J. Leech (Ed.): Computational Problems in Abstract Algebra, pp. 263–297, Pergamon Press, New-York, 1970.
18. Yuri Matiyasevich and Geraud Sénizergues. Decision problems for semi-Thue systems with a few rules. Proceedings of the 11th IEEE Symposium on Logic in Computer Science, IEEE Computer Society Press, pp. 523–531, 1996.
19. S.C. Samuel, D.G. Thomas, P.J. Abisha, K.G. Subramanian. Tree Replacement and Public Key Cryptosystem. In A. Menezes, P. Sarkar (Eds.): INDOCRYPT 2002, Lecture Notes in Computer Science 2551, pp. 71–78, 2002.

20. Barry K. Rosen. Tree-Manipulating Systems and Church-Rosser Theorems. *Journal of the ACM* 20:160–187, 1973.
21. Jean H. Gallier and Ronald V. Book. Reductions in Tree Replacement Systems. *Theoretical Computer Science* 37:123–150, 1985.
22. Nachum Dershowitz and Ralf Treinen. The RTA list of open problems.  
<http://www.lsv.ens-cachan.fr/rtaloop/>
23. Heiko Stamer. Implementation of Oleshchuk's Public Key Cryptosystem.  
<http://www.theory.informatik.uni-kassel.de/~stamer/0lkPK2.tar.gz>