PUBLIC KEY ENCRYPTION CAN BE SECURE AGAINST ENCRYPTION EMULATION ATTACKS BY COMPUTATIONALLY UNBOUNDED ADVERSARY

DENIS OSIN AND VLADIMIR SHPILRAIN

Abstract. The main purpose of this paper is to show why, contrary to a prevalent opinion, public key encryption can be secure against "encryption emulation" attacks by computationally unbounded adversary, with one reservation: a legitimate party decrypts correctly with probability that can be made arbitrarily close to 1, but not equal to 1.

1. Introduction

This paper was prompted, in part, by several discussions, mostly informal, related to an earlier paper [5], where, for the first time, a cryptographic protocol based on non-recursiveness of a decision problem rather than on computational hardness of a search problem was suggested. (The protocol of Magyarik and Wagner [2] is *not* based on non-recursiveness of a decision problem, contrary to what the title of their paper may suggest; this was recently pointed out in [1].) Most readers of [5] were perplexed already by the abstract, where it was claimed that "decision problems may be useful when one addresses the ultimate challenge of public key cryptography: to design a cryptosystem that would be secure against "brute force" attacks by computationally unbounded adversary". The following statement made by one of the reviewers is typical:

> . . . key generations can be performed over and over again, each time with fresh randomness, until the public key to be attacked is obtained – this will happen eventually with overwhelming probability! Already the correctness (no matter if perfect or only with overwhelming probability) of the scheme guarantees now that the corresponding secret key (as obtained by the adversary performing key generation) allows to decrypt illegitimately.

This statement actually consists of two separate claims, in two separate sentences. We note that the first one is correct, whereas the second one is not. The crucial point is that it *does* matter whether the correctness of the scheme is perfect or only with overwhelming probability. If the correctness was perfect, then the quoted claim would be valid indeed. However, if there is a gap, no matter how small (it can be easily made on the order of 10^{-200}, see [5]), between 1 and the probability of correct decryption by a legitimate party, then this gap can be very substantially "amplified" for the adversary, thus making the probability of correct illegitimate decryption anything but overwhelming.

We explain this phenomenon in our Section 2, trying to keep the group-theoretic background to a minimum. One of the problems with the paper [5] is that it is somewhat too heavy on combinatorial group theory, at least for a non-expert. Most of that group theory is needed to separate the receiver (Alice) and the adversary (Eve) in power. This is, indeed, a very non-trivial problem that opens several interesting research avenues. It appears, however, that the main point of perplexity for most

Research of the first author was partially supported by the NSF grant DMS-0605093. Research of the second author was partially supported by the NSF grant DMS-0405105.

cryptographers was the fact that encryption by the *sender* (Bob) can be secure against "encryption emulation" attacks by a computationally unbounded adversary. To explain how and why this is possible, we do not really need to introduce any serious group theory.

We also note, in passing, that the problem of security of the sender's encryption algorithms is of independent interest. Of course, in applications to, say, Internet shopping or banking, both the sender's and the receiver's algorithms are assumed to be known to the adversary, and the receiver's decryption algorithms (or algorithms for obtaining public keys) are usually more vulnerable to attacks. However, in some other applications, say, to electronic signatures (not to mention non-commercial, e.g. military applications), decryption algorithms need not be public, whereas encryption algorithms always are. It is therefore important to have a consensus in the cryptographic community on the security claim in the abstract of the present paper.

The arrangement of the paper is as follows. In Section 2, we briefly describe one of the principal ideas behind our scheme(s), and then present a public key encryption protocol which is secure against the "encryption emulation" attack by a computationally unbounded adversary *who does not know the encrypter's hardware computational limitations.* Then, in Section 3, building on these ideas and combining them with a simple yet subtle trick, we present another protocol which is secure against the "encryption emulation" attack by a computationally unbounded adversary who has complete information on the algorithm(s) and hardware that the sender uses for encryption. More precisely, in our protocol the sender transmits his private bit sequence by encrypting one bit at a time, and the receiver decrypts each bit correctly with probability that can be made arbitrarily close to 1, but not equal to 1. At the same time, the (computationally unbounded) adversary decrypts each bit (by emulating the sender's encryption algorithm) correctly with probability at most $\frac{3}{4}$.

There are essentially no requirements on the sender's computational abilities; in fact, encryption can be done by hand, which can be a big advantage in some situations; for example, a field operative can receive a public key and transmit encrypted information over the phone, without using a computer.

2. Encryption: beta version

We assume that the sender (Bob) is given a presentation Γ (published by Alice) of a group G by generators and defining relators:

$$\Gamma = \langle x_1, x_2, \dots, x_n \mid r_1, r_2, \dots \rangle.$$

No further information about the group G is available to Bob.

Bob is instructed to transmit his private binary sequence to Alice by transmitting a word $u = u(x_1, \dots, x_n)$ equal to 1 in G in place of "1" and a word $v = v(x_1, \dots, x_n)$ not equal to 1 in G in place of "0".

Now we have to specify the algorithms that Bob should use to select his words.

Algorithm "0" (for selecting a word $v = v(x_1, \dots, x_n)$ not equal to 1 in G) is quite simple: Bob just selects a random word by building it letter-by-letter, selecting each letter uniformly from the set $X = \{x_1, \dots, x_n, x_1^{-1}, \dots, x_n^{-1}\}$. The length of such a word should be a random integer from an interval that Bob selects up front, based on his computational abilities.

Algorithm "1" (for selecting a word $u = u(x_1, \dots, x_n)$ equal to 1 in G) is slightly more complex. It amounts to applying a random sequence of operations of the following two kinds, starting with the empty word:

(1) Inserting into a random place in the current word a pair hh^{-1} for a random word h.

(2) Inserting into a random place in the current word a random conjugate $g^{-1}r_i g$ of a random defining relator r_i.

The length of the resulting word should be in the same range as the length of the output of Algorithm "0". We do not go into more details here because they are irrelevant to the main Section 3 of the present paper, i.e., all claims of Section 3 remain valid no matter what algorithm for producing words equal to 1 is chosen, as long as it does not return the empty word. However, we note in passing that for more practical protocols like the one suggested in [5], Algorithm "1" matters a lot, and it definitely deserves further investigation.

Anyway, this is it for Bob; now it is Eve's turn to work. According to the reviewer's recipe quoted in our Introduction, if she wants to use the "encryption emulation" attack, she has to

> ... perform key generations over and over again, each time with fresh randomness, until the public key to be attacked is obtained.

Thus, Eve is building up two lists, corresponding to two algorithms above. Our first observation is that the list that corresponds to the Algorithm "0" is useless to Eve because it is eventually going to contain *all* words in the alphabet $X = \{x_1, \dots, x_n, x_1^{-1}, \dots, x_n^{-1}\}$. Therefore, Eve may just as well forget about this list and concentrate on the other one, that corresponds to the Algorithm "1".

Now the situation boils down to the following: if a word w transmitted by Bob appears on the list, then it is equal to 1 in G. If not, then not. The only problem is: how can Eve possibly conclude that w does *not* appear on the list if the list is infinite? Here our opponent should say "Well, there is no infinity in real life, so the list is actually finite because of Bob's computational limitations". Indeed, whether or not the list in question is finite or infinite is irrelevant to the problem at hand. What matters is that, in theory, Eve does not know a *bound* on the size of the list. Well, then, the opponent might say,

> Already the correctness (no matter if perfect or only with overwhelming probability) of the scheme guarantees now that the corresponding secret key (as obtained by the adversary performing key generation) allows to decrypt illegitimately.

In other words, our opponent suggests that Eve can stop at some point and conclude that $w \neq 1$ with overwhelming probability, just like Alice does. The point however is that this probability may not at all be as "overwhelming" as the probability of the correct decryption by Alice. Compare:

(1) For Alice to decrypt correctly "with overwhelming probability", the probability $P_1(N)$ for a random word w of length N not to be equal to 1 should converge to 1 (reasonably fast) as N goes to infinity.

(2) For Eve to decrypt correctly "with overwhelming probability", the probability $P_2(N, f(N))$ for a random word w of length N produced by the Algorithm "1" to have a *proof* of length $\leq f(N)$ verifying that $w = 1$, should converge to 1 as N goes to infinity. Here $f(N)$ represents Eve's computational capabilities; this function can be arbitrary, but fixed.

We see that the functions $P_1(N)$ and $P_2(N)$ are of very different nature, and any correlation between them is unlikely. We note that the function $P_1(N)$ is generally well understood, and in particular, it is known that in any infinite group G, $P_1(N)$ indeed converges to 1 as N goes to infinity; see Section 5 of [5] for more details on this convergence.

On the other hand, functions $P_2(N, f(N))$ are more complex; note also that they may depend on a particular algorithm used by Bob to produce words equal to 1. The Algorithm "1" described in this section is very straightforward; there are more delicate algorithms that will be discussed in another paper.

Anyway, functions $P_2(N, f(N))$ are currently subject of active research, and in particular, it appears likely that there are groups in which $P_2(N, f(N))$ does not converge to 1 at all, if an algorithm used to produce words equal to 1 is chosen intelligently.

We also note in passing that if in a group G the word problem is recursively unsolvable, then the length of a proof verifying that $w = 1$ in G is not bounded by any recursive function of the length of w.

Of course, in real life, Eve may know a bound on the size of the list based on a general idea of what kind of hardware may be available to Bob; but then again, in real life Eve would be computationally bounded herself. Here we note (again, in passing) that there are groups G with efficiently solvable word problem and words w of length n equal to 1 in G, such that the length of a proof verifying that $w = 1$ in G is not bounded by any tower of exponents in n, see [4].

Thus, the bottom line is: in theory, Eve cannot positively identify the bit that Bob has encrypted by a word w if she just uses the "encryption emulation" attack. In fact, such an identification would be equivalent to solving the *word problem* in G, which would contradict the well-known fact that there are (finitely presented) groups with recursively unsolvable word problem.

It would be nice, of course, if Eve was unable to positively decrypt using "encryption emulation" attacks even if she *did* know Bob's computational limitations. This, too, can be arranged, see the next section.

3. Encryption: trick and treat

Building on the ideas from the previous section and combining them with a simple yet subtle trick, we describe here an encryption protocol with the following features:

(F1) Bob encrypts his private binary sequence by words in a public alphabet X.

(F2) Alice (the receiver) decrypts Bob's transmission correctly with probability that can be made arbitrarily close to 1, but not equal to 1.

(F3) The adversary, Eve, is assumed to have no bound on the speed of computation or on the storage space.

(F4) Eve is assumed to have complete information on the algorithm(s) and hardware that Bob uses for encryption. However, Eve cannot predict outputs of Bob's random numbers generator (the latter could be just coin tossing, say).

(F5) Eve cannot decrypt correctly any bit of Bob's binary sequence with probability $> \frac{3}{4}$ by emulating Bob's encryption algorithm.

This leaves Eve with the only hope: to attack Alice's decryption algorithm or her algorithm for obtaining public keys, but this subject is outside of the scope of the present paper. Here we only discuss the "encryption emulation" attack.

We also have to say up front that the encryption protocol that will be presented in this section is probably not very suitable for commercial applications (such as Internet shopping or banking) due to a large amount of work required from Alice to receive just one bit from Bob. For a commercial implementation of the ideas presented in the previous section, we can recommend the protocol described (with specific parameters) in [5]. That protocol has encryption with a fairly large expansion factor (on the order of 150), but not to the point of being impractical, especially given that the encryption amounts primarily to generating random words of reasonable length, which can be done quite efficiently.

Now we are getting to the protocol description. In one round of this protocol, Bob transmits a single bit, i.e., Alice generates a new public key for each bit transmission.

(P0) Alice publishes two presentations:

$$\Gamma_1 = \langle x_1, x_2, \ldots, x_n \mid r_1, r_2, \ldots \rangle$$

$$\Gamma_2 = \langle x_1, x_2, \ldots, x_n \mid s_1, s_2, \ldots \rangle.$$

One of them defines the trivial group, whereas the other one defines an infinite group, but only Alice knows which one is which. Bob is instructed to transmit his private bit to Alice as follows:

(P1) In place of "1", Bob transmits a pair of words (w_1, w_2) in the alphabet $X = \{x_1, x_2, \ldots, x_n, x_1^{-1}, \ldots, x_n^{-1}\}$, where w_1 is selected randomly, while w_2 is selected to be equal to 1 in the group G_2 defined by Γ_2 (see Algorithm "1" in the previous section).

(P2) In place of "0", Bob transmits a pair of words (w_1, w_2), where w_2 is selected randomly, while w_1 is selected to be equal to 1 in the group G_1 defined by Γ_1.

Under our assumptions (F3), (F4) Eve can identify the word(s) in the transmitted pair which is/are equal to 1 in the corresponding presentation(s), as well as the word, if any, which is not equal to 1. There are the following possibilities:

(1) $w_1 = 1$ in G_1, $w_2 = 1$ in G_2;
(2) $w_1 = 1$ in G_1, $w_2 \neq 1$ in G_2;
(3) $w_1 \neq 1$ in G_1, $w_2 = 1$ in G_2.

It is easy to see that the possibility (1) occurs with probability $\frac{1}{2}$ (when Bob wants to transmit "1" and G_1 is trivial, or when Bob wants to transmit "0" and G_2 is trivial). If this possibility occurs, Eve cannot decrypt Bob's bit correctly with probability $> \frac{1}{2}$. Indeed, the only way for Eve to decrypt in this case would be to find out which presentation Γ_i defines the trivial group, i.e., she would have to attack Alice's algorithm for obtaining a public key, which is outside of the scope of the present paper. Here we just note, in passing, that there are many different ways to construct presentations of the trivial group, some of them involving a lot of random choices. See e.g. [3] for a survey on the subject.

In any case, our claim (F5) was that Eve cannot decrypt Bob's bit correctly with probability $> \frac{3}{4}$ by emulating Bob's encryption algorithm, which is obviously true in this scheme since the probability for Eve to decrypt correctly is, in fact, precisely $\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot 1 = \frac{3}{4}$. (Note that Eve decrypts correctly with probability 1 if either of the possibilities (2) or (3) above occurs.)

Our abrasive opponent may say that $\frac{3}{4}$ is a rather high probability of illegitimate decryption, even though this is just for one bit. Recall however that we are dealing with computationally unbounded adversary, while Bob can essentially do his encryption by hand! All he needs is a generator of uniformly distributed random integers in the interval between 1 and $2n$ (the latter is the cardinality of the alphabet X). Besides, note that with the probability of correctly decrypting one bit equal to $\frac{3}{4}$, the probability to correctly decrypt, say, a credit card number of 16 decimal digits would be on the order of 10^{-7}, which is comparable to the chance of winning the jackpot in a lottery. Of course, there are many tricks that can make this probability much smaller, but we think we better stop here because, as we have pointed out before, our focus here is on the new paradigm itself.

References

[1] J.-C. Birget, S. Magliveras, M. Sramka, *On public-key cryptosystems based on combinatorial group theory*, preprint.
http://eprint.iacr.org/2005/070

[2] M. R. Magyarik, N. R. Wagner, *A Public Key Cryptosystem Based on the Word Problem.* CRYPTO 1984, 19–36, Lecture Notes in Comput. Sci. **196**, Springer, Berlin, 1985.

[3] A. D. Myasnikov, A. G. Myasnikov, V. Shpilrain, *On the Andrews-Curtis equivalence*, Contemp. Math., Amer. Math. Soc. **296** (2002), 183–198.
[4] A. N. Platonov, *An isoparametric function of the Baumslag-Gersten group.* (Russian) Vestnik Moskov. Univ. Ser. I Mat. Mekh. 2004, no. 3, 12–17; translation in Moscow Univ. Math. Bull. 59 (2004), no. 3, 12–17 (2005).
[5] V. Shpilrain and G. Zapata, *Using decision problems in public key cryptography*, preprint. http://www.sci.ccny.cuny.edu/~shpil/wppkc.pdf

Department of Mathematics, The City College of New York, New York, NY 10031
E-mail address: denis.osin@gmail.com, shpil@groups.sci.ccny.cuny.edu
http://www.sci.ccny.cuny.edu/~shpil