Can one design a signature scheme based on error-correcting codes?

Jacques Stern
Laboratoire d'Informatique, École Normale Supérieure

Abstract. In this note, we show that the signature scheme based on error-correcting codes which has been proposed during the ASIACRYPT'94 conference and appears in this volume (see [1]) is not secure. The attack involves gathering a few hundred signatures. From then on, only elementary linear algebra is used.

1 The Use of Error-Correcting Codes in Cryptography.

Since the appearance of public key cryptography, there has been a continuous line of research aiming at the discovery of alternatives to the standard techniques based on arithmetical properties of large numbers. In order to achieve such a program, error-correcting codes seemed quite attractive. As soon as 1978, a public key cryptosystem based on the use of Goppa codes was proposed by Mc Eliece in [3]. This cryptosystem is still standing. Concerning the problem of identification, a scheme was proposed by the author in [4]. This time, random codes with no specific structure were used. It is of course possible to turn our authentication scheme into a signature scheme by the standard technique that replaces queries from the verifier by values suitably obtained from the message to sign together with the initial commitments from the prover (see [2]). Unfortunately, this yields rather long and there are a bit impractical signatures. All attempts to design a signature scheme based on error-correcting codes and having reasonable signature length have failed (see the references [1]). We will show that the same is true with the scheme proposed in the present volume ([1]).

2 The Attack.

The scheme appearing in [1] is a bit intricate. Actually, it is not really necessary to understand the technical details of key generation, signature generation and signature check in order to follow our attack. We will focus on the weak part of the scheme which lies in the following equation, literally taken from [1].

\[ s_j = \{e_j \oplus f(m_j, e_j)W_A \oplus e_jW_A^{-1}G_A\}P_A \]

In the above, \( W_A \) and \( G_A \) are secret \((k \times n)\) matrices, and \( P_A \) is a secret \((n \times n)\) matrix. \( W_A^{-1} \) is a known \((n \times k)\) matrix. Also, \( s_j, e_j, z_j \) and \( f(m_j, e_j) \) are known row vectors, the first three of dimension \( n \) and the last one of dimension \( k \).\( s_j, e_j, z_j \)
$z_j$ are part of the signature and $y_j$ is computed during signature check, as well as $f(m_j, e_j)$.

We write $y_j$ in place of $z_j W_A^{T R}$. Of course all matrices and vectors are over the two element field.

Now, the above equation provides a known linear equation between the coefficients of the unknown matrices $P_A$, $W_A P_A$ and $G_A P_A$. More precisely, each signature gives $n$ scalar equations since both sides of the equation are $n$-bit vectors. Since there are $n(2n + k)$ unknowns, it should take something like $2k + n$ signatures to solve the system. Once this is done, $P_A$, $W_A P_A$ and $G_A P_A$ are known and new signatures can be easily manufactured.

It turns out that the above argument is not completely correct. In order to get the requested conclusion, we need to have a sample of vectors $z_j = \{e_j, f(m_j, e_j), y_j\}$ that generate the whole $2k + n$-dimensional space. There is a subtle point here: although the row vectors $y_j$ and $f(m_j, e_j)$ are basically random, the vectors $e_j$, are randomly chosen among $n$-bit vectors with a prescribed number $t$ of ones (error patterns). Thus, if $t$ is even, there is no way the $z_j$ can generate the whole space since they are all included in the hyperplane consisting of vectors with an even number of ones among the first $n$ bits.

Let $F_N$ be the space generated by the $z_j$ a $N$ signatures have been collected. If $F_N$ is not the entire space, let $H_N$ be a hyperplane containing $F_N$ and let $h = 0$ be a cartesian equation of this hyperplane. Now if the linear function $h$ has some non zero coefficient among the last $2k$ ones, then, it is easily seen that with probability $1/2$, the next signatures takes you outside $H_N$. This shows that we will very quickly come up with a situation where $h$ has its $2k$ trailing coefficients equal to zero. We call such a situation favourable. At this point, the $z_j$ generate a subspace of the form $E \times F_2^{2k}$ and the linear application corresponding to the matrix obtained by appending the blocks $P_A$, $W_A P_A$ and $G_A P_A$ is known on this subspace. Especially $P_A$ is known on $E$ and $W_A P_A$ and $G_A P_A$ are completely determined.

Note that, in the above situation, it is already possible to sign whenever a vector of weight $t$ can be found in $E$: this can be easily checked by going into the construction of $[1]$. Also note that, no such vector can be found in $E$ the next signature will provide one. This enables us to estimate an upper bound on the numbers of signatures required in order to issue a new unauthorized signature. A $N$ signatures, the probability that a favorable situation has not been reached is bounded above by the probability of getting less than $2k + n$ heads a time tossing a fair coin $N$ times. Using classical estimates on binomial coefficients, this is equivalent to $2^N H_k(N-1)$, where $\lambda = \frac{2k}{n}$ and $H_2(x) = -x \log_2 x - (1 - x) \log_2 (1 - x)$. This upper bound quickly decreases when $N$ increases. In $[1]$ the following dimensions are suggested

- $n = 128$ and $k = 65$ in which case $2k + n = 258$. The above probability is bounded by $0.0027$ or $N \approx 600$.
- $n = 256$ and $k = 152$ in which case $2k + n = 508$. The above probability is
bounded by 0.069 or $N = 1200$.

Thus, as announced, a few hundred signatures are enough to break the scheme. Furthermore, our analysis is overly pessimistic and a favorable situation is presumably reached much earlier than what is suggested by the bounds.

3 Can the Scheme be Rescued?

We do not think the scheme can be rescued. Of course, suggestions can be made in order to make the attack slightly more difficult: for example the error vector $\epsilon_i$ could be computed from say $m_j$ by deterministic hashing. Note that this suggestion does not appear in [1] (the same way the discussion about odd and even weights was not present in the paper included in the proceedings). For this reason, we will only offer a qualitative analysis.

As the numbers of known signatures increases, the vector space denoted by $E$ in our above analysis grows. We distinguish two cases

1. If $t$ (the weight of error vectors) is odd, then $E$ ultimately covers the whole space
2. If $t$ is even, then $E$ ultimately covers the whole hyperplane consisting of vectors of even weight

In both cases, any message can ultimately be signed, even or the tentatively “repaired” version of the scheme. Note that, in some unlucky cases, the convergence can be a bit slow, or example we get “stuck” in some hyperplane whose equation is a linear functional with a single variable. Then, it will take an average of $n/t$ signatures to get out. Again, a quantitative worst-case analysis based on these unlucky cases would be overly pessimistic and the generic case should converge ast enough.

References


This article was processed using the \TeX macro package with LLNCS style