Using Low Density Parity Check Codes in the McEliece Cryptosystem

Chris Monico
Department of Mathematics
University of Notre Dame
Notre Dame, Indiana 46556
e-mail: cmonico@nd.edu
http://www.nd.edu/~cmonico/

Joachim Rosenthal[1]
Department of Mathematics
University of Notre Dame
Notre Dame, Indiana 46556
email: Rosenthal.1@nd.edu
http://www.nd.edu/~rosen/

Amin Shokrollahi
Bell Labs
600 Mountain Avenue
Murray Hill, NJ 07974
email:
amin@research.bell-labs.com

Abstract — **We examine the implications of using a Low Density Parity Check Code (LDPCC) in place of the usual Goppa code in McEliece's cryptosystem. Using a LDPCC allows for larger block lengths and the possibility of a combined error correction/encryption protocol.**

I. Introduction

If one wishes to use a LDPCC in the McEliece system, there are several ways to proceed. An efficient way seems to be the following:

As usual, suppose Bob wishes to send Alice a secure message over an insecure channel. Alice chooses a random $(n-k)\times n$ sparse parity check matrix, H, for a binary LDPCC, $\mathcal{C}$, that admits decoding of any pattern of t or fewer errors with, say, belief propagation. She also randomly chooses sparse invertible matrices $S \in GL(k, \mathbb{F}_2)$ and $T \in GL(n-k, \mathbb{F}_2)$. She then calculates $\tilde{H} := TH$ and has keys:

Public Key: $(\tilde{H}, S, t)$

Private Key: (H, T)

Now, if Bob wants to send Alice the message m, he first computes the generator matrix, G, for the code $\mathcal{C}$ in row reduced echelon form, and then computes $\tilde{G} = S^{-1}G$. He then applies the encryption map:

$$m \mapsto m\tilde{G} + e =: y$$

where e is a random error vector of weight at most t. Alice's decryption procedure is then as follows: Since $\tilde{G}$ and G define the same code, $\mathcal{C}$, she can use H to decode the word y to $m\tilde{G} = mS^{-1}G$. Since G is in row reduced echelon form, this reveals mS^{-1} in the k coordinates of $m\tilde{G}$ in which G has only one nonzero entry (i.e., the *systematic coordinates* of G). Right multiplication by S finally recovers Bob's message m.

This seems relatively efficient because the keys consist of sparse matrices, allowing considerable compression. Hence, one could have key sizes comparable to those of a $(1024, 512)$ McEliece system, but for a code of size $(16384, 8192)$.

II. Security

The security of this system is based on two observations:

- If T is chosen with the proper parameters, $\tilde{H}$ will most likely not admit decoding with, e.g. belief propagation, for the correction of up to t errors.
- It *seems* difficult to recover a matrix, H', equivalent to $\tilde{H}$ that admits decoding with, e.g. belief propagation, for the correction of up to t errors. In particular it seems difficult to recover the specific degree structure of the parity check matrix H.

However, a simple observation shows that if T is chosen too sparsely, this latter task is *not* difficult. In what follows, if $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ are two vectors over $\mathbb{F}_2$, $u * v := (u_1v_1, \ldots, u_nv_n)$ denotes the intersection of the binary vectors u, v. This is a vector whose support is exactly $\mathrm{supp}(u) \cap \mathrm{supp}(v)$. Equivalently, it can be considered as the 'AND' of u and v.

Let $h_1, \ldots, h_{n-k}$ denote the row vectors of H and $\tilde{h}_1, \ldots, \tilde{h}_{n-k}$ the row vectors of $\tilde{H}$. Notice that the h_i are sparse vectors and each $\tilde{h}_j$ is a linear combination of the h_i. Furthermore, if T is sparse, each $\tilde{h}_j = h_{j_1} + \cdots + h_{j_{w_j}}$ with the w_j small. That is, each $\tilde{h}_j$ is a linear combination of a small number of rows of H. If the w_j are too small (i.e., T is too sparse), then with reasonable probability one has that $\tilde{h}_j * h_{j_m} = h_{j_m}$ for many of the $1 \le j \le n-k$, $1 \le j_m \le j_{w_j}$. In this case, since each h_{j_m} appears in several of the $\tilde{h}_j$, we can, with non-negligible probability, find j_1, j_2 such that

$$\tilde{h}_{j_1} * \tilde{h}_{j_2} = h_i$$

for some i. Thus, in time $k(k-1)/2$, we can recover some of the original rows of H by computing the intersection of all pairs of rows, checking to see if the intersection is in $Rowsp(\tilde{H})$. Having found some of the original rows, we can determine, with high probability, which of the $\tilde{h}_j$ have these rows as components in their linear combinations. We thus subtract each original row from the $\tilde{h}_j$ that have many nonzero coordinates in common with it. Then go back to computing the intersection of all pairs of rows again, and keep repeating until we've found sufficiently many original rows to allow decoding.

III. Conclusion

Empirical evidence has shown this attack and some variants of it, to be effective enough that we consider this system insecure unless T is chosen to be dense. Thus, there seems to be no advantage to using a parity check matrix as the public key. However, this system is still of possible interest in the following case: If one is using a LDPCC for error correction, some security can be added at very little extra cost.

References

[1] R.J. McEliece, "A Public Key Cryptosystem Based on Algebraic Coding Theory," *Technical Report DSN Progress Report #42-44*, Jet Propulsion Laboratory, Pasadena, California, 1978.

[2] R.G. Gallager, "Low Density Parity Check Codes," *MIT Press*,Cambridge, MA, 1963.

[3] T. Richardson, M.A. Shokrollahi, and R. Urbanke. Design of provably good low-density parity check codes. *IEEE Trans. Inform. Theory (submitted)*, 1999.

[1]The research is supported in part by NSF grant DMS-96-10389.

0-7803-5857-0/00/$10.00 ©2000 IEEE.