Spamming the code offset method

Spamming the Code Offset Method

Niels de Vreede Boris Škorić
Eindhoven University of Technology
Department of Mathematics and Computer Science
5600 MB Eindhoven, The Netherlands
n.d.vreede@tue.nl b.skoric@tue.nl

Abstract

This is an extended abstract of the work published in [5].
We propose an extension of the Code Offset Method, the 'mother of all Secure Sketches', in which we hide the error correction data in a large list of random decoy values. Secure Sketches are an important ingredient for building privacy-preserving biometric databases. Our scheme, the "Spammed Code Offset Method" (SCOM), improves the level of privacy at the cost of extra storage or computational requirements.

1 Introduction

1.1 Helper Data Schemes

Helper Data Schemes (HDSs) are a security primitive that allows for reliable extraction of secret information from noisy data, e.g. biometric data or data from a physical unclonable function (PUF). They make use of a special form of redundancy information, 'helper data', to correct measurement noise. HDSs can be used to construct e.g. privacy-preserving biometric databases.
The functionality of a generic HDS is shown in Fig. 1. There is an enrollment phase and a reconstruction phase. The enrollment procedure Enroll takes as input a measurement value X and optionally a random value R. The output is helper data W and secret data S. The reconstruction procedure Rec takes the helper data W and a fresh sample X', which is a noisy version of X, and produces $\hat{S}$, which is an estimate of S. If the noise between X and X' is not too large, then $\hat{S} = S$. Furthermore, W should not reveal too much information about the secret, ideally none at all. Secrecy of S is preserved even if W is stored publicly. It is always assumed that attackers have access to W.
Two special types of HDS with additional properties are the fuzzy extractor and secure sketch. A fuzzy extractor requires that the secret is uniformly distributed. For a secure sketch, the secret is identical to the measured value, $S = X$, and no uniformity is required.

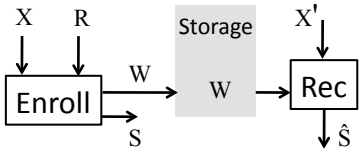

Figure 1: *A generic helper data scheme.*

1.2 The Syndrome-Only Code Offset Method

One of the first introduced helper data schemes is the Code Offset Method (COM)[4, 2]. The COM employs a linear error correcting code to compensate for measurement noise. Below we describe the *Syndrome-Only* COM: a modified version of the COM that is more suitable for our purposes. It additionally requires the existence of an efficient syndrome decoder. The Enroll procedure consists of nothing more than computing a syndrome (Syn),

$$W = \mathsf{Syn}\, X.$$

This construction is a secure sketch, i.e., the secret is the measurement value itself. The X is reconstructed from W and a sample X' as follows:

$$\hat{X} = X' \oplus \mathsf{SDec}(W \oplus \mathsf{Syn}\, X').$$

The linearity of the code ensures that, if X' is sufficiently close to X, then $\hat{X}$ will be equal to X.
In a biometric database, the values stored for each enrolled person would be W and a hash of X. As long as X given W has sufficient entropy, it is infeasible to guess X from the enrolled data.

2 Adding Fake Helper Data

Consider an attacker who tries to guess X given the helper data W. Consider a low-entropy source X, such that the attacker's task is difficult but feasible. We propose to increase the attacker's workload by hiding the real helper in a list of fake entries. If we store m helper data items, only one of which is real, the attacker's average workload increases by a factor of about $m/2$. (For very large values of m, the attacker is even forced to ignore the helper data altogether.) We refer to this technique as *spamming*. The technique can be applied in any HDS, as long as there exists an efficient way to select the true helper data given X'. For the (syndrome only) COM, this is achieved by employing a *Low Density Parity Check* (LDPC) code.
The idea of adding chaff data to hide information is not new [3, 1], but, whereas previous work considered adding chaff points directly to the stored feature vectors, we are the first to apply chaffing in the helper data domain. Our data hiding technique allows us to make more effective use of the source entropy, but it comes at the cost of increased storage requirement or computational workload. An advantage of adding spam in the helper data domain (instead of e.g. X-space) is that it allows for a very precise security analysis.

2.1 The Enrollment and Reconstruction Algorithms

We modify the enrollment procedure such that W is replaced by a list $\mathbf{\Omega}$ of length m. The list $\mathbf{\Omega}$ consists of $m-1$ fake items and W hidden at random secret position Z. One way to do this [5] is to generate the fake items in $\mathbf{\Omega}$ according to the prob. distribution of $\mathsf{Syn}\, X$ and then store the full list. However, this blows up the storage requirements by a factor m. We present an alternative in which $\mathbf{\Omega}$ is generated 'on the fly' from a seed S. We call this the 'generative' Spammed Code Offset Method. The scheme needs a one-way function f and a fast Pseudo Random Number Generator (PRNG) γ that generates uniform bit strings of same length as our code's syndrome. By $\gamma^i(S)$ we denote the i-th string derived from seed S.
Enrollment

1. Measure X.

2. Compute $W = \mathsf{Syn}\, X$.
3. Uniformly draw index $Z \in \{1, \ldots, m\}$.
4. Uniformly draw seed S.
5. Compute mask $B = W \oplus \gamma^Z(S)$.
6. Compute $G = f(S \| B \| X)$.
7. Store public data $P = (S, B, G)$.

We can think of the list $\{B \oplus \gamma^i(S)\}_{i \in \{1,\ldots,m\}}$ as the list $\mathbf{\Omega}$ of fake helper data which contains the real W at position Z.
For clarity, we present a simplified version of the reconstruction algorithm; see [5] for more details. The reconstruction algorithm inspects the Hamming distance d_H between the syndrome of the measured value X' and the candidate helper data items and only carries out the expensive decoding step if the Hamming distance is below a threshold θ.
Reconstruction

1. Read $P' = (S', B', G')$.
2. Measure X'.
3. Compute $M = B' \oplus \mathsf{Syn}\, X'$
4. For $i = 1$ to m:
   (a) If $d_H(M, \gamma^i(S')) \geq \theta$, then next i.
   (b) Compute $\hat{X} = X' \oplus \mathsf{SDec}(M \oplus \gamma^i(S'))$.
   (c) If $G' = f(S' \| B' \| \hat{X})$ then return $\hat{X}$.
5. If the loop is exhausted, then return failure.

Because we use a LDPC code, a small Hamming distance between X' and X implies a small Hamming distance between $\mathsf{Syn}\, X'$ and $\mathsf{Syn}\, X$. For example, a column weight 3 LDPC code ensures that every bit flip between X' and X causes at most three bit flips between $\mathsf{Syn}\, X'$ and $\mathsf{Syn}\, X$.

3 Security Analysis

We express the security properties of our scheme in terms of Shannon entropy H and mutual information I. We start with a general theorem that holds for any method of inserting fake helper data.

Theorem 1 *Let $\mathbf{\Omega}$ be the list of fake helper data in which the real helper data W are inserted at a random position Z. Then the entropy improvement compared to the plain COM is given by*

$$\begin{aligned} \mathsf{H}(X|\mathbf{\Omega}) - \mathsf{H}(X|W) &= \mathsf{H}(W|\mathbf{\Omega}) \\ &= \underbrace{\mathsf{H}(Z)}_{\substack{\text{entropy} \\ \text{gain}}} - \underbrace{\mathsf{H}(Z|W\mathbf{\Omega})}_{\substack{\text{collision} \\ \text{penalty}}} - \underbrace{I(Z;\mathbf{\Omega})}_{\substack{\text{distribution} \\ \text{mismatch} \\ \text{penalty}}}. \end{aligned} \tag{1}$$

In the first term of (1), we recognize the entropy gained from hiding the real helper data at a random position in the list. There are also two clearly interpretable penalty terms in (1). The 'collision penalty' $\mathsf{H}(Z|W\mathbf{\Omega})$ increases with m. It becomes non-negligible when $\mathbf{\Omega}$ contains so many entries that it becomes likely that there exist entries with the same value; then even knowing W and $\mathbf{\Omega}$ does not fix Z.
The 'distribution mismatch penalty' occurs when the fake entries in $\mathbf{\Omega}$ do not look statistically the same as W; then some information about Z can be obtained already from inspecting $\mathbf{\Omega}$.
Next, we provide two lower bounds on the entropy. These bounds follow from (1). Theorem 2 is relevant for the case in which the fake helper data is distributed identically to the real helper data; Theorem 3 is relevant for the generative SCOM.

Theorem 2 *If the distribution of the fake helper data is identical to the distribution of the real helper data and the index Z is drawn uniformly, then*

$$\mathsf{H}(X|\mathbf{\Omega}) - \mathsf{H}(X|W) \geq \log m - \frac{m-1}{\ln 2} \sum_w \left(\Pr[W = w]\right)^2 . \tag{2}$$

If the fake entries are drawn from the same distribution as W, then the distribution mismatch penalty vanishes. Furthermore, if W is not uniform, then this affects the probability of encountering a collision. This is reflected in the $\sum_w$ term of (2). The summation runs over all possible helper data values. As long as W is not too wildly non-uniform and m is not too large, the $\sum_w$ term is negligible w.r.t. $\log m$.

Theorem 3 *Let $W \in \mathcal{W}$. Let U denote a random variable uniform on $\mathcal{W}$. If the fake helper data and the index Z are drawn uniformly, then*

$$\mathsf{H}(X|\mathbf{\Omega}) - \mathsf{H}(X|W) \geq \log m - \frac{m-1}{|\mathcal{W}| \ln 2} - (1 - \frac{1}{m})[D(W\|U) + D(U\|W)], \tag{3}$$

where D is the Kullback-Leibler divergence.

Here the collision penalty has a simple from since it pertains to collisions of uniform variables. In both Theorem 2 and Theorem 3 we see that for $m \ll |\mathcal{W}|$ the improvement in the entropy of X given the public information is approximately $\log m$, as one would intuitively expect.

References

[1] Claude Barral. *Biometrics & Security: Combining Fingerprints, Smart Cards and Cryptography.* PhD thesis, École Polytechnique Fédérale de Lausanne, Switzerland, 2010.

[2] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[3] Ari Juels and Madhu Sudan. A fuzzy vault scheme. In *Proc. IEEE ISIT*, pages 408–410, 2002.

[4] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Proc. ACM CCS*, pages 28–36, 1999.

[5] Boris Škorić and Niels de Vreede. The Spammed Code Offset Method. *IEEE Transactions on Information Forensics and Security*, 9(5):875–884, 2014.