Towards full collusion resistant
ID-based establishment of pairwise keys

Oscar Garcia-Morchon\textsuperscript{1}, Ludo Tolhuizen\textsuperscript{1}, Domingo Gomez\textsuperscript{2}, Jaime Gutierrez\textsuperscript{2}

\textsuperscript{1} Philips Group Innovation, Research
High Tech Campus 34, Eindhoven, Netherlands
oscar.garcia@philips.com, ludo.tolhuizen@philips.com

\textsuperscript{2} University of Cantabria
Avda. de los Castros s/n, Santander, Spain
domingo.gomez@unican.es, jaime.gutierrez@unican.es

Abstract. Usually a communication link is secured by means of a symmetric-key algorithm. For that, a method is required to securely establish a symmetric-key for that algorithm. This old problem is still relevant and of paramount importance both in existing computer networks and new large-scale ubiquitous systems comprising resource-constrained devices. Identity-based pairwise key agreement allows for the generation of a common key between two parties given secret keying material owned by the first party and the identity of the second one. However, existing methods are prone to collusion attacks. In this paper we discuss a new class of key establishment scheme aiming at full collusion resistant identity-based symmetric-key agreement and propose a specific scheme, the HIMMO algorithm, relying on two design concepts: Hiding Information and Mixing Modular Operations. Collusion attacks on schemes from literature cannot readily be applied to HIMMO. Also, the simple logic of the HIMMO algorithm allows for very efficient implementations in terms of both speed and memory. Finally, being an identity-based symmetric-key establishment scheme, HIMMO allows for efficient real-world key exchange protocols.

Key words: Key distribution and establishment, polynomials, identity-based cryptography

1 Introduction

This paper deals with the classical problem of key establishment. As in previous works [4],[2],[7], we focus on an identity-based (ID-based) scheme for symmetric-key agreement between pairs of devices in a network. That is, each node in the network has an identifier, and a trusted third party (TTP) provides it with secret keying material - linked to the device identifier - in a secure way. A node that wishes to communicate with another node uses its own secret keying material and the identity of the other node to generate a common pairwise key.

Existing ID-based symmetric-key agreement schemes are prone to collusion attacks: secret keying material of various nodes can be combined in order to obtain information on the secret key generated by a pair of (other) nodes. This combining can be performed by colluding legitimate owner(s) of the nodes, or by an attacker who has compromised some nodes and obtained their secret keying material. Existing schemes [4],[2],[7] allow for efficient collusion attacks (see Section 2). These efficient collusion attacks imply that it is infeasible to prevent successful attacks by relatively few colluding devices unless much secret keying material is stored in each node, which may be problematic in real-world applications since it increases CPU and storage needs.

This paper discusses a new class of ID-based key establishment schemes allowing for efficient operation – with respect to the amount of stored keying material and key computation time, which is especially relevant for resource-constrained devices – while it is based on mathematical problems for which the collusion attacks on the schemes from literature cannot readily be applied. We hope that our scheme, the HIMMO algorithm, and its underlying design principles can be a step towards full collusion resistant identity-based establishment of symmetric-keys.
Definition 1 (Full collusion resistant) An identity-based symmetric-key establishment scheme is full collusion resistant if for any set of colluding nodes no bit of a key shared by non-colluding nodes can be guessed with a probability higher than 1/2 in polynomial time.

The rest of this paper is organized as follows. In Section 2 we give an overview of related work. In Section 3 describes our HIMMO algorithm. In Section 4 we discuss the design principles and underlying mathematical problems. Finally, we present our conclusions in Section 5.

2 Previous identity-based symmetric-key distribution schemes

Matsumoto and Imai [4] give a nice description of the key distribution problem, and provide a solution that serves as a base for many other schemes from literature. They propose that a trusted third party (TTP) chooses a secret function $f(x, y)$ that is symmetric, that is, $f(x, y) = f(y, x)$. The variables $x$ and $y$ are taken from a set of node identifiers $I$, and the output from $f$ is the key. The secret key material for the node with identifier $\eta$ is a function $KM_\eta(y)$ which is such that $KM_\eta(\eta') = f(\eta, \eta')$ for all $\eta'$. As $f$ is symmetric, it is guaranteed that the keys generated by two nodes for communicating with each other are equal.\footnote{Matsumoto and Imai in fact consider the more general situation that any group of $t$ nodes must generate a common key; we restrict ourselves to the case $t = 2$.}

In [2], Blundo et al. choose the secret function $f(x, y)$ to be a symmetric polynomial over a finite field of degree $\alpha$ in each variable; the identifiers are considered as field elements as well. Blundo et al. show that their scheme offers information-theoretic security as long as an attacker knows the secret keying material of $\alpha$ or less nodes. However, $\alpha + 1$ colluding nodes can obtain the root keying material by simple Lagrange interpolation.

In order to avoid the simple interpolation attack, Zhang et al. [7] proposed a "noisy" version of the scheme of Blundo et al. [2]. Their basic idea is to provide node $\eta$ with a polynomial $KM_\eta(x)$ that is "close" to, but not exactly the same as $f(x, \eta)$. Nodes $\eta$ and $\eta'$ can compute $KM_\eta(\eta')$ and $KM_{\eta'}(\eta)$ as before; these values are no longer equal, but because they are close they can be used to generate a shared key. We now describe the main steps:

- The TTP chooses a random symmetric, bivariate polynomial $f(x, y) \in \mathbb{Z}_p[x, y]$ of degree $\alpha$ in each variable and a noise bound $r$ with $r < p$. It also chooses at random univariate "noise" polynomials $g(y)$ and $h(y)$ of degree $\alpha$ over $\mathbb{Z}_p$. Next, it determines

$$\mathcal{N} := \{\eta \in \mathbb{Z}_p : g(\eta), h(\eta) \in [0, r]\}$$

Each node each given an identifier from $\mathcal{N}$. For each node $\eta \in \mathcal{N}$, the TTP chooses a random bit $b_\eta$ and provides node $\eta$ the univariate polynomial:

$$KM_\eta(x) = f(x, \eta) + b_\eta g(x) + (1 - b_\eta) h(x).$$

- A node $\eta$ wishing to communicate with node $\eta'$ computes $KM_\eta(\eta')$ and takes its $\ell - r$ most significant bits as key (where $\ell$ is such that $2^{\ell - 1} < p \leq 2^\ell$). It sends $h(KM_\eta(\eta'))$ to node $\eta'$, where $h$ is an hash-function. Node $\eta'$ computes three numbers, namely $KM_{\eta'}(\eta), KM_{\eta'}(\eta) + 2r$ and $KM_{\eta'}(\eta) - 2r$, and takes as key the $\ell - r$ most significant bits of the number for which the hash-value agrees with the received hash-value $h(KM_\eta(\eta'))$.

Albrecht et al. [1] designed an efficient collusion attack on the scheme of Zhang et al. based on error-correcting techniques, that works if the $4\alpha + 1$ nodes collude. They also provide an attack that works with $3\alpha$ colluding nodes, but has time complexity $O(r)$. Then, they suggested a generalized scheme based on adding more noise:
The TTP also chooses a natural number $u$ such that $4ur < p$ and, for each node $\eta \in \mathcal{N}$, integers $a_\eta, b_\eta$ and $c_\eta$ such that $a_\eta, b_\eta \in [-u,u]$ and $c_\eta \in [-ur,ur]$, and gives node $\eta$ the univariate polynomial:

$$KM_\eta(x) = f(x,\eta) + a_\eta g(x) + b_\eta h(x) + c_\eta.$$  

They also provided an attack on this new cryptography protocol of time complexity $O(\alpha^3 + 8\alpha u^3)$, and requiring only $\alpha + 3$ compromised nodes. Their attack consists of two steps. In the first step, by means of linear algebra methods, they recover the linear vector space generated by the univariate polynomials $g(x)$ and $h(x)$. In the second step, they use lattice reduction techniques to recover $f$, knowing the polynomials $g$ and $h$.

3 The HIMMO Algorithm

In this section, we describe our HIMMO algorithm for ID-based symmetric-key establishment. It relies on two new design principles:

1. **Hiding of information** by adding noise that is completely independent and random, for each node. This is similar to what is done by Zhang et al. [7], but they have only two possible noise contributions (the noise polynomials $g$ and $h$, see previous section).

2. **Mixing of modular operations** by using $m$ symmetric bi-variate polynomials with coefficients in the integers modulo $p_i$ for generating the secret keying material.

A key difference with all previous schemes [2], [7], [1] is that the modules $p_1, \ldots, p_m$ are kept secret and are only known to the TTP, not to the nodes. The nodes do know, however, that each module differs a multiple of $2^b$ from a known constant $N$.

In our description, we use the following notation. For each real $x$, we denote by $\lfloor x \rfloor$ the value of $x$ rounded downwards to the closest integer, that is,

$$\lfloor x \rfloor = \max\{m \in \mathbb{Z} \mid m \leq x\}.$$  

For integer $a$ and integer $p \geq 2$, we denote by $\langle a \rangle_p$ the remainder of dividing $a$ by $p$. Stated differently,

$$0 \leq \langle a \rangle_p \leq p - 1 \text{ and } a \equiv \langle a \rangle_p \mod p.$$  

3.1 Description

The operation of our ID-based symmetric-key establishment scheme comprises three phases:

1. **System initialization**

The TTP selects a private positive integer $m$, and three public positive integers $b, N$ and $\alpha$ satisfying:

$$2^{(\alpha+2)b-1} < N \leq 2^{(\alpha+2)b}.$$  

The TTP also generates the following private material:

- $m$ distinct positive integers $p_1, \ldots, p_m$ of the form $p_i = N - 2^b \beta_i$, where $1 \leq \beta_i \leq 2^b - 1$, for $i = 1, \ldots, m$;
- $m$ symmetric bi-variate polynomials $f_1(x,y), \ldots, f_m(x,y)$, all of degree at most $\alpha$ in each variable, such that for $i = 1, \ldots, m$, the polynomial $f_i(x,y)$ has its coefficients in the set $\{0,1,\ldots,p_i-1\}$. 
For $1 \leq i \leq m$, we write
\[ f_i(x, y) = \sum_{j=0}^{\alpha} f_{i,j}(y)x^j \text{ with } f_{i,j}(y) \in \mathbb{Z}_p[y]. \]

2. Node registration: distribution of secret keying material

For each node $\eta \in \{1, \ldots, 2^b - 1\}$, that wants to register, the TTP selects $\alpha + 1$ integers $\epsilon_{\eta,j}$ (the noise) satisfying the following equation:
\[ |\epsilon_{\eta,j}| < 2^{(\alpha+1-j)b-2}, \quad j = 0, \ldots, \alpha. \] (1)

The TTP provides node $\eta$ with the secret keying material coefficients $KM_{\eta,0}, KM_{\eta,1}, \ldots, KM_{\eta,\alpha}$, defined as
\[ KM_{\eta,j} = \left(\sum_{i=1}^{m} (f_{i,j}(\eta))p_i + 2^b \epsilon_{\eta,j}\right)N. \] (2)

3. Operational phase: key agreement

Node $\eta$ generates its key with $\eta'$ as:
\[ K_{\eta,\eta'} = \langle\langle \sum_{j=0}^{\alpha} KM_{\eta,j}\eta'^j \rangle\rangle_{2^b}. \] (3)

With explicit examples, it can be shown that $K_{\eta,\eta'}$ and $K_{\eta',\eta}$ are not necessarily equal. It can be shown, however, that the keys are approximately equal, as described in the following theorem.

**Theorem 1** Let $0 \leq \eta, \eta' \leq 2^b - 1$. Then we have that
\[ K_{\eta,\eta'} \in \{\langle\langle f(\eta') + jN \rangle\rangle_{2^b} \mid -\Delta \leq j \leq \Delta\}, \text{ where } \Delta = 3m + \alpha + 1. \]

In order that devices $\eta$ and $\eta'$ agree on a common key, an additional step is performed. In this step, device $\eta$ to device $\eta'$ the value $h(K_{\eta,\eta'})$, where the function $h$ is such that $h(i) \neq h(K_{\eta,\eta'})$ for each potential key $i$ (as indicated in Theorem 1) different from $K_{\eta,\eta'}$. In this way, $\eta'$ finds the key $K_{\eta,\eta'}$ that is subsequently used to secure communications. An example of such a function $h$ is a hash function like in [7].

4 Design principles of the HIMMO algorithm and discussion

As stated before, our HIMMO algorithm relies on two principles, namely (i) hiding of information and (ii) mixing of modular operations. Both principles further exhibit the feature that only partial knowledge on the used modules is available. This is described below.

4.1 Hiding of information ($m \geq 1$)

In Equation 2, we see that for each key material coefficient $KM_{\eta,j}$, parts of the sum of the polynomial evaluations are hidden by the noisy term $2^b \epsilon_{\eta,j}$. This design concept is related to the so-called Extended Hidden Number Problem (EHNP) [5], which can be stated as follows:

**Problem 1 (EHNP)** Let $p$ be a prime and $b$ a positive integer $2^b < p$. Suppose for many random values $\eta \in \{0, 1, \ldots, p - 1\}$, the value $\langle\langle f(\eta) \rangle\rangle_{2^b}$ is given, where and $f(x) \in F_p[x]$ is an unknown polynomial of known degree $\alpha$. Recover $f(x)$ in polynomial time.
Among other applications, Boneh and Venkatesan in [3] found nice links between the EHNP for \(p_1\) and the security of the Diffie-Hellman Key Exchange protocol. Others interesting generalizations can be consulted in [5]. When \(p_1 = p\) is a prime number, attacks are known, e.g. [6] that work if the number of colluding nodes is sufficiently large.

The main security issue with this design principle is that the usage of a single polynomial does not remove the underlying ring structure because the generated key is approximately equal\(^2\) to the one generated from the original polynomial:

\[
K_{\eta,\eta'} \approx \langle (f_1(\eta, \eta'))_{p_1} \rangle_{2^b} = \langle (f_1(\eta', \eta))_{p_1} \rangle_{2^b} \approx K_{\eta',\eta}
\]

However, existing attacks cannot directly be applied to our scheme with \(m = 1\) if \(p_1\) is secret, as we assumed above. Also, if \(p_1\) would be known, possibly an attack could be derived that requires less colluding nodes than current attacks, using that the identifiers for our scheme are in the relatively small set \(\{1, \ldots, 2^b - 1\}\), while current attacks assume that the identifiers are uniformly distributed on \(\{0, 1, \ldots, p - 1\}\).

### 4.2 Mixing of modular operations (\(m \geq 2\))

In Equation 2, we see (for \(m \geq 2\)) a mixing of modular operations in the sum \(\sum_{i=1}^{m} \langle f_{i,j}(\eta) \rangle_{p_i}\). This part of our scheme is thus related to a problem more formally described below.

**Problem 2 (Mixing of modular operations)** Let \(p_1, \ldots, p_m\) be \(m\) distinct positive integer numbers such that \(p_i = N - \beta_i 2^b\), where \(2^{(\alpha+2)} b - 1 < N \leq 2^{(\alpha+2)} b\) and \(\beta_i < 2^b\). Moreover, for \(i = 1, \ldots, m\), be \(f_i(x) \in \mathbb{Z}_{p_i}[x]\) have degree at most \(\alpha\). For \(\eta\) in \(S = \{1, \ldots, 2^b - 1\}\), we define \(H(\eta) := \sum_{i=1}^{m} \langle f_i(\eta) \rangle_{p_i} \). Given a number \(N_\Sigma\) of pairs \((\nu, H(\nu))\), the problem consists in guessing any bit of \(H(\eta)\) associated to a known input value \(\eta\) with a probability higher than \(1/2\).

**Remark** Problem 2 is further enhanced by the fact that the attacker does not know the modules \(p_1, \ldots, p_m\); all he knows is that each \(p_i\) differs the \(b\) bit unknown integer \(\beta_i\) multiple of \(2^b\) from \(N\).

In order to explain the idea behind this second design principle, we consider a simple special case, \(viz.\) that for \(1 \leq i \leq m\), we have that \(f_i(x, y) = A_i x^\alpha y^\beta\) for some \(A_i \in \{1, \ldots, p_i - 1\}\). Moreover, we take \(N = 2^{b(\alpha+2)} - 1\) and \(\epsilon_{\eta,\alpha} = 0\). We write:

\[
A_i \eta^i = R_{i,\eta}^{(2)} \eta^b(\alpha+2) + R_{i,\eta}^{(1)} \eta^b + R_{i,\eta}^{(0)}
\]

with \(0 \leq R_{i,\eta}^{(0)} \leq 2^b - 1\) and \(0 \leq R_{i,\eta}^{(1)} \leq 2^b(\alpha+1) - 1\).

As \(p_i = 2^{b(\alpha+2)} - \beta_i 2^b - 1\), the single non-zero coefficient \(K_{M,\eta,\alpha}\) of node \(\eta\) is given by

\[
\left\langle \sum_{i=1}^{m} \langle f_{i,j}(\eta) \rangle_{p_i} \right\rangle_N = \sum_{i=1}^{m} \left\langle \langle A_i \eta^i \rangle_{p_i} \right\rangle_N = \sum_{i=1}^{m} \left\langle \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} \right) \eta^b + \left( R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \right) \right\rangle_{p_i} \approx 3
\]

\[
\left\langle \sum_{i=1}^{m} \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} + \left( R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \right)/2^b \right) \right\rangle_{p_i} \approx 3
\]

\[
\left\langle \sum_{i=1}^{m} \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} + \left( R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \right)/2^b \right) \right\rangle_{p_i} \approx 3
\]

In this example, we observe that the modulo computations affect the \(b(\alpha + 1)\) most significant bits of the keying material in a way that is dependent on \(\beta_i\). By adding over \(i\), these \(\beta_i\)-dependencies

---

\(^2\) Equation 3 uses module \(N\), where \(\beta_i << N\) is missing, while here all reductions are module \(p_i\).

\(^3\) The effect of the reduction module \(p_i\) due to carry propagation is limited due to the form of \(p_i\).
are mixed. We also see mixing in the b least significant bits of the keying material, as they depend on the sum of the most and least significant bits of \( A \eta^i \). The nice aspect of the design is that the components originating from different polynomials \( f_i(x,y) \) hide each other so that an attacker can only observe the sum modulo \( N \), learning nothing about the individual components.

Thus, our HIMMO algorithm applies the second design concept by using \( p_i \) with such a form that they introduce non-linear operations when the TTP generates the secret keying material for node \( \eta \) from the secret bivariate polynomials. However, the public modulus \( N \) and the \( p_i \) share a given structure that still allows for the generation of a b bit key by means of Equation 3. Thus, the smart part of the cryptoblock happens in the step in which the TTP generates the keying material shares from the secret root keying material creating a non-linear keying material structure in the most significant bits of the secret keying material coefficients as shown in the specific example in Equation 4. Later, during key establishment only the common terms of \( p_i \) and \( N \) are used so that a common key can be generated mod \( N \), i.e., without requiring knowledge of the secret terms \( \beta_i \). Thus, the resulting b-bit key combines the contributions from all polynomials over different rings:

\[
K_{\eta, \eta'} \approx \left( \sum_{i=1}^{m} \langle f_i(\eta, \eta') \rangle_{p_i} \right)_{2^b} = \left( \sum_{i=1}^{m} \langle f_i(\eta', \eta) \rangle_{p_i} \right)_{2^b} \approx K_{\eta', \eta} 
\]

5 Conclusions

Our HIMMO algorithm addresses the old key establishment problem in a different way bringing many advantages. Operationally, it allows for direct ID-based pairwise key establishment simplifying protocol operation. Computationally, the design concepts relying on polynomials allow for very fast operation with minimal memory needs. From a security point of view, although the design concepts seem to be sound, further analysis is required because they are also fairly new. In particular, the first design concept presents some links to the EHNP, and thus, it might make possible partial security analysis of our scheme. To the best of our knowledge, our second design concept, mixing of the evaluation of polynomials using different modules, has not been explored in literature so far. The task of an attacker with regard to both design concepts is further complicated by the fact that he only has partial knowledge on which modules have been used.

References