# An ultra-lightweight ID-based pairwise key establishment scheme aiming at full collusion resistance

Oscar García-Morchón[1], Ronald Rietman[1], Ludo Tolhuizen[1],
Domingo Gómez-Pérez[2], Jaime Gutiérrez[2], and Santos Merino del Pozo[2]

[1] Philips Group Innovation, Research, Eindhoven, The Netherlands
[2] Univ. of Cantabria, Santander, Spain

**Abstract.** This paper introduces a new key establishment scheme aiming at fully collusion-resistant identity-based symmetric-key agreement. In an identity-based pairwise key agreement scheme, a Trusted Third Party (TTP) manages the system and securely provides any node, e.g., Alice or Bob, with private keying materials. Alice can generate a pairwise key with Bob given her own secret keying material and Bob's identity. The full collusion resistance property would ensure that the scheme remains secure even if arbitrarily many devices collude or are compromised.
Our scheme, the HIMMO algorithm, relies on two design concepts: Hiding Information and Mixing Modular Operations. Hiding information is related to the Noisy Interpolation Problem; the Mixing Modular Operations problem seems to be a new hard problem. We describe our scheme, the security of its underlying design principles and give order of magnitude estimations for secure configuration parameters. For these parameters, we show that our prototypic implementation of HIMMO on the 8-bit CPU ATmega128L can generate 128-bit keys in less than 7 ms based on an algorithm fitting in 428 B and with secret keying materials of size 656 B.

**Keywords:** ID-based symmetric-key generation, collusion resistance, mixing modular operations, noisy interpolation problem.

## 1 Introduction

This paper deals with the classical problem of key establishment. 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.

The key distribution problem - as discussed in this paper - was first described by Matsumoto and Imai [1]. They propose that a TTP chooses a secret function $f(X, Y)$ that is *symmetric*, that is, $f(X, Y) = f(Y, X)$. Each node with identifier $\eta$ receives a secret keying material - a function - $KM_\eta(X) = f(X, \eta)$ such that $KM_\eta(\eta') = f(\eta, \eta')$ for any other $\eta'$. In [2], Blundo *et al.* choose the secret function $f(X, Y)$ to be a symmetric bivariate polynomial over a finite field of degree $\alpha$ in each variable and show that their scheme offers information-theoretic security if an attacker knows the secret keying material of $c$ colluding nodes whenever $c \leq \alpha$. If $c \geq \alpha+1$, an attacker can recover $f(X, Y)$ by means of Lagrange interpolation. Zhang *et al.* [3] proposed a "noisy" version

of [2] aiming at being fully-collusion resistant. This scheme was generalized and broken by Albrecht *et al.* [4] in different ways including error-correcting and lattice techniques. Their idea was to provide node $\eta$ with a polynomial $KM_\eta(X) = f(X,\eta) + s_{1,\eta}p_1(X) + s_{2,\eta}p_2(X)$ of degree $\alpha$ where $p_1(X)$ and $p_2(X)$ are two perturbation polynomials hiding $f(X,\eta)$ and $s_{1,\eta}, s_{2,\eta} \in \{-u-u+1,\ldots,u-1,u\}$. The lattice attack in [4] has time complexity $O(\alpha^3 + 8\alpha u^3)$ and requires only $\alpha+3$ compromised nodes. It consists of two steps. In the first step, by means of linear algebra methods, it recovers the linear vector space generated by the perturbation polynomials. In the second step, it uses lattice reduction techniques to recover $f$, knowing the perturbation polynomials.

This paper introduces a new ID-based key establishment scheme allowing for efficient operation – with respect to both the amount of stored keying material and the 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. To achieve the collusion resistance property, we rely on two design principles based on two mathematical problems. Hiding Information is related to the Noisy Interpolation Problem and used in HIMMO during the key generation phase by outputing only part of a polynomial evaluation. The Mixing Modular Operations problem seems to be a new mathematical problem to the best of our knowledge. HIMMO uses it when the TPP generates the secret keying material by combining polynomial evaluations in different rings such that modular operations are mixed in a smart way, and thus structure is removed making interpolation or lattice attacks rather complex, but still, allowing for ID-based key agreement. Our security analysis and SW implementation show the feasibility of HIMMO that might make possible scenarios as the one in which many devices running a small 8-bit CPU can set-up 128-bit pairwise keys among each other requiring just 7 ms and 1 KB of memory for both algorithm and keying material. Even if an attacker compromised very many devices, the connections between the remaining devices would still be secure.

The rest of this paper is organized as follows. Section 2 describes the HIMMO algorithm and its design principles and rationale. Section 3 presents the security analysis of HIMMO. Next we introduce our implementation and performance results in Section 4. Section 5 discusses configuration parameters and compares with existing methods. Finally, Section 6 concludes this paper.

## 2 The HIMMO algorithm

In HIMMO, the TTP is in charge of initializing and keeping secret the root keying material that is used later for the generation of keying material shares for the nodes. Each pair of nodes can generate a pairwise key based on their respective keying material shares and identities. To ensure the full collusion resistance property, we have to avoid two main types of attacks:

- **Attack/recover the root keying material**: from any set of keying material shares or pairwise keys;
- **Attack/recover a keying material share**: given many compromised pairwise keys.

HIMMO protects against these attacks by relying on two design principles: *Mixing Modular Operations* and *Hiding Information*. Mixing modular operations refers to the mixing of results – by adding over the integers or modulo $N$ – obtained from different polynomials in different rings. We have not found any results in the literature related

Table 1: Notation

| Term | Definition |
|---|---|
| $\lfloor x \rfloor$ | $\max\{m \in \mathbb{Z} \mid m \leq x\}$ |
| $\langle a \rangle_q$ | Integer $x$ such that $0 \leq x \leq q-1$ and $x \equiv a \bmod q$ |
| $\mathbb{Z}_q$ | Ring of integers modulo $q$, where each element is identified with an integer in the range $\{0, \ldots, q-1\}$ |
| $\mathbb{Z}_q[X]$ | Polynomial ring in the variable $X$ with coefficients in $\mathbb{Z}_q$ |
| $N$ | Public modulus |
| $b$ | Public size of the identifiers and pairwise keys |
| $\alpha$ | Public polynomial degree (in each variable) |
| $m$ | Public number of bivariate polynomials comprising the root keying material with $m \geq 2$ |
| $\eta$ | Public node identifier with $\eta = 1, \ldots, 2^b - 1$ |
| $\beta_i$ | Secret positive $b$-bit number with $i = 1, \ldots, m$ |
| $q_i$ | Secret moduli such that $q_i = N - \beta_i 2^b$ with $i = 1, \ldots, m$ |
| $KM_\eta(X)$ | secret keying material function of node $\eta$ |
| $KM_{\eta,j}$ | $KM_\eta(X) = \sum_{j=0}^{\alpha} KM_{\eta,j} X^j$ with $KM_{\eta,j} \in \mathbb{Z}_N[X]$ |
| $f_i(X,Y)$ | $i$-th secret bivariate polynomial at the TTP |
| c | number of colluding devices. |

to this MMO Problem, but as we show later in Section 3, there are some indications that it is a hard computational problem.

**Mixing Modular Operations (MMO) Problem:** *Let $m \geq 2$ and $q_1, \ldots, q_m$ and $N$ be different positive numbers. Let $f_1, \ldots, f_m$ be polynomials of degree at most $\alpha$ with $\alpha \geq 2$ and integer coefficients. Defining $h(\eta) = \langle \sum_{i=1}^{m} \langle f_i(\eta) \rangle_{q_i} \rangle_N$, for $\eta \in \mathbb{Z}$, the MMO problem is to recover $f_1, \ldots, f_m$, given $m$, $\alpha$, $N$ and $c$ pairs $(\eta_k, h(\eta_k))$*

Hiding Information corresponds to the Noisy Polynomial Interpolation Problem of recovering an unknown polynomial $f(X) \in \mathbb{Z}_q[X]$ from approximate values of $f(\eta)$ at polynomially many points $\eta \in \mathbb{Z}_q$, see [5] and [6]. In HIMMO, these approximations are with respect to the LSBs.

**Hiding Information (HI) Problem:** *Let $KM(X) \in \mathbb{Z}_N[X]$ be a polynomial of degree at most $\alpha$, and let $K_\eta = \langle \langle KM(X = \eta) \rangle_N \rangle_{2^b}$, the last $b$ bits of $KM(X = \eta)$ modulo $N$. Reconstruct $KM(X)$ based on $c$ different pairs $(\eta, K_\eta)$.*

In the following, we first formally present HIMMO in Section 2.1; then we explain how the above problems are applied to HIMMO in Section 2.2.

### 2.1 HIMMO Operation and Parameters

HIMMO Operation comprises three phases:

**1. System initialization**

The TTP selects four public positive integers $m \geq 2, \alpha \geq 2$, $b$ and $N$, an odd positive integer of $(\alpha + 2)b$ bits. After this choice, the TTP generates the following private material: $m$ distinct positive integers $q_1, \ldots, q_m$ of the form $q_i = N - 2^b \beta_i$ where $1 \leq \beta_i \leq 2^b - 1$; $m$ symmetric bivariate 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 $f_i$ is in $\mathbb{Z}_{q_i}[X,Y]$. For $1 \leq i \leq m$, we write

$$f_i(X,Y) = \sum_{j=0}^{\alpha} f_{i,j}(Y) X^j \text{ with } f_{i,j}(Y) \in \mathbb{Z}_{q_i}[Y] \subset \mathbb{Z}_{q_i}[X,Y].$$

**2. Node registration**
For each node $\eta \in \{1, \ldots, 2^b - 1\}$ that wants to register, the TTP selects $\alpha + 1$ integers $\epsilon_{\eta,j}$ (from now on called noise[3]) such that $|\epsilon_{\eta,j}| < 2^{(\alpha+1-j)b-2}$ for $j = 0, \ldots, \alpha$. The TTP provides node $\eta$ with the secret keying material consisting of the coefficients $KM_{\eta,0}, KM_{\eta,1}, \ldots, KM_{\eta,\alpha}$, defined as:

$$KM_{\eta,j} = \left\langle \sum_{i=1}^{m} \langle f_{i,j}(\eta) \rangle_{q_i} + 2^b \epsilon_{\eta,j} \right\rangle_N . \tag{1}$$

**3. Operational phase: key agreement**
Node $\eta$ generates its key with node $\eta'$ from its keying material share as:

$$K_{\eta,\eta'} = \left\langle \langle KM_\eta(\eta') \rangle_N \right\rangle_{2^b} = \left\langle \left\langle \sum_{j=0}^{\alpha} KM_{\eta,j} \eta'^j \right\rangle_N \right\rangle_{2^b} \tag{2}$$

$K_{\eta,\eta'}$ and $K_{\eta',\eta}$ are approximately equal, as described in Theorem 1 (proof in Appendix A) where the term $3m$ is due to the mixing of the $m$ modulo reductions $q_i$ from the $m$ bivariate polynomials and $N$ and the term $\alpha + 1$ is due to the noise addition.

**Theorem 1** *Let $0 \leq \eta, \eta' \leq 2^b - 1$. Then we have that*

$$K_{\eta,\eta'} \in \{\langle K_{\eta',\eta} + jN \rangle_{2^b} \mid -\Delta \leq j \leq \Delta\},$$

*where $\Delta = 3m + \alpha + 1$.*

Key agreement between devices $\eta$ and $\eta'$ requires removing this small error. To this end, device $\eta$ sends the value $\mathcal{H}(K_{\eta,\eta'})$ to device $\eta'$, where the function $\mathcal{H}$ is such that $\mathcal{H}(i) \neq \mathcal{H}(K_{\eta,\eta'})$ 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 the communication link. An example of such a function $\mathcal{H}$ is a hash function like in [3].

### 2.2 Design Rationale

HIMMO applies the MMO problem by using secret modules $q_i$ 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. The MMO problem protects then HIMMO's root keying material and keying material shares (Equation (1)) from being recovered by an attacker from many colluding keying material shares. However, the large public modulus $N$ and the secret moduli $q_1, \ldots q_m$ share a structure that allows for the generation of a much smaller $b$-bit key by means of Equation (2). This is the HI problem that protects the secret keying materials of the devices from being recovered by an attacker from many colluding pairwise keys. Thus, the resulting $b$-bit key combines the contributions from all polynomials over different rings (MMO problem) and is much smaller than $N$ (HI problem):

$$K_{\eta,\eta'} \approx \left\langle \sum_{i=1}^{m} \langle f_i(\eta, \eta') \rangle_{q_i} \right\rangle_{2^b} = \left\langle \sum_{i=1}^{m} \langle f_i(\eta', \eta) \rangle_{q_i} \right\rangle_{2^b} \approx K_{\eta',\eta}.$$

[3] The noise captures the removal of some bits of secret keying material coefficients, see Section 4 for more details.

The above approximation is valid because in the HIMMO algorithm the first reduction during the keying material share generation is modulo $\{q_1, \ldots, q_m\}$ but the second one is modulo $N$ (Equation (2)). If we had a second reduction modulo $\{q_1, \ldots, q_m\}$, the $\beta_i$ terms in $q_i$ might lead to a bit of carry that would impact in least significat bit. This is the logic behind the term $3m$ in Theorem 1. Next, we state the general MMO and HI problems for the specific parameters of HIMMO.

**The MMO Problem in HIMMO** protects the root keying material in Equation (1) from being recovered from many compromised keying material shares. In order to explain the effects of mixing, we consider a simple special case, *viz.* that for $1 \leq i \leq m$, we have that $f_i(x, y) = A_i x^\alpha y^\alpha$ for some $A_i \in \{1, \ldots, q_i - 1\}$. Moreover, we take $N = 2^{b(\alpha+2)} - 1$ and $\epsilon_{\eta,\alpha} = 0$. We write: $A_i \eta^\alpha = R_{i,\eta}^{(2)} 2^{b(\alpha+2)} + R_{i,\eta}^{(1)} 2^b + R_{i,\eta}^{(0)}$ with $0 \leq R_{i,\eta}^{(0)} \leq 2^b - 1$, $0 \leq R_{i,\eta}^{(1)} \leq 2^{b(\alpha+1)} - 1$, and $0 \leq R_{i,\eta}^{(2)} \leq 2^{\alpha b} - 1$. As $q_i = 2^{b(\alpha+2)} - \beta_i 2^b - 1$, the single non-zero coefficient $KM_{\eta,\alpha} = \left\langle \sum_{i=1}^m \langle f_{i,\alpha}(\eta) \rangle_{q_i} \right\rangle_N$ of node $\eta$ is given by:

$$
\begin{aligned}
KM_{\eta,\alpha} &= \left\langle \sum_{i=1}^{m} \langle A_i \eta^\alpha \rangle_{q_i} \right\rangle_N = \left\langle \sum_{i=1}^{m} \left\langle \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} \right) 2^b + \left( R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \right) \right\rangle_{q_i} \right\rangle_N \\
&= \left\langle \sum_{i=1}^{m} \left\langle \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} + \left\lfloor \frac{R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)}}{2^b} \right\rfloor \right) 2^b + \langle R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \rangle_{2^b} \right\rangle_{q_i} \right\rangle_N \\
&\approx^4 \left\langle \sum_{i=1}^{m} \left( R_{i,\eta}^{(1)} + \beta_i R_{i,\eta}^{(2)} + \left\lfloor \frac{R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)}}{2^b} \right\rfloor \right) 2^b + \langle R_{i,\eta}^{(0)} + R_{i,\eta}^{(2)} \rangle_{2^b} \right\rangle_N \qquad (3)
\end{aligned}
$$

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 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 $A_i \eta^i$. 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. This makes (at least) difficult the recovery of the root keying material.

**The HI Problem** appears in Equation 2 as only the $b$ least significant bits of the polynomial evaluation are used as the key. This principle prevents an attacker from recovering the secret keying material function $KM_\eta(x)$ of a device $\eta$ even if the attacker knows many outputs (pairwise keys) of a device's keying material. As we describe in Section 3.2, an attacker might derive many $b$-bits keys from compromised keying materials, and from them, he could try to recover a polynomial of degree $\alpha$ with coefficients of size $(\alpha+2)b$ bits according to HIMMO's specification. The HIMMO design makes this difficult because if we increase $\alpha$ we do not only increase the polynomial degree but also the ratio $(\alpha + 2)$ between the number of bits that an attacker can see and the size of the coefficients, namely $b(\alpha + 2)$, to recover.

[4] The effect of the reduction module $q_i$ due to carry propagation is limited due to the form of $q_i$.

## 3 Security Analysis

In the following, we first analyze the minimum number of colluding devices required to attack the root keying material and a keying material share in the next section. Then, we analyze the feasibility of lattice-based cryptanalysis for attacking a keying material share (Section 3.2) and the root keying material (Section 3.3).

### 3.1 Bounds on the number of colluding nodes tolerated by HIMMO

**Tolerated number of colluding nodes - root keying material:** In HIMMO, reconstructing the root keying material from the keying material of a number of colluding nodes amounts to solving $\alpha + 1$ instances of the MMO problem from Section 2.

The aim of the MMO problem is to reconstruct polynomials $f_i(x) \in \mathbb{Z}_{q_i}[x]$ of degree at most $\alpha$, based on the observation of $h(x) = \left\langle \sum_{i=1}^{m} \langle f_i(x) \rangle_{q_i} \right\rangle_N$ for $c$ different values of $x$. We can offer the following heuristic argument why $c$ must be at least $m(\alpha + 1)$ to obtain a unique reconstruction.

If $c = 0$, we have no information, so that the number of possible reconstructions is at most $\prod_{i=1}^{m} q_i^{\alpha+1}$. It is less than that, because of two effects. Firstly, if $q_i$ is not a prime number, the number of different functions $\langle f_i(x) \rangle_{q_i}$ is less than $q_i^{\alpha+1}$ because of degeneracies, e.g., $\langle 4x \rangle_8 = \langle 4x^3 \rangle_8$. Secondly, degeneracies occur in the sum over $i$, because there exist polynomials $f_i(x)$ that vary so little over the domain $\{1, 2, \ldots, 2^b - 1\}$, that the modulo-$q_i$ operation does not affect them. If such functions are added in the sum over $i$, only their sum matters, not the individual functions. These effects are hard to quantify, but we do not expect them to fundamentally change our result. If we neglect these effects, and further assume that the modulo operations uniformize $h(x)$ over the $\{0, 1, \ldots, N - 1\}$ and make its values for different $x$ independent, it follows that each observation reduces the number of reconstructions by a factor $N$. Hence the expected number of reconstructions after $c$ observations is $\prod_{i=1}^{m} q_i^{\alpha+1}/N^c$. This becomes of order unity for $c \sim (\alpha + 1) \sum_{i=1}^{m} \log(q_i)/\log(N) \approx m(\alpha + 1)$ where the latter approximation holds because the $q_i$s and $N$ all have the same order of magnitude in HIMMO. This is also the range where the independence assumption must break down, since the number of reconstructions must be at least 1.

**Required number of colluding nodes to attack a node's keying material:** Similarly, for retrieving the keying material share of an uncompromised node $\eta$, we refer to the HI Problem. If the $\langle KM_\eta(\eta') \rangle_N$ were known, at least $\alpha + 1$ nodes $\eta'$ need to collude to recover $KM_\eta(Z)$. Since only $b$ of the $\lceil \log_2(N) \rceil = (\alpha + 2)b$ bits of $\langle KM_\eta(x) \rangle_N$ are exposed and taking into account that the identifiers are $b$ bits long, it is expected that at least $c = (\alpha + 1)(\alpha + 2)/2$ keys (colluding nodes) are needed to recover $KM_\eta(X)$. Section 3.2 discusses this further.

### 3.2 Attacking a keying material share

This section considers an attack that aims to obtain the secret keying material function $KM_\eta(X)$ of a node $\eta$. To this end, the attacker compromises $c$ nodes with identifiers $\eta_1, \ldots, \eta_k, \ldots \eta_c$ together with their shares $KM_{\eta_k}$. For each $\eta_k$, the attacker can readily obtain $KM_{\eta_k}(\eta)$. We assume that the attacker can also obtain $\langle \langle KM_\eta(\eta_k) \rangle_N \rangle_{2^b}$. We thus are in the situation of Problem HI.

To analyze the severity of this attack we use Theorem 8 from [5] and Theorem 5 from [6] based on lattice techniques. The algorithm from these references has five main requirements to work: First, $N$ is a prime number (although this can be avoided using heuristic arguments); Second, the identifiers are uniformly distributed in the set $\{0, \ldots, N-1\}$; Third, the number of colluding nodes $c$ is greater than $2\alpha\sqrt{b}$; Fourth, the quality of the approximation should be good enough; this is true if $\alpha$ is $O(\sqrt{b})$, and fifth, we have to assume that $\langle KM_{\eta}(\eta_k)\rangle_{2^b} = \langle KM_{\eta_k}(\eta)\rangle_{2^b}$.

Computer experiments confirm that the attack does indeed work if $c$ is large enough and $\alpha < \sqrt{b}$, even if the identifiers are only uniformly distributed in the much smaller range $\{1, \ldots, 2^b - 1\}$. However, if these conditions are not fulfilled, the lattice reduction does not provide a correct answer. This is a positive result since it allows us to derive the expected order of magnitude for HIMMO parameters. It is an open question if the fact that the HIMMO identifiers are small affects the requirements on the HIMMO parameters or can even allow for a better attack. A more general open question about the noisy polynomial reconstruction is posed in paper [7].

Independently of the above results that already provide us secure parameters, this approach cannot be applied always because the common key generated between two nodes can differ a value $\Delta$ specified in Theorem 1, i.e., the fifth condition in the above list does not hold. This further decrease the performance of the above algorithm.

### 3.3 Attacking the root keying material

In this attack, the attacker compromises $c$ nodes with identifiers $\eta_1, \ldots, \eta_k, \ldots, \eta_c$ and obtains the integers $KM_{\eta_k,j}$ for $k = 1, \ldots, c$ and $j = 0, 1, \ldots, \alpha$. The attacker aims to recover the root keying material, i.e., the polynomials $f_i(X, Y)$ for $i = 1, \ldots, m$ used in (1). This is the MMO Problem. We have studied this problem by means of lattice techniques. The main result is that since the $q_i$s are secret, it seems infeasible to construct any lattice to solve this problem because the $q_i$s are needed to construct the lattice[5].

Under the assumption that the $q_i$s are public, Appendix B describes a lattice construction in which we show that recovering the $m(\alpha+1)$ polynomial coefficients of the polynomials $f_i(X)$ from the value of $h(X) = \left\langle \sum_{i=1}^{m} \langle f_i(X)\rangle_{q_i} \right\rangle_N$ in $c$ points corrsponding to public identifiers $\{\eta_1, \ldots, \eta_k, \ldots, \eta_c\}$ is equivalent to finding vectors in a lattice $\mathcal{L}$ that are, in the $\ell_\infty$-norm, close to a target vector. This technique can be applied $\alpha+1$ times to recover all coefficients of the bivariate polynomials $f_i(X, Y)$ for $i = 1, \ldots, m$ comprising the root keying material.

This lattice $\mathcal{L}$ has dimension $m(c+\alpha+1)$ and is a subset of $\mathbb{R}^{(m+1)c+m(\alpha+1)}$. The latter dimension is the sum of the number of polynomial coefficients and the number of modulo $q_i$ and modulo $N$ operations in the resulting equations. The dimension of the lattice itself is $c$ less than that, because there are $c$ equations.

We know from Section 3.1, that we need around $c = m(\alpha+1)$ compromised devices in order to get the unique solution. If we construct the lattice with $c < m(\alpha+1)$, then we can find many solutions that fit the equations in the compromised points, but that do not fit other non-compromised devices.

There are aspects to consider: Firstly, the lattice cannot be constructed if the $q_i$s are secret. Secondly, for $c = m(\alpha+1)$, the lattice dimension is $m(m+1)(\alpha+1)$ so by

[5] Note that even for $m = 1$ the secrecy of the $q_i$s can make an attack infeasible. The work in [8] shows that a polynomial of degree $\alpha = 1$ can be recovered even if the module is secret. Results for degree $\alpha > 1$ are unknown.

taking $m$ relatively large, the lattice becomes too big for practical applications. Thirdly, the $\ell_\infty$-norm is required to ensure that all components in the minimization problem are small; the usage of the Euclidean $\ell_2$-norm does not provide satisfactory solutions since it can lead to a wrong solution in which all components are small, except for one that is too big.

## 4 Implementation

HIMMO's design is done keeping in mind that we want to achieve very good performance on small devices. The mixing of modular operations is performed at the TTP so that it does not incur additional overhead on a device. Key generation consists of the evaluation a polynomial module $N$ and taking the $b$ LSBs so that an attacker has to deal with the HI Problem. A good choice for $N$ is $2^{b(\alpha+2)}-1$ because this simplifies the implementation of modular reductions; still the differnt $q_i$ ensure the mixing of modular operations at the TTP. In the following, we first present the optimized key generation algorithm implemented on ATmega128L. Later, we describe the performance results.

### 4.1 Optimized HIMMO on ATmega128L

We have implemented HIMMO in the low-power 8-bit ATmega128L microcontroller based on the AVR RISC architecture [9]. The ATmega128L has 32 8-bit general-purpose registers (R0 to R31) of which six of them can be used in pairs as indirect address registers (X=R27:R26, Y=R29:R28 and Z=R31:R30) [10]. The program code and the initialized data are stored in a 128kB flash memory and the uninitialized data resides in the 4kB internal SRAM. Possible clock frequency values are in the range 0-8MHz. In addition, the ATmega128L has an on-chip 2-cycle multiplier which stores the 16-bit result in the registers R0 (lower 8-bit word) and R1 (higher 8-bit word).

In Algorithm 1 we show the key generation algorithm which we have implemented in the ATmega128L. The underlying method is the Horner's Rule [11]. To compute each intermediate value

$$\langle R_j \rangle_N = \langle R_{j+1} \times \eta' + KM_j \rangle_N \text{ with } j = \alpha - 1, \ldots, 0. \tag{4}$$

without performing the $\bmod N$ reduction we take advantage of $N$'s specific form, $N = 2^{(\alpha+2)b}-1$ and the small size of $\eta$. Thus, $\langle R_j \rangle_N = \langle R_{j,1} \times 2^{(\alpha+2)b} + R_{j,0} \rangle_N \approx R_{j,1} + R_{j,0}$ where $R_{j,1}$ and $R_{j,0}$ are $b$ and $(\alpha+2)b$ bits long, respectively. Note that the approximation comes from the fact that if both $R_{j,1}$ and $R_{j,0}$ are very big, then there might be a bit of carry requiring a second reduction. This carry would lead to a difference of one. The probability of this happening is obviously very low and can be easily solved during key agreement phase (See Section 2.1). In the $j^{th}$ loop iteration of the algorithm, the $b$ LSB and the $(j+2)b$ MSB of $R_j$ are handled in a different way. The $(j+2)b$ MSB of $R_j$ – in the algorithm *temp* – are multiplied by $\eta$ and added to the $(j+1)b$ MSB of the next coefficient $KM_{\eta,j-1}$ (Line 5). The $b$ MSB of $R_j$ – in the algorithm *key* – are multiplied by $\eta$ and added to the $b$ LSB of the next coefficient $KM_{\eta,j-1}$ (Line 6). The modular reduction happens when the value of *key* is updated with the contribution of the MSB stored in *temp* after being shifted $(j+2)b$ bits and added to *key* (Line 7). Next, we show that we can use the noise $\epsilon_{\eta,j}$ in Equation 2 to reduce the storage requirements. Indeed, let $c_0, c_1, \ldots, c_{\alpha(b+2)-1} \in \{0,1\}$ be such that

$$\langle \sum_{i=1}^{m} \langle f_{i,j}(\eta) \rangle_{q_i} \rangle_N = \sum_{i=0}^{(\alpha+2)b-1} c_i 2^i.$$

We choose $\epsilon_{\eta,j} = -\sum_{i=b}^{b(\alpha-j)-1} c_i 2^{i-b}$. Then clearly $|\epsilon_j| < 2^{b(\alpha-j-1)}$, so we can apply Theorem 1 to bound the difference between generated key pairs. Moreover, for this specific choice of $\epsilon_{\eta,j}$, the binary representation of $KM_{\eta,j}$ has zeros in bit positions $b, b+1, \ldots, b(\alpha-j)-1$. We need not store these zero bits, and use only the $b$ LSBs and the $(j+2)b$ MSBs of $KM_{\eta,j}$ in the computations.

**Algorithm 1** Optimized key generation

**INPUT:** $b$, $\alpha$, $\eta'$, $KM_{\eta,j}$ **where** $j \in \{0, \ldots, \alpha\}$
**OUTPUT:** $\langle\langle \sum_{j=0}^{\alpha} KM_{\eta,j}\eta'^{j}\rangle_N\rangle_{2^b}$
$key \Leftarrow \langle KM_{\eta,\alpha}\rangle_{2^b}$
$temp \Leftarrow KM_{\eta,\alpha}$
**for** $j = \alpha - 1$ **to** $0$ **do**
    $temp \Leftarrow \langle \frac{temp}{2^b}\rangle_{2^{(j+2)b}}$
    $temp \Leftarrow temp \times \eta' + \frac{KM_{\eta,j}}{(\alpha-j)b}$
    $key \Leftarrow \langle key \times \eta'\rangle_{2^b} + \langle KM_{\eta,j}\rangle_{2^b}$
    $key \Leftarrow \langle key + \frac{temp}{(j+2)b}\rangle_{2^b}$
**end for**
**return** $key$

Some additional implementation details are: the integer multiplications are performed with the operand-scanning method; and the implementation is generic working for several $\alpha$ and $b$ configurations.

As described in Equation 2, HIMMO outputs a $b$ bits long pairwise key between two devices $\eta$ and $\eta'$. CPU and memory needs can be further reduced if instead of taking a large value for $b$, we compute a long key by concatenating several $n$ $b$-bit keys, e.g. a 128 bits long key from 4 32-keys. This technique is applied in [12] to generate a 128-bit key by concatenating 8 keys generated from polynomials in $F_{2^{16}+1}$. Each of the $n$ $b$-bit keys is generated from a different keying material so, in the system initialization step, the TTP has to generate $n$ sets of $m$ symmetric bi-variate polynomials of degree $\alpha$.

With these optimizations and for any $n \cdot b$ bits key, $b$, $\alpha$, and CPU word size, our algorithm runs in time $\mathcal{O}\left(n\left(\frac{\alpha b}{WordSize}\right)^2\right)$. Memory needs for the storage of keying material are $n\frac{b}{8}\frac{\alpha^2+7\alpha+4}{2}$ bytes and RAM needs are $n\frac{(2\alpha+7+n)}{8}$ bytes.

### 4.2 Performance results

The algorithm has been implemented in assembly to allow fine-grained resource allocation and to avoid inefficiencies which the compiler could introduce. We consider three different metrics: flash use (in bytes), RAM use (in bytes) and cycle count. In Table 2 we detail the results for $n = 1$ and different $b$ and $\alpha$.

The timings have been measured with the Atmel Studio 6 and the sizes are based in the information reported by the toolchain employed to generate the binary files, in this case the avr assembler 2.

## 5 Discussion

**Secure configuration parameters:** Based on the analysis in Section 3, we argue why the HI and MMO Problems prevent an attack from launching the two main types of

Table 2: Performance evaluation of our implementation

| **Parameters** | $\alpha$ | 6 | 8 | | 10 | | 15 | | | 20 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $b$ | 32 | 32 | 64 | 32 | 64 | 32 | 64 | 128 | 32 | 64 | 128 |
| **Flash [B]** | Code | 428 | 428 | 428 | 428 | 428 | 428 | 428 | 428 | 428 | 428 | 428 |
| | Keying material | 164 | 248 | 496 | 348 | 696 | 668 | 1336 | 2672 | 1088 | 2176 | 4352 |
| | Total | 592 | 676 | 924 | 776 | 1124 | 1096 | 1764 | 3100 | 1516 | 2604 | 4780 |
| **SRAM [B]** | Data | 80 | 96 | 192 | 112 | 224 | 152 | 304 | 608 | 192 | 384 | 768 |
| **Performance** | Cycles | 12584 | 19468 | 63756 | 27696 | 91536 | 54146 | 181706 | 660026 | 88996 | 301476 | 1100836 |
| | $m$s@8MHz | 1.57 | 2.43 | 7.97 | 3.46 | 11.44 | 6.77 | 22.71 | 82.50 | 11.13 | 37.69 | 137.61 |

attacks introduced Section 2 ensuring the full collusion resistance property. As for the MMO Problem, Section 3.1 shows that an attacker needs at least $c = m(\alpha+1)$ colluding devices to attempt to attack the root keying material. This is a good result since we can configure HIMMO with an almost arbitrary high value of $m$ without increasing the resource needs for key establishment. Furthermore, Section 3.3 discusses that a lattice cannot be constructed because the $q_i$s are unknown. Under the assumption that the $q_i$s are known, we can relate the recovery of the root keying material to finding a close vector in a lattice whose dimension grows with $m^2$. With regard to the HI Problem, Subsection 3.2 discusses that if $\alpha > \sqrt{b}$, it seems to be difficult to attack HIMMO even if an attacker captures $c > 2\alpha\sqrt{b}$ nodes. Since the performance of our algorithm decreases with $\alpha^2$, it is better to compute a $nb$-bit key as the concatenation of $n$ b-bit keys generated from $n$ HIMMO instances. From our security analysis, the most efficient way of configuring HIMMO is with $n = 4$ HIMMO instances of $b = 32$-bits and $\alpha = 6$. With these settings $O(2^{32})$ nodes can be registered and the size of the generated keys is 128 bits. This leads to a key computation time of 7 ms with an algorithm and keying material size of 428 B and 656 B, respectively.

**Related work:** Table 3 compares our *generic* HIMMO implementation with a polynomial scheme [2,12], the previous broken attempt of creating an scheme with the full collusion resistance property [3] and pairings [13]. The results are taken from those works when implemented and evaluated in the AVR ATmega128L MCU as in our experiments. HIMMO's performance is as the one of a polynomial scheme of degree 40 with the advantage that HIMMO aims at full collusion resistance based on the HI and MMO design concepts. Although Zhang's scheme is broken, we observe that HIMMO improves the performance of their design. We further compare HIMMO with pairings [13] because HIMMO is also ID-based showing some related operational features. We note that HIMMO is by far more efficient here.

Table 3: Performance comparison with related work

| | HIMMO | Zhang *et al.* [3] | Liu *et al.* [12]$t = 40$ | Oliveira *et al.* [13] |
|---|---|---|---|---|
| CPU Cycles | $2.5 \times 10^4$ | $9.6 \times 10^5$ | $4.0 \times 10^4$ | $1.4 \times 10^7$ |
| Flash (bytes) | 428 + 552 | 15005 | 416 + 656 | 38800 |
| RAM (bytes) | 88 | 325 | 20 | 512 |

## 6 Conclusions

This paper has presented HIMMO, a scheme allowing for ID-based pairwise key establishment, and its underlying design principles. We have argued that these design principles, the HI and MMO Problems, can allow for good collusion resistance properties and very efficient implementations. As such, HIMMO can find application in networks of resource-constrained devices, in systems in which timing is crucial, and the combination of ID-based and full-collusion resistance properties can also bring operational advantages in traditional computer networks.

Further research is required to analyze the design principles, in particular the MMO Problem, on which HIMMO relies. We have given indications of the hardness of this problem, and how it can be applied to create a cryptosystem such as HIMMO. We believe that many other cryptographic blocks/functionalities – such as the provided example about secure broadcast – can be designed/enabled based on the MMO Problem also with very good performance features.

## References


1. T. Matsumoto and H. Imai, "On the key predistribution system: A practical solution to the key distribution problem," in *A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, ser. CRYPTO '87. London, UK, UK: Springer-Verlag, 1988, pp. 185–193. [Online]. Available: http://dl.acm.org/citation.cfm?id=646752.704749
2. C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '92. London, UK, UK: Springer-Verlag, 1993, pp. 471–486. [Online]. Available: http://dl.acm.org/citation.cfm?id=646757.705531
3. W. Zhang, M. Tran, S. Zhu, and G. Cao, "A random perturbation-based scheme for pairwise key establishment in sensor networks," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '07. New York, NY, USA: ACM, 2007, pp. 90–99. [Online]. Available: http://doi.acm.org/10.1145/1288107.1288120
4. M. Albrecht, C. Gentry, S. Halevi, and J. Katz, "Attacking cryptographic schemes based on "perturbation polynomials"," in *Proceedings of the 16th ACM conference on Computer and communications security*, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/1653662.1653664
5. I. E. Shparlinski, "Sparse polynomial approximation in finite fields," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, ser. STOC '01. New York, NY, USA: ACM, 2001, pp. 209–215. [Online]. Available: http://doi.acm.org/10.1145/380752.380803
6. I. Shparlinski and A. Winterhof, "Noisy interpolation of sparse polynomials in finite fields," *Appl. Algebra Eng., Commun. Comput.*, vol. 16, no. 5, pp. 307–317, Nov. 2005. [Online]. Available: http://dx.doi.org/10.1007/s00200-005-0180-1
7. S. R. Blackburn, D. Gomez-Perez, J. Gutierrez, and I. E. Shparlinski, "Reconstructing noisy polynomial evaluation in residue rings," *J. Algorithms*, vol. 61, no. 2, pp. 47–59, Nov. 2006. [Online]. Available: http://dx.doi.org/10.1016/j.jalgor.2004.07.002
8. J. Stern, "Secret linear congruential generators are not cryptographically secure," in *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, ser. SFCS '87. Washington, DC, USA: IEEE Computer Society, 1987, pp. 421–426. [Online]. Available: http://dx.doi.org/10.1109/SFCS.1987.51
9. Atmel, "8-bit AVR Microcontroller with 128K Bytes In-System Programmable Flash," http://www.atmel.com/Images/doc2467.pdf, June 2011. [Online]. Available: http://www.atmel.com/Images/doc2467.pdf

10. ——, "8-bit AVR Instruction Set," http://www.atmel.com/images/doc0856.pdf, July 2010. [Online]. Available: http://www.atmel.com/images/doc0856.pdf
11. D. E. Knuth, *The art of computer programming, volume 2 (3rd ed.): seminumerical algorithms*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.
12. D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 1, pp. 41–77, Feb. 2005. [Online]. Available: http://doi.acm.org/10.1145/1053283.1053287
13. L. B. Oliveira, D. F. Aranha, C. P. L. Gouvêa, M. Scott, D. F. Címara, J. López, and R. Dahab, "Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," *Comput. Commun.*, vol. 34, no. 3, pp. 485–493, Mar. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.05.013

## Appendix A: proof of Theorem 1

In this appendix, we provide a proof of Theorem 1. In the proof, we rely on the fact that for each integer $a$ and each positive integer $N$ we have that

$$a = \lfloor \frac{a}{N} \rfloor N + \langle a \rangle_N. \tag{5}$$

**Lemma 1.** Let $a, b, N$ be integers, with $N$ positive, then we have

$$\langle a+b \rangle_N = \langle a \rangle_N + \langle b \rangle_N - \lambda N$$

for some integer $\lambda \in \{0, 1\}$.

*Proof.* Using (5), we have that $\langle a+b \rangle_N = \langle \langle a \rangle_N + \langle b \rangle_N \rangle_N = \langle a \rangle_N + \langle b \rangle_N - \lambda N$ where $\lambda = \left\lfloor \frac{\langle a \rangle_N + \langle b \rangle_N}{N} \right\rfloor$. As $0 \leq \lambda < 2$ and $\lambda$ is integer, the lemma follows. □

**Lemma 2.** Let $\alpha, b, p, N$ and $\beta$ be positive integers such that $p = N - \beta 2^b$, $\beta < 2^b$ and $2^{(\alpha+2)b-1} < N \leq 2^{(\alpha+2)b}$. Let $h(X) = \sum_{i=0}^{\alpha} h_i X^i$ be a polynomial with integer coefficients such that $0 \leq h_i \leq p-1$. If $\eta \in \{1, \ldots, 2^b - 1\}$, then

$$\langle h(\eta) \rangle_p = \langle h(\eta) \rangle_N + \mu 2^b - \lambda N$$

for some integers $\mu$, $\lambda$ and with $\lambda \in \{0, 1, 2\}$.

*Proof.* It is clear that $h(\eta) = \lambda_1 p + \langle h(\eta) \rangle_p = \lambda_2 N + \langle h(\eta) \rangle_N$ with $\lambda_1 = \left\lfloor \frac{h(\eta)}{p} \right\rfloor$ and $\lambda_2 = \left\lfloor \frac{h(\eta)}{N} \right\rfloor$. Hence,

$$\begin{aligned} \langle h(\eta) \rangle_p &= \langle h(\eta) \rangle_N + \lambda_2 N - \lambda_1 p = \langle h(\eta) \rangle_N + \lambda_1 (N-p) + (\lambda_2 - \lambda_1) N \\ &= \langle h(\eta) \rangle_N + 2^b \beta \lambda_1 + (\lambda_2 - \lambda_1) N = \langle h(\eta) \rangle_N + \mu 2^b - \lambda N. \end{aligned}$$

where $\mu = \beta \lambda_1$ and $\lambda = \lambda_1 - \lambda_2$. Since $N > p$, $\lambda \geq 0$.
Moreover,we have that

$$\lambda = \lambda_1 - \lambda_2 \leq \frac{h(\eta)}{p} - \lambda_2 < \frac{h(\eta)}{p} - \frac{h(\eta)}{N} + 1$$

and thus

$$\lambda < h(\eta) \frac{2^b \beta}{Np} + 1.$$

We clearly have that

$$h(\eta) = \sum_{i=0}^{\alpha} h_i \eta^i < \sum_{i=0}^{\alpha} p 2^{ib} < p \frac{2^{(\alpha+1)b}}{2^b - 1}, \text{ so } \lambda < \frac{2^{(\alpha+2)b}\beta}{N(2^b-1)} + 1.$$

The bounds on $\beta$ and $N$ imply that $\lambda < 3$ and so, as $\lambda$ is an integer, $\lambda \leq 2$. $\square$

*Remark 1. Note that if $\beta_i < \frac{N}{2^{b(\alpha+2)}}(2^b - 1)$, then in Lemma 2 we have that $\lambda \in \{0, 1\}$.*

We are now in a position to prove Theorem 1. By definition,

$$K_{\eta,\eta'} = \langle \langle \sum_{j=0}^{\alpha} KM_{\eta,j} (\eta')^j \rangle_N \rangle_{2^b} \text{ with } KM_{\eta,j} = \sum_{i=1}^{m} \langle f_{i,j}(\eta) \rangle_{q_i} \ + 2^b \epsilon_{\eta,j}.$$

We write

$$\phi_{i,\eta}(x) = \sum_{j=0}^{\alpha} \langle f_{i,j}(\eta) \rangle_{q_i} x^j \text{ and } F_\eta(x) = \sum_{i=1}^{m} \phi_{i,\eta}(x) + 2^b \sum_{j=0}^{\alpha} \epsilon_{\eta,j} x^j.$$

Then

$$K_{\eta,\eta'} = \langle \langle F_\eta(\eta') \rangle_N \rangle_{2^b}.$$

We introduce the following notation for convenience,

$$\phi_{i,\eta}(X) = \langle f_i(X, \eta) \rangle_{q_i} = \sum_{j=0}^{\alpha} \langle g_{i,j}(\eta) \rangle_{q_i} X^j,$$

where $f_i(X, Y) = \sum_{j=0}^{\alpha} g_{i,j}(Y) X^j$, where $g_{i,j}(Y) \in \mathbb{F}_{q_i}[Y]$. From this, it is clear that

$$F_\eta(X) = \sum_{i=1}^{m} \phi_{i,\eta}(X) + 2^b \sum_{j=0}^{\alpha} \epsilon_j X^j.$$

*Proof (*Proof of Theorem 1). First, we compute the following

$$\langle F_\eta(\eta') \rangle_N = \left\langle \sum_{i=1}^{m} \phi_{i,\eta}(\eta') + 2^b \sum_{j=0}^{\alpha} \epsilon_j (\eta')^j \right\rangle_N =$$

$$= \sum_{i=1}^{m} \langle \phi_{i,\eta}(\eta') \rangle_N + \langle \sum_{j=0}^{\alpha} 2^b \epsilon_j (\eta')^j \rangle_N + \lambda_1 N$$

for some integer $\lambda$ with $-m \leq \lambda_1 \leq 0$. Using the upper bound on the $\epsilon$'s we find

$$\langle F_\eta(\eta') \rangle_N = \sum_{i=1}^{m} \langle \phi_{i,\eta}(\eta') \rangle_N + 2^b \sum_{j=0}^{\alpha} \epsilon_j (\eta')^j + \lambda_2 N$$

$$\text{with } -m - \frac{1}{2}(\alpha + 1) \leq \lambda_2 \leq \frac{1}{2}(\alpha + 1).$$

By applying Lemma 2 we obtain that

$$\sum_{i=1}^{m} \langle \phi_{i,\eta}(\eta') \rangle_N = \sum_{i=1}^{m} \langle \phi_{i,\eta}(\eta') \rangle_{q_i} + \lambda_3 N + \mu 2^b,$$

for some integers $\mu$ and $\lambda_3$ with $-2m \leq \lambda_3 \leq 0$. We thus have that

$$K_{\eta,\eta'} = \left\langle \left\langle F_\eta(\eta') \right\rangle_N \right\rangle_{2^b} = \left\langle \sum_{i=1}^{m} \left\langle f_i(\eta', \eta) \right\rangle_{q_i} + \lambda N \right\rangle_{2^b}.$$

for $\lambda = \lambda_2 + \lambda_3$ and so $-\frac{1}{2}(\alpha+1) - 3m \leq \lambda \leq \frac{1}{2}(\alpha+1)$. Similarly,

$$K_{\eta',\eta} = \left\langle \left\langle F_{\eta'}(\eta) \right\rangle_N \right\rangle_{2^b} = \left\langle \sum_{i=1}^{m} \left\langle f_i(\eta, \eta') \right\rangle_{q_i} + \lambda' N \right\rangle_{2^b},$$

for some integer $\lambda'$, with $-\frac{1}{2}(\alpha+1) - 3m \leq \lambda' \leq \frac{1}{2}(\alpha+1)$.
As the $f_i$'s are symmetric, we have proven Theorem 1. $\square$

## Appendix B: equivalence of the MMO problem and a 'close vector problem' in a lattice

In this appendix we show that the MMO problem defined in Section 2 is equivalent to a close vector problem the moduli $q_i$, $1 \leq i \leq m$, are known.

From the identifiers $\eta_1, \ldots, \eta_c$ of the compromised devices we build the Vandermonde matrix $\mathbf{V}$ of size $c \times (\alpha+1)$ as

$$\mathbf{V} = \begin{pmatrix} 1 & \eta_1 & \eta_1^2 & \cdots & \eta_1^\alpha \\ 1 & \eta_2 & \eta_2^2 & \cdots & \eta_2^\alpha \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \eta_c & \eta_c^2 & \cdots & \eta_c^\alpha \end{pmatrix}.$$

The polynomial $f_i$ is defined by coefficients $r_{i,k} \in \{0, 1, \ldots, q_i - 1\}$, $0 \leq k \leq \alpha$ as $f_i(x) = \sum_{k=0}^{\alpha} r_{i,k} x^k$. We define the column vector $\mathbf{r}_i = (r_{i,0}, r_{i,1}, \ldots, r_{i,\alpha})^{\mathrm{t}}$.

The MMO problem can now be formulated as follows: given the vector $\mathbf{h}$ of which the components are the function values in the identifiers, $\mathbf{h} = (h(\eta_1), \ldots . h(\eta_c))^{\mathrm{t}}$, find integer vectors $\mathbf{r}_1, \ldots, \mathbf{r}_m$ of length $\alpha + 1$ such that

$$0 \leq \mathbf{r}_i \leq q_i - 1$$

and

$$\begin{aligned} \mathbf{h} &= \left\langle \sum_{i=1}^{m} \left\langle \mathbf{V}\mathbf{r}_i \right\rangle_{q_i} \right\rangle_N \\ &= \sum_{i=1}^{m} \left( \mathbf{V}\mathbf{r}_i - q_i \left\lfloor \frac{\mathbf{V}\mathbf{r}_i}{q_i} \right\rfloor \right) - N \left\lfloor \frac{\sum_{i=1}^{m} \left( \mathbf{V}\mathbf{r}_i - q_i \left\lfloor \frac{\mathbf{V}\mathbf{r}_i}{q_i} \right\rfloor \right)}{N} \right\rfloor \end{aligned}$$

where the inequalities and modulo and rounding operations act component-wise.

Note that for any integer $a$ and $q$ with $q > 0$, $\lfloor a/q \rfloor$ is equal to the unique integer $\lambda$ such that $|(a/q) - \lambda - (q-1)/(2q)| < 1/2$. Similarly, for an integer vector $\mathbf{a}$, $\lfloor \mathbf{a}/q \rfloor$ is equal to the unique integer vector $\boldsymbol{\lambda}$ such that for each component it holds that $|a_k/q - \lambda_k - (q-1)/(2q)| < 1/2$. The latter condition is equivalent to $\max_k |a_k/q - \lambda_k - (q-1)/(2q)| < 1/2$.

This motivates the following equivalent formulation of the MMO problem: given $\mathbf{h}$, find integer column vectors $\mathbf{r}_1, \ldots, \mathbf{r}_m$ of length $\alpha+1$ and $\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_m$ of length $c$ such that

$$\mathbf{h} = -N\boldsymbol{\lambda}_0 + \sum_{i=1}^{m} \left(-q_i\boldsymbol{\lambda}_i + \mathbf{V}\mathbf{r}_i\right), \tag{6}$$

$$\left|\left| \frac{\sum_{i=1}^{m} \mathbf{V}\mathbf{r}_i - q_i\boldsymbol{\lambda}_i}{N} - \boldsymbol{\lambda}_0 - \frac{(N-1)\mathbf{e}_c}{2N} \right|\right|_\infty < \frac{1}{2}, \tag{7}$$

and for $1 \le i \le m$

$$\left|\left| \frac{\mathbf{V}\mathbf{r}_i}{q_i} - \boldsymbol{\lambda}_i - \frac{(q_i-1)\mathbf{e}_c}{2q_i} \right|\right|_\infty < \frac{1}{2} \tag{8}$$

and

$$\left|\left| \frac{\mathbf{r}_i}{q_i} - \frac{(q_i-1)\mathbf{e}_{\alpha+1}}{2q_i} \right|\right|_\infty < \frac{1}{2}. \tag{9}$$

Here $\mathbf{e}_c$ and $\mathbf{e}_{\alpha+1}$ denote column vectors of length $c$ and $\alpha+1$, respectively, of which all components are equal to 1 and $||\cdot||_\infty$ denotes the $\ell_\infty$-norm of a vector, the maximum of the absolute values of its components. Inequalities (7) and (8) embody the constraints that the vectors $\boldsymbol{\lambda}_0, \ldots, \boldsymbol{\lambda}_m$ are equal the result of the rounding operation, while inequality (9) is equivalent to $0 \le \mathbf{r}_i \le q_i - 1$.

We concatenate the vectors $-\boldsymbol{\lambda}_i$ and $\mathbf{r}_i$ into an integer column vector $\mathbf{x}$ of length $(m+1)c + m(\alpha+1)$:

$$\mathbf{x}^{\mathrm{t}} = (-\boldsymbol{\lambda}_0^{\mathrm{t}}, -\boldsymbol{\lambda}_1^{\mathrm{t}}, \ldots, -\boldsymbol{\lambda}_m^{\mathrm{t}}, \mathbf{r}_1^{\mathrm{t}}, \ldots, \mathbf{r}_m^{\mathrm{t}})^{\mathrm{t}}$$

and define a matrix $\mathbf{A}$ of size $c \times ((m+1)c + m(\alpha+1))$ as a horizontal concatenation of $m+1$ instances of the $c \times c$ identity matrix $\mathbf{I}_c$ multiplied by $N$, resp. $q_i$ and $m$ copies of $\mathbf{V}$:

$$\mathbf{A} = \left(N\mathbf{I}_c\ q_1\mathbf{I}_c \ldots q_m\mathbf{I}_c\ \mathbf{V} \ldots \mathbf{V}\right),$$

so that equation (6) becomes

$$\mathbf{h} = \mathbf{A}\mathbf{x}. \tag{10}$$

Furthermore we define a matrix $\mathbf{B}$ of size $((m+1)c + m(\alpha+1)) \times ((m+1)c + m(\alpha+1))$ as the block matrix

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_c & \frac{q_1}{N}\mathbf{I}_c & \cdots & \frac{q_m}{N}\mathbf{I}_c & \frac{1}{N}\mathbf{V} & \cdots & \frac{1}{N}\mathbf{V} \\ 0 & \mathbf{I}_c & \cdots & 0 & \frac{1}{q_1}\mathbf{V} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{I}_c & 0 & \cdots & \frac{1}{q_m}\mathbf{V} \\ 0 & 0 & \cdots & 0 & \frac{1}{q_1}\mathbf{I}_{\alpha+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \frac{1}{q_m}\mathbf{I}_{\alpha+1} \end{pmatrix}$$

and the column vector $\mathbf{u}$ of length $(m+1)c + m(\alpha+1)$ as

$$\mathbf{u}^{\mathrm{t}} = \left( \frac{N-1}{2N}\mathbf{e}_c^{\mathrm{t}}, \frac{q_1-1}{2q_1}\mathbf{e}_c^{\mathrm{t}}, \ldots, \frac{q_m-1}{2q_m}\mathbf{e}_c^{\mathrm{t}}, \frac{q_1-1}{2q_1}\mathbf{e}_{\alpha+1}^{\mathrm{t}}, \ldots, \frac{q_m-1}{2q_m}\mathbf{e}_{\alpha+1}^{\mathrm{t}} \right)^{\mathrm{t}}.$$

Now the inequalities (7), (8) and (9) are equivalent to the single inequality

$$||\mathbf{B}\mathbf{x} - \mathbf{u}||_\infty < \frac{1}{2}. \tag{11}$$

So finding a solution to the MMO problem is equivalent to finding an integer solution of equation (10) that satisfies the constraint inequality (11).

Let $\mathbf{x}_0$ be an arbitrary integer solution to equation (10). Every integer solution $\mathbf{x}$ of equation (10) can now be written as $\mathbf{x} = \mathbf{x}_0 + \mathbf{w}$, where $\mathbf{A}\mathbf{w} = 0$. Substituting this into equation (11), we obtain

$$||\mathbf{y} - (\mathbf{u} - \mathbf{B}\mathbf{x}_0)||_\infty < \frac{1}{2} \text{ with } \mathbf{y} \in \mathcal{L},$$

where $\mathcal{L}$ is defined as

$$\mathcal{L} = \left\{ \mathbf{B}\mathbf{w} \,\middle|\, \mathbf{w} \in \mathbb{Z}^{(m+1)c+m(\alpha+1)}, \mathbf{A}\mathbf{w} = 0 \right\}.$$

$\mathcal{L}$ is a discrete abelian subgroup of $\mathbb{R}^{(m+1)c+m(\alpha+1)}$, in other words, a lattice. The dimension of the lattice is $m(c + \alpha + 1)$.

This shows that every solution $\mathbf{x}$ to the MMO problem can be mapped to a lattice vector $\mathbf{y} = \mathbf{B}(\mathbf{x} - \mathbf{x}_0)$ that has $\ell_\infty$-distance less than $1/2$ to the target vector $\mathbf{u} - \mathbf{B}\mathbf{x}_0$ Note that the matrix $\mathbf{B}$ is invertible, which implies that the correspondence between MMO solutions and close vectors is one-to-one.

All that remains is to find any solution of equation (10) that can play the role of $\mathbf{x}_0$. That is particularly easy if there is a pair $(N, q_i)$ or $(q_i, q_j)$ with $\gcd(N, q_i) = 1$ or $\gcd(q_i, q_j) = 1$. We assume that this holds, and without loss of generality, we assume that $\gcd(N, q_1) = 1$. Then there exist integers $\mu_0$ and $\mu_1$ such that $\mu_0 N + \mu_1 q_1 = 1$. It follows that the integer vector $(\mu_0 \mathbf{h}^\mathrm{t}, \mu_1 \mathbf{h}^\mathrm{t}, \mathbf{0}^\mathrm{t}_{(m-1)c+m(\alpha+1)})^\mathrm{t}$ satisfies equation (10).