Attacks and parameter choices in HIMMO

Oscar García-Morchón[1], Ronald Rietman[1], Ludo Tolhuizen[1], Jose-Luis Torre-Arce[1], Moon Sung Lee[2], Domingo Gómez-Pérez[3], Jaime Gutiérrez[3], and Berry Schoenmakers[4]

[1] Philips Research, Eindhoven, The Netherlands
[2] University of Luxembourg
[3] University of Cantabria, Santander, Spain
[4] TU Eindhoven, The Netherlands

Abstract. The HIMMO scheme has been introduced as a lightweight collusion-resistant key pre-distribution scheme, with excellent efficiency in terms of bandwidth, energy consumption and computation time. As its cryptanalysis relies on lattice techniques, HIMMO is also an interesting quantum-safe candidate. Unlike the schemes by Blom, by Matsumoto and Imai, and by Blundo *et al*, which break down once the number of colluding nodes exceeds a given threshold, it aims at tolerating any number of colluding nodes.
In 2015, a contest for the verification of the scheme was held. During the contest, a method was developed to guess a key by finding an approximate solution of one of the problems underlying the scheme. This attack involves finding a short vector in a lattice of dimension linear in a system parameter α and allowed key recovery for several challenges. Thwarting this attack by increasing α would lead to a significant performance degradation, as CPU and memory requirements for the implementation of the scheme scale quadratically in α.
This paper describes a generalization of HIMMO parameters that allows configuring the scheme such that both its performance and the dimension of the lattice involved in the attack grow linearly in α. Two attacks inspired by the one developed in the contest are described, and the impact of those attacks for different parameter choices is discussed. Parameters choices are described that thwart existing attacks while enabling high performance implementations of the scheme.

Keywords: Key predistribution scheme, collusion attack, identity, lattice analysis

1 Introduction

Background More and more devices are getting connected to the Internet forming the so called Internet of Things (IoT). Strong security is fundamental for the IoT to be deployed in critical applications like healthcare, smart cities and smart

energy. However, currently, it is estimated that 70 % of IoT devices have security vulnerabilities and devices are often poorly managed. Security methods need to be scalable and be easily deployable, require limited resources regarding bandwidth, energy or CPU time, and remain secure during the long lifetime of IoT devices. Simultaneously, quantum computers are expected to be developed in the next years and as a consequence traditional public-key primitives in use today will be broken. This also motivates research into novel security architectures for the Internet so that it can be become more secure and efficient.

In [14], [12] the HIMMO scheme is proposed as an innovative method for pairwise key establishment. It is a lightweight key pre-distribution scheme that enables secure sharing of keys and verification of credentials between any pair of devices in a single message - ideal for real-time interactions. The scheme relies on a trusted third party (TTP), or an infrastructure of TTPs, to handle the root keying material and distribute to each device secret key generating functions linked to its identifier or credentials. By re-using existing device identities like MAC addresses, integration of HIMMO in existing systems is easy and attractive. It can be easily integrated in existing protocols such as TLS so that it can be readily applied to address many real-world use cases. The fact that HIMMO cryptanalysis relies on lattice techniques makes it also an interesting quantum-safe candidate. The goal of HIMMO is to achieve excellent efficiency in terms of bandwidth, energy consumption, and computation time to generate a key while being a collusion resistant and quantum-safe key pre-distribution scheme.

During the NIST workshop on "Cyber-security in a post-quantum world" and later during the rump session at Crypto2015, a contest (`https://www.himmo-scheme.com`) was announced for facilitating the assessment and further improvements of the HIMMO scheme by the security community. The contest ended on December 31st 2015 with different results for the three challenges linked to the two mathematical problems underlying HIMMO, the HI and MMO problems, and to the scheme itself.

In this paper, we introduce parameter updates in the HIMMO scheme that deal with the approach used by one of the participants in the contest to find an approximate solution to the MMO problem, and in this way, to solve several challenges in the contest. We describe the corresponding security analysis and discuss how with these updated parameters the performance of the scheme improves compared with previous results reported in [14].

Related work We motivate our approach by discussing related work covering three alternative approaches to key establishment.

The first approach is to use any Diffie-Hellman based scheme, involving a one-way function defined for a discrete log setting (including elliptic curve cryptography). The main drawback of these schemes is simply that evaluation of the one-way function is too costly, and this drawback extends to schemes involving pairings. Some relevant examples are the schemes and related constructions in [6,8,9,10,15,21,22].

The second approach is to use a simpler one-way function. For instance, [17] proposes a key exchange protocol based on the NTRU one-way function.

Basically, two nodes establish a common key by exchanging NTRU encryptions of their private keys, which is computationally efficient. However, several serious drawbacks remain. In the first place, there is no protection against man-in-the-middle attacks as the corresponding public keys of the nodes are not certified. Secondly, the communication complexity of such a key establishment protocol is high, as it involves an exchange of public key encryptions; hence the protocol is too costly for resource-constrained devices. And, finally such a scheme lacks identity-based modes.

The third approach is to actually build a key predistribution scheme [18]. Key predistribution schemes allow for the establishment of pairwise keys in a large network of nodes, where each node uses its so-called keying material as secret input [18]. All pairwise keys are determined by a master key, and one gets all the benefits mentioned above. Clearly, the main challenge is to achieve collusion resistance while being efficient and that's what HIMMO is solving. In Blundo et al.'s scheme [5], a master key consisting of a symmetric bivariate polynomial over a finite field is generated by a trusted party, and each node's keying material consists of the univariate polynomial obtained as the evaluation of the master key at a field element associated with the node. Key establishment in Blundo et al.'s scheme is very efficient, as it amounts to the evaluation of a polynomial over a finite field. The scheme is secure against collusions whose size does not exceed the degree of the polynomial. For larger collusions, however, the security of Blundo et al.'s scheme breaks down completely. In the literature there has only been one attempt at constructing an efficient scheme with collusion-resistance against arbitrary collusions [24]. However, the collusion resistance claims turned out to be flawed [2]. The starting point for HIMMO goes back to Blundo et al.'s elegant key predistribution scheme [5] and the general definition of Matsumoto and Imai [18] and aims at defining a key predistribution scheme that achieves efficiency comparable to Blundo et al.'s scheme but with much better security, basically tolerating collusions of practically any size.

Roadmap The paper is organized as follows. In Section 2, we describe generalized HIMMO parameters and different choices for them. In Section 3, we detail and describe the analysis of the different lattice-based attacks aiming at compromising the TTP, a node, or a single key. The TTP can be attacked by solving the MMO problem which, if successful, allows the attacker to obtain all keys in the network. A node can be attacked by solving the HI problem or obtaining an approximate solution to the MMO problem. If successful, the attacker can and obtain all the keys of the node under attack. We discuss which parameters should be used. Section 4 discusses the efficient implementation, associated performance, and compares with related work. In Section 5, we draw conclusions and indicate directions for further research.

2 HIMMO

In this section we describe the generalized parameters for the HIMMO scheme. Throughout this paper we use the following notation. For all integers x and all

positive integers N, we denote with $\langle x \rangle_N$ the integer in the interval $[0, N-1]$ that is equivalent to x modulo N, i.e., that differs an integer multiple of N. Moreover, we denote with $\{x\}_N$ the integer in the interval $(-\frac{N}{2}, \frac{N}{2}]$ that is equivalent to x modulo N.

2.1 HIMMO Operation

HIMMO has several system parameters, viz. B, the bit length of node identifiers, b, the key length, N, an odd reduction integer, and integers α and m. Other system parameters are non-negative integer functions $\phi_0, \ldots, \phi_\alpha$ on $\{0, 1, \ldots, 2^B - 1\}$. We remark that α plays a key role both in the security and performance of the system.

As in any key pre-distribution scheme, HIMMO relies on a trusted third party (TTP), and several phases can be distinguished during its operation.

In the **set-up phase**, the TTP, given the system parameters, secretly and randomly generates the following root keying material:

- m distinct random moduli $q_1, q_2, \ldots, q_m$ of the form $q_i = N - 2^b\beta_i$, where $0 \leq \beta_i < 2^B$ and at least one of $\beta_1, \ldots, \beta_m$ is odd.
- for $1 \leq i \leq m$ and $0 \leq j \leq k \leq \alpha$, a random integer $R^{(i)}_{j,k}$ with $0 \leq R^{(i)}_{j,k} \leq q_i - 1$, and for $k < j \leq \alpha, R^{(i)}_{j,k} = R^{(i)}_{k,j}$

In the **keying material extraction phase**, the TTP provides node $\xi \in [0, 2^B)$ with coefficients $G_0(\xi), \ldots, G_\alpha(\xi)$, where for $0 \leq k \leq \alpha$

$$G_k(\xi) = \big\langle \sum_{i=1}^{m} \langle \sum_{j=0}^{\alpha} R^i_{j,k} \phi_j(\xi) \rangle_{q_i} \big\rangle_N. \tag{1}$$

A later phase comprises a **key establishment protocol** in which if node ξ wishes to communicate with node η, it computes the key $K_{\xi,\eta}$ defined as

$$K_{\xi,\eta} = \langle G^\xi(\eta) \rangle_{2^b}, \text{ where } G^\xi(\eta) = \langle \sum_{k=0}^{\alpha} G_k(\xi)\phi_k(\eta) \rangle_N. \tag{2}$$

2.2 Key symmetry and reconciliation

In general, for $\xi, \eta \in [0, 2^B)$, the generated keys $K_{\xi,\eta}$ and $K_{\eta,\xi}$ need not be equal. The following theorem, that relates $K_{\xi,\eta}$ and $K_{\eta,\xi}$, can be proved analogously to [12, Thm. 1].

Theorem 1 *Let $0 \leq \xi, \eta \leq 2^B - 1$. We have that*

$$K_{\eta,\xi} \in \{\langle K_{\xi,\eta} + jN \rangle_{2^b} \mid j \in \mathbb{Z}, |j| \leq \Delta\}, \text{ where}$$

$$\Delta = 1 + \lfloor \frac{2^b}{N} \cdot \max\{\sum_{k=0}^{\alpha} \phi_k(x) \mid 0 \leq x \leq 2^B - 1\} \cdot \sum_{i=1}^{m} \beta_i \rfloor.$$

In order to arrive at a common key, node ξ sends to node η some helper data, for instance, the s least significant bits of $K_{\xi,\eta}$, where $s := \lceil \log_2(2\Delta+1) \rceil$. From these s bits and $K_{\eta,\xi}$, node η can determine $K_{\xi,\eta}$, see [12] for more details. The remaining $b-s$ most significant bits of $K_{\xi,\eta}$ then serve as a common key for ξ and η.

2.3 Parameter choices

In [12],[14], $\phi_k(x) = x^k$ was used for $0 \leq k \leq \alpha$. In this case, $\sum_{k=0}^{\alpha} \phi_k(x) \leq 2^{\alpha B}$ for $0 \leq x < 2^B$, and so $\Delta \leq 1 + \frac{2^{\alpha B+b}}{N} \sum_{i=1}^{m} \beta_i \leq 1 + \frac{m 2^{(\alpha+1)B+b}}{N}$. Hence, if N has a length of $(\alpha+1)B+b$ bits, then $\Delta \leq m$. Note, however, that N (and therefore the coefficients $G_k(\xi)$) grow in α.

In the present paper, we focus on parameters that allow to take the size of N independent of α. In particular, we stipulate that for $0 \leq k \leq \alpha$ and $x \in [0, 2^B)$,

$$0 \leq \phi_k(x) \leq 2^B. \tag{3}$$

For N, we take an odd integer with a length of $(2B+b)$ bits. As (3) is satisfied, it follows that $\Delta \leq 1 + m(\alpha+1)$ in Theorem 1. The bit length of N is thus independent of α, and much smaller than with the choices for ϕ_k from [12], [14]. This leads to an improved performance-security trade-off, as explained in subsection 3.4.

With the above choice for the size of N, the scheme is operationally correct, irrespective of the precise form of the functions ϕ_k, as long as (3) is satisfied. In order that different nodes have different keying materials, we require that for any distinct $\xi, \eta \in [0, 2^B)$

$$(\phi_0(\xi), \ldots, \phi_\alpha(\xi)) \neq (\phi_0(\eta), \ldots, \phi_\alpha(\eta)).$$

In the remainder of the paper, we will use $\phi_k(x) = \langle \langle x^k \rangle_p \rangle_{2^B}$, where p is a prime slightly larger than 2^B. These functions can be easily evaluated, and fulfill the above condition. We note that other choices for $\phi_k(x)$ are feasible, e.g., the hash function of x concatenated with k. This choice would lead to more uncorrelated $\phi_k(x)$ values.

3 Security Analysis

We assume an attack model in which an adversary has obtained the key generating functions of c different nodes. The identifiers of these compromised nodes are assumed to be uniformly drawn random B-bit strings, denoted $\xi_1, \ldots, \xi_c$. We consider three attacks. In the first attack, the adversary aims at recovering the TTP's root keying material and therefore at attacking the whole system. In the second attack, the adversary aims at recovering the keying material of an uncompromised node $\xi \notin \{\xi_1, \ldots, \xi_c\}$. We shall analyze two methods for doing this. In the third attack, the adversary aims at recovering the pairwise key $K_{\xi,\eta}$ between two uncompromised nodes $\xi, \eta \notin \{\xi_1, \ldots, \xi_c\}$.

Notation. Most attacks in this section is based on lattices. A lattice is a set of all integer linear combination of some basis vectors of $\mathbb{Z}^d$. When a lattice L is formed from the basis matrix B, we denote $L = \mathcal{L}(B)$. For any two vectors u and v, we say that they are orthogonal if their inner product is zero, $u \cdot v = 0$. For a lattice $L \subset \mathbb{Z}^d$, we denote by $L^\perp$ the set of vectors orthogonal to vectors of L: $L^\perp = \{v \in \mathbb{Z}^d \ : \ v \cdot u = 0, \forall u \in L\}$. Note that $\dim L + \dim L^\perp = d$. For more properties and applications of orthogonal lattices, we refer [20].

3.1 Attacking the TTP

A coefficient of a node's keying material is obtained by mixing of operations with m different moduli (Equation 1). When attacking the TTP, the goal is to recover the root keying material so that the attacker can generate the complete keying material of any other node, and therefore, have access to all the communications of any device. For this, the attacker has to deal with the following problem.

Problem 1 (Mixing Modular Operations (MMO) Problems). Let the parameter set be $\{m, \alpha, \phi_0, \ldots, \phi_\alpha\}$, where $m \geq 2$, $\alpha \geq 0$, and $\phi_0, \ldots, \phi_\alpha$ are functions. Let x_i, $(0 \leq i \leq c)$ be given and let it be given that $y_i = \sum_{j=1}^{m} \langle \sum_{k=0}^{\alpha} g_{j,k} \phi_k(x_i) \rangle_{q_j}$, for $0 \leq i \leq c$, for some integer coefficients $g_{j,k}$ and moduli q_j.
MMO problem: given the parameter set, the number x_0 and the pairs of numbers (x_i, y_i) for $1 \leq i \leq c$, output an estimate $\hat{y}_0$ of y_0.
MMO problem with known moduli: given the input of the MMO problem and, additionally, the moduli q_j $(1 \leq j \leq m)$, output an estimate $\hat{y}_0$ of y_0.

If $\hat{y}_0 = y_0$, we say that the solution to the MMO problem is exact.

The general strategy for solving the MMO problem is to first guess $q_1, \ldots, q_m$ and then solve the MMO problem with known moduli. In [11], a strategy for recovering $q_1, \ldots, q_m$ is presented that relies on applying finite differences to obtain a set of equations that only depend on $q_1, \ldots, q_m$. This method does not seem to scale well with increasing m, so that guessing the moduli remains difficult. Note that most MMO challenges remain unsolved in the contest as well.

In [13], the exact MMO problem with known moduli was studied for the case $m = 2$ and $\phi_k(x) = x^k$ for $0 \leq k \leq \alpha$. Generalizing this method, it can be shown that solving the exact MMO problem with known moduli and c colluding nodes requires solving a close vector problem in a lattice with dimension $m(\alpha + 1) + (m - 1)c$, and that c must be at least $m(\alpha + 1)$ to find a unique solution. Thus the adversary has to solve a close vector problem in a lattice of dimension at least $m^2(\alpha + 1)$, which quickly becomes infeasible if m grows.

In the following, we first show that obtaining an approximate solution $\hat{y}_0$ to the MMO problem can be useful to attack a single node, if $\langle \hat{y}_0 \rangle_N$ differs a small multiple integer from $\langle y_0 \rangle_N$.

3.2 Attacking a single node

Let the coefficients of node ξ's keying material, as provided by the TTP, be $G_k(\xi)$, $0 \leq k \leq \alpha$. The key $K_{\xi,\eta}$ is given by

$$K_{\xi,\eta} = \big\langle \big\langle \sum_{k=0}^{\alpha} G_k(\xi)\phi_k(\eta) \big\rangle_N \big\rangle_{2^b}.$$

If the adversary finds an estimate g_k for $G_k(\xi)$ such that for each k

$$g_k = G_k(\xi) + 2^b\epsilon_k \text{ with } \max_{\eta\in[0,2^B)} |\epsilon_k\phi_k(\eta)| < N/2^b, \tag{4}$$

then

$$\begin{aligned}\big\langle \sum_{k=0}^{\alpha} g_k\phi_k(\eta) \big\rangle_N &= \big\langle \sum_{k=0}^{\alpha} G_k(\xi)\phi_k(\eta) + 2^b \sum_{k=0}^{\alpha} \epsilon_k\phi_k(\eta) \big\rangle_N \\ &= \big\langle \sum_{k=0}^{\alpha} G_k(\xi)\phi_k(\eta) \big\rangle_N + 2^b \sum_{k=0}^{\alpha} \epsilon_k\phi_k(\eta) - \lambda N\end{aligned}$$

for some integer λ with $|\lambda| \leq (\alpha+1)$. Reducing this result modulo 2^b, it follows that the key calculated using the estimated keying material coefficients differs a small multiple of N from the real key, and is thus likely to differ a small enough multiple of N from the key $K_{\eta,\xi}$ calculated by node η.

A. Simulating the keying material from the keying materials of other nodes In terms of the root-keying material, the k-th coefficient of the keying material of node ξ is given by

$$G_k(\xi) = \big\langle \sum_{i=1}^{m} \big\langle \sum_{\ell=0}^{\alpha} R_{k\ell}^{(i)} \phi_\ell(\xi) \big\rangle_{q_i} \big\rangle_N,$$

so finding an estimate to $G_k(\xi)$, given the values $G_k(\xi_i)$, $1 \leq i \leq c$ amounts to solving an MMO problem, like in Section 3.1, where the adversary is allowed to make an error of the form $2^b\epsilon_k$, with $|\epsilon_k|$ small enough. The adversary can do this, even without knowing the secret moduli q_i, by using Orthogonal lattices in the following way.

Consider the k-th keying material coefficient of device ξ:

$$\begin{aligned}
G_k(\xi) &= \big\langle \sum_{i=1}^{m} \big\langle \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) \big\rangle_{q_i} \big\rangle_N \\
&= \big\langle \sum_{i=1}^{m} \big(\sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) - (N - 2^b \beta_i) \lfloor q_i^{-1} \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) \rfloor \big) \big\rangle_N \\
&= \big\langle \sum_{i=1}^{m} \big(\sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) + 2^b \beta_i \lfloor q_i^{-1} \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) \rfloor \big) \big\rangle_N \\
&= \big\langle \sum_{i=1}^{m} \big(\sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) + 2^b \beta_i (q_i^{-1} \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) - \varepsilon_{ik}(\xi)) \big) \big\rangle_N \\
&= \big\langle \sum_{i=1}^{m} \sum_{l=0}^{\alpha} \frac{N R_{kl}^{(i)}}{q_i} \phi_l(\xi) - \sum_{i=1}^{m} 2^b \beta_i \varepsilon_{ik}(\xi) \big\rangle_N
\end{aligned}$$

where we defined

$$\varepsilon_{ik}(\xi) = q_i^{-1} \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) - \lfloor q_i^{-1} \sum_{l=0}^{\alpha} R_{kl}^{(i)} \phi_l(\xi) \rfloor, \text{ with } 0 \le \varepsilon_{ik}(\xi) < 1.$$

The integral quantity between the $\langle\,\rangle_N$ brackets is written as the difference of a linear combination of the function $\phi_l(\xi)$ with rational coefficients and the sum $\sum_{i=1}^{m} 2^b \beta_i \varepsilon_{ik}(\xi)$.

Let $\xi_0 = \xi$ and consider the column vector $Y_k = (G_k(\xi_0), \ldots, G_k(\xi_c))^T$. It can be written in terms of the $(\alpha+1) \times (c+1)$ matrix

$$V_{ij} = \phi_i(\xi_j), \quad 0 \le i \le \alpha,\ 0 \le j \le c,$$

a column vector ρ in $\mathbb{Q}^{\alpha+1}$ of rational polynomial coefficients and the column vectors $E_{ik} = (\varepsilon_{ik}(\xi_0), \ldots \varepsilon_{ik}(\xi_c))^T$ as

$$Y_k = \big\langle V^T \rho - \sum_{i=1}^{m} 2^b \beta_i E_{ik} \big\rangle_N,$$

where the modulo-N operation works component-wise.

Let $c > \alpha$, and consider $\mathcal{L}(V)$, a $(\alpha+1)$-dimensional lattice generated by basis matrix V. From this, we compute an orthogonal lattice $\mathcal{L}(V)^\perp$ and let K be a short basis of this lattice. Then it is easy to see that $\dim \mathcal{L}(K) = c - \alpha$ and $K \cdot V^T = 0$ from the definition.

As $KV^T = 0$, it follows that

$$\begin{aligned}
KY_k &\equiv KV^T \rho - \sum_{i=1}^{m} 2^b \beta_i K E_{ik} \\
&\equiv -2^b K \sum_{i=1}^{m} \beta_i E_{ik} \pmod N,
\end{aligned}$$

and so

$$\left\{2^{-b}KY_k\right\}_N = -K\sum_{i=1}^{m}\beta_i E_{ik},$$

provided that the infinity norm of $-K\sum_{i=1}^{m}\beta_i E_{ik}$ is less than $N/2$.

To continue the attack, let $Y_{k0} = G_k(\xi_0)$ be the first component of Y_k, and let Y'_k be the column vector $Y'_k = (G_k(\xi_1), \ldots, G_k(\xi_c))^T$. Similarly, let $K = [K_0\,|K']$ where K_0 is the first column vector of K. Then the following holds:

$$\left\{2^{-b}KY_k\right\}_N = \left\{2^{-b}(K_0Y_{k0} + K'Y'_k)\right\}_N = -K\sum_{i=1}^{m}\beta_i E_{ik}.$$

Let w be a row vector such that $w = (-K\sum_{i=1}^{m}\beta_i E_{ik})^T$. Now consider the lattice generated by the basis matrix L:

$$L = \begin{pmatrix} 1 & 0 & C\cdot\langle 2^{-b}K_0^T\rangle_N \\ 0 & N/2 & C\cdot\langle 2^{-b}(K'Y'_k)^T\rangle_N \\ 0 & 0 & C\cdot N\cdot\mathbb{I}_{c-\alpha} \end{pmatrix}, \tag{5}$$

where $C \approx N/\|w\|$ and $\mathbb{I}_{c-\alpha}$ is the $(c-\alpha)\times(c-\alpha)$ identity matrix. Now from the construction, the lattice $\mathcal{L}(L)$ contains a vector:

$$v = (Y_{k0}, N/2, C\cdot w).$$

By Minkowski's theorem, the length of the shortest vector of this lattice is less than:

$$\lambda_1(\mathcal{L}(L)) \leq (\det L)^{\frac{1}{c-\alpha+2}}$$

where

$$\det L \approx C^{c-\alpha}N^{c-\alpha+1} \approx N^{2c-2\alpha+1}/\|w\|^{c-\alpha}.$$

As a consequence, if $\|w\| < N^{\frac{c-\alpha-1}{c-\alpha}}$, then $(\det L)^{\frac{1}{c-\alpha+2}} > N$, which means that if $\|w\|$ is much less than N, then we can hope to find v among the short vectors in this lattice by lattice reduction algorithms. And from v, we can obtain Y_{k0}.

However, $\mathcal{L}(L)$ contains a trivial short vector $v_1 = (2^b, 0, C\cdot\langle K_0^T\rangle_N)$ with $\|v_1\| < \|v\|$. Thus, the attacker can only find v modulo v_1 and this reveals approximations to $Y_{k0} = G_k(\xi_0)$, which can then be used to calculate an estimate for the key between node ξ_0 and any other node.

In order for this attack to work, we need $\|w\|$ to be smaller than N. If (3) is satisfied and N has bit length $2B+b$, then by using Equation 7 from the appendix, $\|w\| \approx 2^B 2^{\frac{(\alpha+1)B}{c-\alpha}}$ and $N \approx 2^{2B+b}$, we infer that it is required that $c > \alpha + \frac{(\alpha+1)B}{B+b}$. If $\phi_k(x) = x^i$ for $0 \leq i \leq \alpha$ and N has bitlength $(\alpha+1)B+b$, then by using Equation 6 from the appendix, we infer that $c > \alpha + \frac{1}{2}\frac{\alpha(\alpha+1)B}{(\alpha+1)B+b}$. Thus, the parameter α needs to be set such that finding short vectors in a $(c-\alpha)$-dimensional lattice $\mathcal{L}(K) \in \mathbb{Z}^{c+1}$ is hard.

Since the matrix K, with short basis vectors for the left integer kernel for V^T, does not depend on k, it can be reused for attacking each of the $\alpha+1$ coefficients of a device. Doing this, requires further solving $\alpha+1$ close vector problems, as the basis matrix L in (5) depends on k. However, these problems can be solved in parallel and in our experience are less complicated than finding the short matrix K.

B. Simulating the keying material from keys shared with other nodes
Another approach to find approximations to l $G_0(\xi),\ldots,G_\alpha(\xi)$ is by using the keys between ξ and the c nodes $\xi_1,\ldots,\xi_c$. To this end, the attacker computes the key $K_{\xi_j,\xi}=\langle G^{\xi_j}(\xi)\rangle_{2^b}$ for $1\le j\le c$, uses K_{ξ,ξ_j} as an approximation for $K_{\xi_j,\xi}$ and tries to obtain the $G_0(\xi),\ldots,G_\alpha(\xi)$ from the c equations

$$K_{\xi,\xi_j}=\Big\langle\langle\sum_{k=0}^{\alpha}G_k(\xi)\phi_k(\xi_j)\rangle_N\Big\rangle_{2^b}\text{ for }1\le j\le c.$$

This is an instance of the hiding information (HI) problem: the attacker tries to obtain $G_0(\xi),\ldots,G_\alpha(\xi)$ from the b least significant bits of the evaluation of G^ξ in $\xi_1,\ldots,\xi_c$. That is, part of the evaluation of G^ξ is hidden.

The algorithm for obtaining the coefficients $G_0(\xi),\ldots,G_\alpha(\xi)$, makes use of the equivalence of this reconstruction problem to a lattice problem, as described in [19]. The lattice is spanned by the rows of the block matrix

$$\begin{pmatrix} N\mathbb{I}_c & 0\\ \mathbb{V} & 2^{-b}\mathbb{I}_{\alpha+1}\end{pmatrix},$$

where $\mathbb{I}_c$ and $\mathbb{I}_{\alpha+1}$ denote unit matrices of size $c\times c$ and $(\alpha+1)\times(\alpha+1)$ respectively, and $\mathbb{V}$ denotes the $(\alpha+1)\times c$ matrix with elements $V_{k,j}=\phi_k(\xi_j)$, $0\le k\le\alpha$, $1\le j\le c$. The problem is to find a lattice vector that lies inside a hypercube of edge length $N/2^b$ around a target vector that is constructed with the values K_{ξ,ξ_j}, $1\le j\le c$. For more details, we refer to [12, Sec. 6.2].

We now give a bit counting argument to see how large c should be in order to arrive at a correct solution. It is sufficient that the adversary finds an approximation g_k for $G_k(\xi)$ satisfying (4).
In the case of N independent of α, N has a length of $2B+b$ bits, and $\phi_k(x)\in[0,2^B)$. It is thus sufficient to obtain the B most significant bits and the b least significant bits of each $G_k(\xi)$. As each key yields b observed bits, and we need to estimate $(\alpha+1)(B+b)$ bits, we need that $c\ge(\alpha+1)\frac{B+b}{b}$.
If $\phi_k(x)=x^k$ for $0\le k\le\alpha$, and N has $(\alpha+1)B+b$ bits, we have in (4) that $\mid\epsilon_k\mid<2^{((\alpha+1)-k)B}$ bits for $0\le k\le\alpha$. This implies that we need to estimate $(\alpha+1)B+b-((\alpha+1-k)B)=b+kB$ bits of coefficient $G_k(\xi)$. The total numbers of bits to be estimated thus equals $(\alpha+1)b+\frac{1}{2}\alpha(\alpha+1)B$. As each key yields b observed bits, we need that $c\ge\frac{(\alpha+1)(2b+\alpha B)}{2b}$.

3.3 Finding a single key of a node from its keys with other nodes

In this attack, the adversary aims to find just a single key $K_{\eta,\xi}$ from observations of keys $K_{\xi_i,\eta}$ between the colluding nodes ξ_i, $1 \le i \le c$ and the node under attack η. To this end, a method similar to the one from Section 3.2.A can be used.

Since K_{η,ξ_i} and $K_{\xi_i,\eta}$ differ by a small multiple of N, modulo 2^b, the attacker uses $K_{\xi_i,\eta}$ as an approximation to $y_i = K(\eta, \xi_i)$ in the attack, and recovers $y_0 = K_{\eta,\xi}$ with an error which is small multiple of N modulo 2^b. Let G denote the column vector of node η's keying material coefficients, and Y the column vector of the y_i, $0 \le i \le c$. Then $Y = \langle \langle V^T G \rangle_N \rangle_{2^b}$, so that $\langle V^T G \rangle_N = Y + 2^b \Lambda$, where each integer component Λ_i satisfies $0 \le \Lambda_i \le N/2^b$. Multiplying by vK, where $v = \langle 2^{-b} \rangle_N$ and K is the kernel of V^T, and splitting K into its first column, L_0, and the rest, L_1, and similarly for the vector Y, we obtain

$$\left\{ y_0 v L_0 + v(L_1 Y_1) + K\Lambda \right\}_N = 0.$$

The components of $K\Lambda$ are of order $N' = \kappa N/2^b$, where is the order of magnitude of the is length of short vectors in the kernel. Consider the lattice basis matrix

$$\begin{pmatrix} N' & 2^b \langle v L_0^T \rangle_N \\ 0 & 2^b N \mathbb{I}_c \end{pmatrix}$$

and solve the CVP for target vector $(2^{b-1}N', -2^b \langle v(L_1 Y_1)^T + [N/2^{b+1}] K e \rangle_N$, where e is the all-ones column vector of length $c+1$. Let Z be the resulting lattice vector. Its first component Z_0 is a multiple of N' and the estimate for $K_{\eta,\xi}$ is obtained as $Z_0/N' + 2^{b-1}$.

For the above method to work, it is required that $N' < N$, or equivalently that $\kappa < 2^b$. For the case that the bit length of N depends on α, Equation 6 from the appendix implies that $\kappa = 2^{\alpha(\alpha+1)B/2(c-\alpha)}$, from which we infer the requirement $c > \alpha + \frac{1}{2}\alpha(\alpha+1)\frac{B}{b}$.

For the case of N of bit length independent of α, Equation 7 from the appendix implies that $k = 2^{(\alpha+1)B/(c-\alpha)}$, from which we infer the requirement $c > \alpha + (\alpha+1)\frac{B}{b}$.

The computational bottleneck in this attack is the reduction of the kernel, which has dimension $c - \alpha$.

3.4 Parameters

Table1 summarizes the results of the analysis. Disregarding the MMO attack, as it seems infeasible to find coefficients q_i for reasonable values of m, the attack on coefficients with the orthogonal lattice as described in Section 3.2.A yields the smallest lattice dimension.

Table 2 compares HIMMO when configured with N dependent on α ([12], [14]) or independent of α (this paper, N of bitlength $2B + b$ and the functions ϕ_k satisfying (3)). The table considers the two types of attacks described in Section 3.2, namely HI attack and approximate solution to the MMO problem, and the memory requirements to store the coefficients $G_k(\eta)$. When N depends on

Table 1. Minimum required number of colluding nodes, c_{min}, and lattice dimension related to various attacks described in the previous sections

	N dependent on α ([12], [14])	N independent of α (This paper)
Section 3.2.B	$\frac{(\alpha+1)(\alpha B+2b)}{2b}$, $c+\alpha+1$	$(\alpha+1)\frac{B+b}{b}$, $c+\alpha+1$
Section 3.2.A	$\alpha+\frac{\alpha(\alpha+1)B}{2((\alpha+1)B+b)}$, $c-\alpha$	$\alpha+(\alpha+1)\frac{B}{B+b}$, $c-\alpha$
Section 3.3	$\alpha+\frac{\alpha(\alpha+1)}{2}\frac{B}{b}$, $c-\alpha$	$\alpha+(\alpha+1)\frac{B}{b}$, $c-\alpha$

α, the memory requirements and lattice dimension associated to the HI problem are proportional to α^2. However, finding an approximate solution to the MMO problem only requires finding a reduced basis of a lattice of dimension $\sim \alpha/2$, when $b = B$. This method refines the method that was used to solve some of the HIMMO Challenges in the contest (`wwww.himmo-scheme.com`). This paper updates HIMMO parameters in such a way that N is independent of α. With this update, the memory requirements and the dimensions of the lattices associated to the HI problem and the approximate MMO problem both grow linearly with α, resulting in an improved security/complexity trade off.

Table 2. Performance and security level for N dependent on and independent of α for $b = B = 256$.

	N dependent on α ([12], [14])			N independent of α (this paper)		
α	Dimension HI	Dimension Orthogonal lattice Node	Memory	Dimension HI	Dimension Orthogonal lattice Node	Memory
25	351	13	10.97 KB	75	13	1.56 KB
50	1326	26	41.44 KB	150	26	3.13 KB
100	5151	50	160.97 KB	300	50	6.25 KB
800	320000	400	10.07 MB	2400	400	50.00 KB
1000	500000	500	15.67 MB	3000	500	62.50 KB
1500	1100000	750	35.23 MB	4500	750	93.75 KB
2000	2000000	1000	62.60 MB	6000	1000	125.00 KB
4000	8000000	2000	250.19 MB	12000	2000	250.00 KB
8000	32000000	4000	1 GB	24000	4000	500.00 KB

Next, we report on the performance of attacks presented in this paper. We took $b = B = 32$, the bit length of N equal to $2B + b$, and $\phi_k(x) = \langle\langle x^k\rangle_p\rangle_{2^B}$, where p is a prime slightly larger than 2^B. The attacks perform as follows:

– Guessing the root keying material to the TTP (Section 3.1) requires finding the exact solution to the MMO problem. This involves (i) guessing the secret q_is and then (ii) finding a close vector in a lattice of dimension $\approx m^2\alpha$. We

have only managed to solve this for very small parameters (till $m = 2$ and $\alpha = 5$).
- Guessing the key generating function of a node (Section 3.2) involves solving the HI problem or finding an approximate solution to the MMO problem that requires finding a close/short vector in a lattice of dimension that grows linearly with α, see Table 1. For the HI problem, experiments fail to find a correct solution with LLL when $\alpha > 180$. We have verified that the method for finding an approximate solution of the MMO problem works up to at least $\alpha = 500$, but expect it to fail for large values.
- Directly guessing a key of a node (Section 3.3) involves finding a short basis of a kernel of dimension $c - \alpha$, where c needs to be at least 2α. In this case, experiments using LLL fail to find the correct solution when $\alpha > 200$.

These experimental results are for LLL. Other algorithms to find an approximate reduced basis of a lattice of high dimension, such as BKZ, can achieve better performance, i.e., a shorter reduced basis (at the expense of a longer running time), but we expect them to fail also at some point, as reported in [12] for the case that N depends on α. In that case, the HI attack worked with LLL till an α value of 20 corresponding to a lattice dimension of 252. Using BKZ, the HI attack worked till $\alpha = 26$ corresponding to a lattice dimension of 405.

Thus, taking an ample margin, we suggest using the following parameters: $m = 10$, $\alpha \approx 2000$, $N \approx 2^{3B}$, $b = B$, and $\phi_k(x) = \langle\langle x^k \rangle_p \rangle_{2^B}$. These parameters lead to an efficient implementation for which all methods for guessing a key shared between a pair of nodes seem not to be feasible. We note that the scheme could be applied with an even larger α value, for instance, 8000 or even higher, while still being very efficient. This will be described in the next section.

4 Efficient Implementation and Performance

The design principles of HIMMO enable very efficient implementation. If N depends on α as $(\alpha + 1)B + b$ and $\phi_k(x) = x^k$, then Equation (2) represents the evaluation of a polynomial that can be implemented as described in [14]. If the modulus N is independent of α and its bit length is equal to $2B + b$, $B = b$ and $\phi_k(x) = \langle\langle x^k \rangle_p \rangle_{2^B}$, then key generation (Equation (2)) can be implemented as shown in Algorithm 1. This algorithm shows a loop with $\alpha + 1$ iterations, each iteration requiring a partial evaluation of $\phi_k(x)$ and the corresponding multiplication modulo N with the k^{th} coefficient. Evaluating $\phi_k(x)$ involves a reduction modulo p that can be efficiently implemented if $p = 2^B + r$ for a small integer r. The reduction modulo N can also be efficiently implemented by taking $N = 2^{3B} - 1$.

Table 3 provides preliminary performance results when the above algorithm runs on the 32-bit NXP LPC1769 LPCXpresso Board at 120 MHz.The results are based on an implementation in C only, and C with some assembler optimizations.

Comparing the performance of HIMMO with N dependent on α ([14], [12]) and N independent of α leads to several conclusions. Firstly, the CPU time and memory requirements when N is (in)dependent of α are proportional to

Algorithm 1 Optimized key generation

INPUT: B, α, η, p $G_{\xi,j}$ with $j \in \{0, \ldots, \alpha\}$
OUTPUT: key
$\phi \leftarrow 1$
$ID \leftarrow 1$
$key \leftarrow 0$
for $j = 0$ **to** α **do**
  $\phi \leftarrow \langle \phi \cdot \eta \rangle_p$
  $ID \leftarrow \langle \phi \rangle_{2^B}$
  $key \leftarrow \left\langle key + \left\langle \langle G_{\xi,j} \rangle_{2^B} \cdot ID \right\rangle_{2^B} \right\rangle_{2^B}$
  $key \leftarrow \left\langle key + \left\langle \frac{\frac{G_{\xi,j}}{2^{2B}} \cdot ID}{2^B} \right\rangle_{2^B} \right\rangle_{2^B}$
end for
return key

Table 3. HIMMO performance for $B = b = 32$ as a function of α.

		α					
		800	1000	1500	2000	4000	8000
Keying material size (KB)		6.25	7.82	11.72	15.63	31.25	62.50
CPU time (msec) on NXP LPC1769	C	0.69	0.86	1.29	1.72	3.44	6.87
(32-bit @ 120 MHz)	C+ASM	0.29	0.36	0.54	0.72	1.44	2.88

(α) α^2. Thus, the proposed parameters in this paper allow us to achieve much better performance for a given value of α. For instance, for $\alpha = 2000$, if N is independent of α, we see in Table 2 that the keying material is around 500 times smaller than when N depends on α. Secondly, the usage of N independent of α leads to very low RAM requirements, basically only one or two numbers of size N need to be stored in RAM. Thirdly, in both cases, bandwidth consumption is low, since a pair of nodes, ξ and η can directly communicate with each other, as follows. The first party, ξ, computes a common key $K_{\xi,\eta}$ with η, uses this key to protect a message, and then sends to η the protected message together with some reconciliation data of $K_{\xi,\eta}$ so that the other party, η, can securely process the received information. This method can be easily extended to include the implicit verification of ξ's credentials [14], [12]. As the implementation is very efficient in RAM, computation time, and bandwidth, its energy consumption is small as well.

As the above values and analysis show, HIMMO is a key pre-distribution scheme that is both efficient and collusion resistant. Its features compare favorably with those of existing public-key schemes, in particular, if we consider devices and use cases with limited bandwidth or energy budget, or with strict timing requirements. One advantage is that HIMMO can achieve key agreement and credential verification in a single message, while with public-key several message exchanges are required. Additionally, most public-key systems, in par-

ticular, quantum-safe schemes, have very long public-key and signature sizes. For instance, XMSS has signatures of 8 KB [1], one NTRUMLS signature [16] is around 988 Bytes long, the original McEliece scheme involves extremely large public-keys of around 200 KB and variants are more efficient but still relatively bulky [4], NTRUEncrypt requires around 610 Bytes [23]. In HIMMO, the B-bits identities play a role similar to the public keys, and signing can be done implicitly. For instance, integration of HIMMO with DTLS [14] enables both key agreement and mutual authentication with very low overhead that would be suitable for lightweight use cases. Computation-wise HIMMO is also very efficient, even for large α parameters, achieving a performance for key generation similar to that of Curve 25519 [3]. Indeed, the authors of [7] report a computation time of 72 msec for generating a 32 Byte value with Curve25519 on a Cortex M0 processor at 50 MHz. This is comparable to the values for HIMMO that would be obtained if we take Table 3 as a reference.

HIMMO also supports a collusion-resistant TTP infrastructure ensuring that the manager of a single TTP (or an attacker breaking a TTP) cannot overhear all the communications. This can be achieved without increasing communication and computation overhead, see [14].

5 Conclusions

This paper has presented attacks and parameter choices for the HIMMO scheme. The attack based on orthogonal lattices as presented in Section 3.2 allows obtaining an approximate solution to the MMO problem, and in this way, to attack HIMMO. For the parameter choices from [12],[14], the dimension of the related lattice grows linearly in α, but CPU and memory requirements scale quadratically in α. With the parameter choices presented in this paper, the CPU and memory requirements and the dimension of the lattices associated to all attacks described in this paper depend linearly on α, and a high α value can be used. A value of $\alpha = 2000$ seems to be a secure choice while still enabling a very well performing scheme in terms of CPU time and energy consumption. In particular, the fact that HIMMO relies on identities leads to a quasi-non-interactive key exchange while still enabling multiple protocol extensions such as the implicit certification and verification of credentials.

Our future work firstly focuses on the further security analysis of HIMMO and its operational features. To this end, we will analyze the performance of the presented attacks when other lattice reduction algorithms are applied. We are studying the usage of the HI and MMO problems for the design of other security building blocks enabling additional functionality beyond key pre-distribution. We are also analyzing how HIMMO can be used to enable more secure, efficient and scalable security architectures for the Internet.

References

1. S. Gazdag A. Hlsing, D. Butin and A. Mohaisen. XMSS: Extended hash-based signatures (Draft RFC). draft-irtf-cfrg-xmss-hash-based-signatures-01, 2015.

2. Martin Albrecht, Craig Gentry, Shai Halevi, and Jonathan Katz. Attacking Cryptographic Schemes Based on "Perturbation Polynomials". In *CCS09, Proc. 16th ACM Conference on computer and communications security*, pages 1–10. ACM, 2009.
3. Daniel J. Bernstein. *Public Key Cryptography - PKC 2006: 9th International Conference on Theory and Practice in Public-Key Cryptography, New York, NY, USA, April 24-26, 2006. Proceedings*, chapter Curve25519: New Diffie-Hellman Speed Records, pages 207–228. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
4. Daniel J. Bernstein, Tanja Lange, and Christiane Peters. *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 – December 2, 2011. Proceedings*, chapter Wild McEliece Incognito, pages 244–254. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
5. C. Blundo, A. de Santis, A.Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly secure key distribution for dynamic conferences. *Information and Computation*, 146:1–23, 1998.
6. Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology-ASIACRYPT 2013*, pages 280–300. Springer, 2013.
7. Michael Düll, Björn Haase, Gesine Hinterwälder, Michael Hutter, Christof Paar, Ana Helena Sánchez, and Peter Schwabe. High-speed curve25519 on 8-bit, 16-bit, and 32-bit microcontrollers. *Designs, Codes and Cryptography*, 77(2):493–514, 2015.
8. Régis Dupont and Andreas Enge. Provably secure non-interactive key distribution based on pairings. *Discrete Applied Mathematics*, 154(2):270–276, 2006.
9. Eduarda S.V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. Non-interactive key exchange. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 254–271. Springer Berlin Heidelberg, 2013.
10. Eduarda SV Freire, Dennis Hofheinz, Kenneth G Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *Advances in Cryptology–CRYPTO 2013*, pages 513–530. Springer, 2013.
11. O. Garcia-Morchon, R. Rietman, I. Shparlinski, and L. Tolhuizen. Results on polynomial interpolation with mixed modular opeations and unknown moduli. Cryptology ePrint Archive, Report 2015/1003, 2015. `http:eprint.iacr.org`.
12. O. García-Morchón, R. Rietman, L. Tolhuizen, D. Gómez-Pérez, and J. Gutiérrez. HIMMO - A Lightweight, Fully Colluison Resistant Key-Predistribution Scheme. Cryptology ePrint Archive, Report 2014/698, 2014. `http://eprint.iacr.org/`.
13. Oscar García-Morchón, Domingo Gómez-Pérez, Jaime Gutiérrez, Ronald Rietman, and Ludo Tolhuizen. The MMO problem. In *Proc. ISSAC'14*, pages 186–193. ACM, 2014.
14. Oscar Garcia-Morchon, Ronald Rietman, Sahil Sharma, Ludo Tolhuizen, and JoseLuis Torre-Arce. Dtls-himmo: Achieving dtls certificate security with symmetric key overhead. In Gnther Pernul, Peter Y A Ryan, and Edgar Weippl, editors, *Computer Security – ESORICS 2015*, volume 9326 of *Lecture Notes in Computer Science*, pages 224–242. Springer International Publishing, 2015.
15. Rosario Gennaro, Shai Halvei, Hugo Krawczyk, Tal Rabin, Steffen Reidt, and Stephen D. Wolthusen. Strongly-resilient and non-interactive hierarchical key-agreement in manets. In *ESORICS 2008*, volume 5283 of *Lecture Notes in Computer Science*, pages 49–65. Springer, 2008.

16. C. J. Hoffstein, J. Pipher, J. M. Schanck, J. H. Silverman, and W. Whyte. Transcript secure signatures based on modular lattices. In *Post-Quantum Conference 2014*, 2014.
17. Xinyu Lei and Xiaofeng Liao. NTRU-KE: A lattice-based public key exchange protocol. Cryptology ePrint Archive, Report 2013/718, 2013.
18. T. Matsumoto and H. Imai. On the key predistribution system: a practical solution to the key distribution problem. In C. Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, LNCS 293, pages 185–193. Springer, 1988.
19. Oscar García Morchon, Ronald Rietman, Igor E. Shparlinski, and Ludo Tolhuizen. Interpolation and approximation of polynomials in finite fields over a short interval from noisy values. *Experimental mathematics*, 23:241–260, 2014.
20. Phong Q. Nguyen and Jacques Stern. Merkle-Hellman revisited: A cryptoanalysis of the Qu-Vanstone cryptosystem based on group factorizations. In Burton S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *LNCS*, pages 198–212. Springer, 1997.
21. Kenneth G Paterson and Sriramkrishnan Srinivasan. On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. *Designs, Codes and Cryptography*, 52(2):219–241, 2009.
22. Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *The 2000 Symposium on Cryptography and Information Security, Okinawa, Japan*, pages 135–148, 2000.
23. W. Whyte. EEES1: Implementation aspects of NTRUEncrypt, Version 3.1,. online, 2015. `https://github.com/NTRUOpenSourceProject/ntru-crypto/blob/master/doc/EESS1-v3.1.pdf`.
24. W. Zhang, M. Tran, S. Zhu, and G. Cao. A Random Perturbation-based Scheme for Pairwise Key Establishment in Sensor Networks. In *8th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc) 2007*, pages 90–99, 2007.

Appendix: lattice heuristics

In our analysis, we needed estimates for the length of short vectors in a lattice. In this appendix, we describe the heuristics that we used for obtaining this estimate. We consider basic functions $\phi_0, \ldots, \phi_\alpha$ and distinct elements $\xi_0, \ldots, \xi_c \in [0, 2^B)$ and let V be an $(\alpha+1) \times (c+1)$ integer matrix with rows $v_0, \ldots, v_\alpha$, defined as $V_{i,j} = \phi_i(\xi_j)$ for $0 \leq i \leq \alpha$ and $0 \leq j \leq c$. The lattice obtained by taking integral linear combinations of these rows will be denoted by $\mathcal{L}$. Since V is an integral matrix, $\mathcal{L}$ is a subset of $\mathbb{Z}^{c+1}$, and as each point of $\mathcal{L}$ lies in the span of $v_0, \ldots, v_\alpha$, it holds that

$$\mathcal{L} \subseteq \mathrm{span}(v_0, \ldots, v_\alpha) \cap \mathbb{Z}^{c+1}.$$

The integer kernel $\mathcal{K}$ consists of the integral vectors that are orthogonal to all rows of V, so

$$\mathcal{K} = (\mathrm{span}(v_0, \ldots, v_\alpha))^\perp \cap \mathbb{Z}^{c+1}.$$

In order to obtain estimates for the length of small vectors in $\mathcal{K}$, we consider lattice volumes. We clearly have that

$$\mathrm{vol}(\mathcal{L}) \geq \mathrm{vol}(\mathrm{span}(v_0, \ldots, v_\alpha) \cap \mathbb{Z}^{c+1}),$$

and by definition

$$\mathrm{vol}(\mathcal{K}) = \mathrm{vol}((\mathrm{span}(v_0, \ldots, v_\alpha))^\perp \cap \mathbb{Z}^{c+1}).$$

For any m-dimensional linear subspace S of $\mathbb{R}^n$, $m < n$, it holds that $\mathrm{vol}(S \cap \mathbb{Z}^n) = \mathrm{vol}(S^\perp \cap \mathbb{Z}^n)$, so the two volumes on the right hand sides of the above equations are equal, and hence $\mathrm{vol}(\mathcal{K}) \leq \mathrm{vol}(\mathcal{L})$.

The lattice volume of $\mathcal{L}$ is upper bounded by the product of the lengths of the basis vectors which are upper bounded by $\sqrt{c+1}$ times the upper bound for its elements: $\mathrm{vol}(\mathcal{L}) < \prod_{i=0}^{\alpha} \sqrt{c+1} \max_{\eta \in [0, 2^B)} \phi_i(\eta)$. As a consequence,

$$\mathrm{vol}(\mathcal{L}) < (c+1)^{\frac{1}{2}(\alpha+1)}\ 2^{\frac{1}{2}\alpha(\alpha+1)B} \text{ if } \phi_i(x) = x^i,$$

$$\mathrm{vol}(\mathcal{L}) < (c+1)^{\frac{1}{2}(\alpha+1)}\ 2^{(\alpha+1)B} \text{ if } \phi_i(x) \leq 2^B$$

By the Gaussian heuristic, short vectors in $\mathcal{K}$ are expected to have a length $\sim (\mathrm{vol}(\mathcal{K}))^{1/(c-\alpha)}$. In our heuristics, we assume that $\mathcal{K}$ and $\mathcal{L}$ have approximately equal volumes, and approximate the length of short vectors in $\mathcal{K}$ by $(\mathrm{vol}(\mathcal{L}))^{1/(c-\alpha)}$. As a result, the length of short vectors in $\mathcal{K}$ is approximated by

$$(c+1)^{\frac{\alpha+1}{2(c-\alpha)}}\ 2^{\frac{\alpha(\alpha+1)}{2(c-\alpha)}B} \text{ if } \phi_i(x) = x^i, \tag{6}$$

and by

$$(c+1)^{\frac{\alpha+1}{2(c-\alpha)}}\ 2^{\frac{\alpha+1}{c-\alpha}B} \text{ if } \phi_i(x) \leq 2^B. \tag{7}$$