# High-Performance and Lightweight Lattice-Based Public-Key Encryption

Johannes Buchmann
Technische Universität Darmstadt, Germany
buchmann@cdc.informatik.tu-darmstadt.de

Florian Göpfert
Technische Universität Darmstadt, Germany
fgoepfert@cdc.informatik.tu-darmstadt.de

Tim Güneysu
University of Bremen, Germany
tim.gueneysu@uni-bremen.de

Tobias Oder
Ruhr-University Bochum, Germany
tobias.oder@ruhr-uni-bochum.de

Thomas Pöppelmann
Infineon Technologies AG
thomas.poeppelmann@infineon.com

## ABSTRACT

In the emerging Internet of Things, lightweight public-key cryptography is an essential component for many cost-efficient security solutions. Since conventional public-key schemes, such as ECC and RSA, remain expensive and energy hungry even after aggressive optimization, this work investigates a possible alternative. In particular, we show the practical potential of replacing the Gaussian noise distribution in the Ring-LWE based encryption scheme by Lindner and Peikert/Lyubashevsky et al. with a binary distribution. When parameters are carefully chosen, our construction is resistant against any state-of-the-art cryptanalytic techniques (e.g., attacks on original Ring-LWE or NTRU) and suitable for low-cost scenarios. In the end, our scheme can enable public-key encryption even on very small and low-cost 8-bit (ATXmega128) and 32-bit (Cortex-M0) microcontrollers.

## Keywords

Public key encryption, High-speed implementation, Lightweight cryptography, Ring learning with errors.

## 1. INTRODUCTION

Two important challenges in the Internet of Things (IoT) are secure (ad-hoc) communication and the establishment of temporary session keys. While symmetric cryptography has become remarkably efficient (c.f., lightweight ciphers like PRESENT [9]), this is currently not the case for asymmetric cryptography which is usually considered expensive for device metrics such as code size, chip area, runtime, or energy consumption. In general, the availability of inexpensive asymmetric schemes could help to eliminate key distribution problems and preshared keys (e.g., master/manufacturer keys) which frequently result in security problems in real-world applications (see [17, 49]).

An encryption scheme suitable to establish session keys for IoT devices has to fulfill several requirements. Most of them stem from the fact that the scheme is only run once per session. Consequently, it is very important that the scheme does not occupy many resources while it is not used, which makes small key and code sizes the primary goal. The second goal is fast key generation, encryption, and decryption. Furthermore, it is advantageous in practice to avoid large multipliers, samplers for complex noise distributions, or expensive input/output transformations even though this makes a scheme much less appealing from a theoretical point of view. Finally, we consider it a tremendous advantage if a scheme is as simple as possible and allows easy testing and auditing of the code usually written by non-experts.

*Related Work.*

Although several hardware implementations of public key encryption schemes and key exchange protocols have been proposed, they often require a significant amount of device resources. The most promising schemes are based on problems related to number theory, lattices, or codes. Until now, the main approach to meet the desire for public key cryptography with a low resource footprint have been efficient implementations of RSA or ECC for a specific target platform [30, 43, 22, 7, 6, 23]. Implementations of code-based cryptography still suffer from large key sizes [16] or, with more structured codes, from running times of several hundreds of milliseconds [25]. A great step forward towards an efficient asymmetric scheme is NTRU [27] which recently moved back into focus due to the increased popularity of lattice-based cryptography and which has been successfully implemented on several architectures [5, 24]. However, drawbacks like costly key generation and the need for complex message transformations in order to achieve CPA security have presumably prevented a wide adoption in practice so far. Note that the NTRU variant proposed by Stehlé and Steinfeld [47] is an interesting work from a theoretical perspective but does not lead to a practical scheme [12].



*IoTPTS'16, May 30-June 30 2016, Xi'an, China*



DOI: http://dx.doi.org/10.1145/2899007.2899011

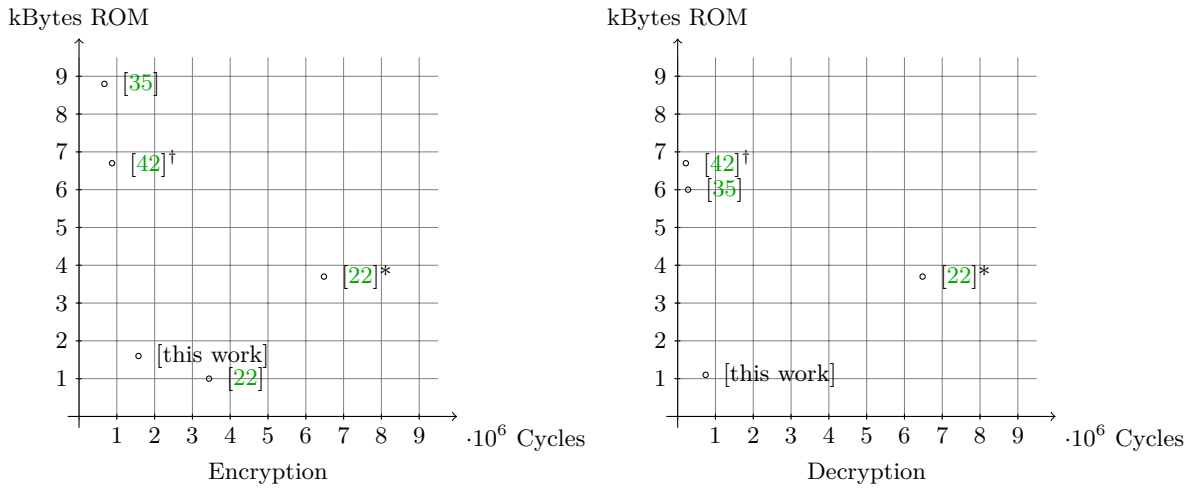


**Figure 1: Comparison of memory consumption and speed of AVR implementations. Implementation [22]* is a point multiplication on an elliptic curve. Implementation [42]† gives no separate data about memory consumption for encryption and decryption.**

*Contribution.*

In this work we propose a variant of the ideal-lattice based encryption scheme R-LWEENC by Lindner and Peikert [32] in which we replace the Gaussian noise distribution with a uniform binary distribution and show how to handle the non-symmetric error during decryption. We show that it is very suitable for small devices by comparing an efficient implementation for an ARM and an AVR microcontroller with existing microcontroller implementations of public-key encryption schemes. By performing 21.2 encryptions per seconds on the AVR and 33 encryption per second on the ARM the achieved performance is comparable to other schemes while providing a much lower memory footprint of 1.6 kBytes for AVR and 3.5 kBytes for ARM. The same is true for decryption, for which we achieve a memory footprint of 1.1 kBytes (AVR) and 2.1 kBytes (ARM) while executing 45.7 decryptions per second on AVR and 79.4 decryptions per second on ARM. A comparison with other AVR implementations is given in Figure 1 and Table 2.

The security of the scheme is based on the hardness of *Ring-Binary LWE* (Ring-BLWE), a variant of the learning with errors problem [45]. Our security analysis indicates a security level of 84 bits for an implementation-friendly parameter set ($n = 256, q = 256$) based on state-of-the art cryptanalytic techniques [2, 28, 26, 15, 11].

## 2. RING-LWE BASED PUBLIC KEY ENCRYPTION SCHEME

In this section we introduce the required notation, describe our public key encryption scheme, show the probability analysis for decoding failures, and provide parameters.

### 2.1 Preliminaries

Since its introduction by Regev [45], the learning with error problem (LWE) served as a fundamental building block for an astonishing variety of cryptographic schemes. To set up an LWE-distribution for integers $n, q$, and an error distribution $\psi$ over $\mathbb{Z}_q$, one samples a secret $\mathbf{s} \in \mathbb{Z}_q^n$ according to $\psi^n$. To create an LWE-sample, one samples a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, an error $e \in \mathbb{Z}_q$ according to $\psi$, and outputs the tuple $(\mathbf{a}, b)$ with $b = \mathbf{a}^T\mathbf{s} + e$. The LWE-samples can be collected to get $\mathbf{b} = \mathbf{As} + \mathbf{e}$ for an $m \times n$ matrix $\mathbf{A}$, the secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ and two vectors $\mathbf{e}, \mathbf{b} \in \mathbb{Z}_q^m$. Note that LWE can also be defined with different distributions for error $\mathbf{s}$ and secret $\mathbf{e}$, but it is known that any LWE instance can be transformed into an LWE instance with secrets distributed according to the error distribution.

The most efficient lattice-based schemes are based on a more structured variant of LWE, called Ring-LWE [36]. While certain properties can be established for various rings, we define $\mathcal{R}$ as the ring $\mathbb{Z}[x]/\langle x^n + 1\rangle$ and $\mathcal{R}_q$ as $\mathbb{Z}_q[x]/\langle x^n + 1\rangle$ for an integer $q$, $\mathbb{Z}_q = \mathbb{Z}/(q\mathbb{Z})$, and a power of two $n$. We write elements $\mathbf{p} \in \mathcal{R}_q$ with maximum degree $n - 1$ as $\mathbf{p} = \sum_{i=0}^{n-1}[\mathbf{p}]_i x^i$ with $[\mathbf{p}]_i \in (-\lfloor q/2\rfloor + 1, \lfloor q/2\rfloor)$ being the $i$-th coefficient.

To setup a Ring-LWE-distribution for integers $n, q$, and an error distribution $\psi$ over $\mathcal{R}_q$, one samples a secret $\mathbf{s} \in \mathcal{R}_q$ according to $\psi$. To create a Ring-LWE-sample, one samples a polynomial $\mathbf{a} \in \mathcal{R}_q$ uniformly at random, and an error $\mathbf{e} \in \mathcal{R}_q$ according to $\psi$, and outputs the tuple $(\mathbf{a}, \mathbf{b})$ with $\mathbf{b} = \mathbf{as} + \mathbf{e}$.

The search variant of the (Ring-)LWE problem is to find $\mathbf{s}$, given arbitrary many (Ring-)LWE-samples. The decision variant is to distinguish between arbitrary many (Ring-)LWE-samples and the same number of samples with uniform $\mathbf{a}$ and $b$ (respectively $\mathbf{a}$ and $\mathbf{b}$).

Typical choices for $\psi$ are discrete (or discretized) Gaussian distributions or uniform distributions over a small set. Complementary to Ring-LWE, we define the Ring-BLWE problem as the instance of Ring-LWE, where $\psi$ is the uniform distribution on the polynomials in $\mathcal{R}_q$ with binary coefficients (i.e., coefficients in $\{0, 1\}$).

### 2.2 The Scheme

In [32, 36] a semantically secure public key encryption scheme (from now referred to as R-LWEENC) is described whose security is based on the hardness of the Ring-LWE problem. While it has been

| | |
|---|---|
| $\text{GEN}(\mathbf{a})$: | Choose $\mathbf{r}_1, \mathbf{r}_2$ uniformly at random among the polynomials in $\mathcal{R}_q$ with binary entries and let $\mathbf{p} = \mathbf{r}_1 - \mathbf{a}\mathbf{r}_2 \in \mathcal{R}_q$. The public key is $\mathbf{p}$ and the secret key is $\mathbf{r}_2$. |
| $\text{ENC}(\mathbf{a}, \mathbf{p}, \mathbf{m} \in \{0,1\}^n)$: | Choose $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ uniformly at random among the polynomials in $\mathcal{R}_q$ with binary entries. Let $\bar{\mathbf{m}} = \text{ENCODE}(\mathbf{m}) \in \mathcal{R}_q$, and compute the ciphertext $[\mathbf{c}_1 = \mathbf{a}\mathbf{e}_1 + \mathbf{e}_2, \mathbf{c}_2 = \mathbf{p}\mathbf{e}_1 + \mathbf{e}_3 + \bar{\mathbf{m}}] \in \mathcal{R}_q^2$. |
| $\text{DEC}(\mathbf{c} = [\mathbf{c}_1, \mathbf{c}_2], \mathbf{r}_2)$: | Output $\text{DECODE}(\mathbf{c}_1\mathbf{r}_2 + \mathbf{c}_2) \in \{0,1\}^n$. |

**Figure 2: The Scheme R-BinLWEEnc**

shown that high-performance implementations are possible [20, 46, 14], the scheme still has the disadvantage of large ciphertexts and requires complex sampling of discrete Gaussian noise. However, its simple structure can be used as starting point to derive a variant which is much better suited for practice. The main difference is that we now use binary errors and secrets instead of errors or secrets chosen from a Gaussian distribution. While just using binary noise in the R-LWEEnc scheme seems straightforward, the choice of the encoding and decoding functions and the decryption error analysis is not. The scheme (from now referred to as R-BinLWEEnc) is defined in Figure 2. It is parameterized by integers $n$ and $q$ and uses a uniformly random chosen global constant polynomial $\mathbf{a} \in \mathcal{R}_q$. Furthermore, it requires a pair of error-tolerating encoding and decoding functions. Our instantiations of these functions are given in Equations (1) and (2), the justification for this choices in Section 2.3.

As we replace Gaussian by binary noise our scheme is now based on the Ring-BLWE problem whose hardness is assessed in Section 3. Following [32], we can conclude that the scheme remains semantically secure as long as decisional Ring-BLWE in dimension $n$ with modulus $q$ is hard. We note that in R-BinLWEEnc, $\mathbf{r}_1$ in Gen is noise and hence not needed anymore after key generation. The scheme uses the error-tolerant encoding and decoding functions

$$\text{ENCODE: } \{0,1\}^n \to \mathcal{R}_q, \quad (m_0, \ldots, m_{n-1}) \mapsto \sum_{i=0}^{n-1} m_i \cdot (q/2) x^i \tag{1}$$

and $\text{DECODE} : \mathcal{R}_q \to \{0,1\}^n$ as given in Equation (2). Note that DECODE differs from the decoding function by Lindner and Peikert. This is due to the fact that the binary distribution is (unlike the Gaussian distribution) typically not centered around zero and this asymmetry of the error leads to an asymmetry of the coefficients. In the next section, we discuss the implications of the different decoding function on the correctness of the scheme.

## 2.3 Correctness

Similar to the correctness result of [32], the message is decrypted correctly if (and only if)

$$\text{DECODE}(\text{ENCODE}(\mathbf{m}) + \mathbf{e}_1\mathbf{r}_1 + \mathbf{e}_2\mathbf{r}_2 + \mathbf{e}_3) = \mathbf{m}.$$

Note that for all $i \in \{1,2\}$ and $k \in \{0, \ldots, n-1\}$

$$[\mathbf{e}_i\mathbf{r}_i]_k = \sum_{j=0}^{k} [\mathbf{e}_i]_j [\mathbf{r}_i]_{k-j} - \sum_{j=k+1}^{n-1} [\mathbf{e}_i]_j [\mathbf{r}_i]_{n+k-j},$$

and the coefficients $[\mathbf{e}_i]_j, [\mathbf{r}_i]_j$ are statistically independent and binary. Consequently, the coefficients $[\mathbf{e}_i\mathbf{r}_i]_k$ are approximately distributed according to a Gaussian distribution modulo $q$ (this is basically a random walk). Therefore, the distribution of the coefficients of the noise polynomial $\mathbf{n} = \mathbf{e}_1\mathbf{r}_1 + \mathbf{e}_2\mathbf{r}_2 + \mathbf{e}_3$ is likewise close to a Gaussian distribution. The natural choice for the decoding function is therefore to determine the expected values $\text{E}([\mathbf{n}]_k)$ and decode all coefficients closer to $\text{ENCODE}(0) + \text{E}([\mathbf{n}]_k)$ to zero and the elements closer to $\text{ENCODE}(1) + \text{E}([\mathbf{n}]_k)$ to one. Since

$$\begin{aligned}\text{E}([\mathbf{e}_i\mathbf{r}_i]_k) &= \text{E}(\sum_{j=0}^{k} [\mathbf{e}_i]_j [\mathbf{r}_i]_{k-j} - \sum_{j=k+1}^{n-1} [\mathbf{e}_i]_j [\mathbf{r}_i]_{n+k-j}) \\ &= (-n + 2k + 2)/4,\end{aligned}$$

the desired expected value is

$$\begin{aligned}\text{E}([\mathbf{n}]_k) &= \text{E}([\mathbf{e}_1\mathbf{r}_1]_k) + \text{E}([\mathbf{e}_2\mathbf{r}_2]_k) + \text{E}([\mathbf{e}_3]_k) \\ &= k - n/2 + 3/2.\end{aligned}$$

Therefore, we choose the decoding function

$$\text{DECODE: } \mathcal{R}_q \to \{0,1\}^n, \quad \sum_{k=0}^{n-1} \alpha_k x^k \mapsto (m_0, \ldots, m_{n-1}) \tag{2}$$

with

$$m_k = \begin{cases} 0 & \text{if } |\alpha_k - k - \lfloor \frac{n-3}{2} \rceil| \leq \frac{q}{4} \\ 1 & \text{else.} \end{cases}$$

Please note that since all the above probability distribution have finite support, it is possible to calculate the probability of decoding errors exactly. In fact, we used Sage [48], a computer algebra program, to calculate the probabilities in Table 1.

## 2.4 Parameter Selection

In Table 1 we provide three parameter sets for R-BinLWEEnc based on the security analysis (see Section 3). For comparison, we also include selected NTRU [26] and Ring-LWE Encryption (R-LWEEnc) [32] parameter sets. Note that it is also possible to use a $q$ between 128 and 512 to balance security and error probability between the different parameter sets given in Table 1. While even a small failure probability like $2^{-32}$ is clearly undesirable in practice, some applications are able to deal well with such a small probability (e.g., interactive applications already have to account for data corruption during transmission). It can further be seen that our proposal leads to smaller key and ciphertext sizes than

| Set/Scheme | $n$ | $q$ | Bit Sec. | Failure Probability | Size [bits] | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | Message | Secret Key | Public Key | Ciphertext |
| R-BinLWEEnc-I | 256 | 128 | 94 | $2^{-10}$ | 256 | 256 | 1,792 | 3,584 |
| R-BinLWEEnc-II | 256 | 256 | 84 | $2^{-32}$ | 256 | 256 | 2,048 | 4,096 |
| R-BinLWEEnc-III | 512 | 256 | 190 | $2^{-18}$ | 512 | 512 | 4,096 | 8,192 |
| NTRU ($d_f = 113$) [26] | 401 | 2048 | 112 | $< 2^{-112}$ | 401 | 636 | 4,401 | 4,401 |
| NTRU ($d_f = 49$) [26] | 541 | 2048 | 112 | $< 2^{-112}$ | 541 | 858 | 5,951 | 5,951 |
| NTRU ($d_f = 38$) [26] | 659 | 2048 | 112 | $< 2^{-112}$ | 659 | 1045 | 7,249 | 7,249 |
| R-LWEEnc [20] | 256 | 7681 | 106 | $\approx 2^{-7}$ | 256 | 1,504 | 3,304 | 6,608 |
| R-LWEEnc [20] | 512 | 12289 | 157 | $\approx 2^{-7}$ | 512 | 3,062 | 6,956 | 13,912 |

**Table 1: Proposed Parameter Sets for R-BinLWEEnc. Hardness Result for LWE taken from [34].**

R-LWEEnc and is comparable to NTRU. Only the size optimized (but computationally more complex) parameter set NTRU ($d_f = 113$) allows a slightly smaller ciphertext. For a more detailed comparison of different lattice-based encryption schemes we refer to [12]. Additionally, we would like to note that the ciphertext size could presumably be reduced in future work by removing redundant information in the ciphertext [41] or by techniques presented by Peikert [40].

# 3. HARDNESS ASSESSMENT OF BINARY LWE

In this section, we discuss the theoretical and concrete hardness of Ring-BLWE.

## 3.1 Theoretical Hardness of Ring-BLWE

When Regev [45] introduced LWE, he provided a quantum reduction that showed its worst-case hardness if at least one of two well-known lattice problems (namely gapSVP, the decisional variant of the shortest vector problem SVP, and SIVP, the shortest independent vector problem) is hard in the average case. Nowadays, there are classical reductions [10] and worst case results for LWE with uniformly distributed error [37], for LWE instances with leaky secret [19], and for Ring-LWE [36].

Two of the above reductions can in principle be applied for LWE with binary error or secret. Goldwasser et al. [19] gave a reduction from LWE with binary (and possibly leaky) secret to LWE with uniform secret. However, these results are solely valid for LWE with Gaussian error and therefore not applicable to Ring-BLWE. Micciancio and Peikert [37] showed the worst-case hardness of LWE with uniform error distribution if the number of samples is restricted. Unfortunately, their worst case result for binary errors requires a strong restriction on the number of samples and furthermore does not transfer to the ring setting.

Consequently, R-BinLWEEnc is not worst-case secure, but only based on the average-case hardness of Ring-BLWE. Other examples of the common practice to base the security on average-case problems are the signature schemes by Lyubashevsky et al. [21, 15] and the NTRU encryption scheme [27]. Likewise, the parameter sets proposed by Lindner and Peikert for their encryption scheme [32] have not been shown to provide worst-case security.

## 3.2 Attacks on Ring-BLWE

Despite several algorithms dedicated to problems in ideal lattices [31, 18], there is still no breakthrough result that can break schemes based on ideal lattices considerably faster. In the lattice challenges [33], the current record for solving the shortest integer solution problem in standard lattices is dimension 140, while the record for SVP in ideal lattices is dimension 128. In particular it has not been shown that any known attack considerably profits from the additional structure of Ring-LWE. Hence, according to current knowledge, Ring-LWE is assumed to be not easier than LWE.

The hardness of BLWE was recently studied by Buchmann et al. [11]. The authors present a new attack, and compare it with existing approaches like the distinguishing attack [38, 32], the embedding approach [29], the decoding attack [32], and the meet-in-the-middle attack [3]. Additionally, they argue that algebraic attacks like the Arora-Ge algorithm [4, 1] or the BKW approach [8, 1] can not be applied to Ring-BLWE instances with few samples.

Applying the methods presented in [11] leads to the hardness estimations given in Table 1. For all instances, the hybrid approach [11] performed best.

# 4. IMPLEMENTATION AND PERFORMANCE EVALUATION

Due to the simple structure of R-BinLWEEnc it is well suited for an implementation on embedded devices. To underline the relevance of the scheme for the Internet of Things, we chose two low-cost microcontrollers as target platform for our implementation, namely the ARM Cortex-M0 and the Atmel AVR ATXmega128A1. While our AVR implementation is solely focused on low memory consumption, we apply a few optimization techniques to our ARM implementation to speed up the implementation at the cost of a slightly higher memory footprint.

## 4.1 Microcontroller Implementation

Since the scheme does not require costly Gaussian sampling, polynomial multiplication is the most expensive operation. To be more precise, the core operation is a multiplication of a polynomial with uniformly distributed coefficients by a polynomial with binary coefficients. That means that the product is computed by just shifting the input polynomial and adding up those intermediate polynomials. By shifting, we mean multiplying a polynomial $p(x)$ by x. The number of zero coefficients between two non-zero coefficients in the second (binary) input polynomial determines the shift width. This technique yields a very efficient multiplication without requiring complex algorithms like the number-theoretic transform. We exclude the key generation step from our work and instead store the precomputed keys in a non-volatile memory.

**Table 2: Cycle counts and flash consumption of our implementation of R-BinLWEEnc on an 8-bit ATxmega128 and 32-bit Cortex-M0 microcontroller both clocked with 32 MHz. The Flash memory consumption in bytes includes the public and secret key for encryption and decryption, respectively. We also compare our implementations with implementations of Ring-LWE, RSA, ECC, and QC-MDPC McEliece on AVR microcontrollers. Our implementations are marked with †.**

| Scheme | Device | Operation | Cycles/$10^3$ | | ROM/kBytes | |
|---|---|---|---|---|---|---|
| R-BinLWEEnc-I † ($\mathbf{n = 256, q = 128}$) | ATXmega (32 MHz) | Enc/Dec | 1,573 | 740 | 1.6 | 1.1 |
| R-BinLWEEnc-II † ($\mathbf{n = 256, q = 256}$) | ATXmega (32 MHz) | Enc/Dec | 1,507 | 700 | 1.6 | 1.1 |
| R-BinLWEEnc-III † ($\mathbf{n = 512, q = 256}$) | ATXmega (32 MHz) | Enc/Dec | 5,899 | 2,791 | 2.1 | 1.4 |
| R-BinLWEEnc-I † ($\mathbf{n = 256, q = 128}$) | Cortex-M0 (32 MHz) | Enc/Dec | 999 | 437 | 3.5 | 2.1 |
| R-BinLWEEnc-II † ($\mathbf{n = 256, q = 256}$) | Cortex-M0 (32 MHz) | Enc/Dec | 944 | 403 | 3.5 | 2.1 |
| R-BinLWEEnc-III † ($\mathbf{n = 512, q = 256}$) | Cortex-M0 (32 MHz) | Enc/Dec | 3,483 | 1,701 | 4.6 | 2.2 |
| Ring-LWE ($n = 256, q = 7681$)[35] | ATXmega (32 MHz) | Enc/Dec | 671 | 276 | 8.8 | 6.0 |
| Ring-LWE ($n = 512, q = 12289$)[35] | ATXmega (32 MHz) | Enc/Dec | 2,617 | 686 | 13.5 | 8.5 |
| Ring-LWE ($n = 256, q = 7681$)[42] | ATXmega (32 MHz) | Enc/Dec | 874 | 216 | 6.7 | |
| Ring-LWE ($n = 512, q = 12289$)[42] | ATXmega (32 MHz) | Enc/Dec | 2,197 | 600 | 9.3 | |
| QC-MDPC [25] | ATXmega (32 MHz) | Enc/Dec | 26,767 | 86,874 | 3.7 | 2.2 |
| RSA-1024 [22] | ATmega (8 MHz) | Enc/Dec | 3,440 | 87,920 | 1.0 | 6.3 |
| ECC-ecp160r1 [22] | ATmega (8 MHz) | Point mul. | 6,480 | | 3.7 | |

## 4.2 Cortex M0

The ARM Cortex-M0 is a low-cost 32-bit microcontroller and therefore of particular interest. Note that the Cortex-M0 is either shipped with a single cycle multiplier or without. The Infineon XMC 2Go evaluation board that we used for this work, does feature a single cycle multiplier, but, in contrast to other lattice-based cryptosystems, R-BinLWEEnc does not require MUL instructions and therefore the results should be similar on processors without single cycle multiplier. To obtain random numbers, we initialize the internal pseudo-random number generator (PRNG) with a secret bitstring. Every 9-10 clock cycles the PRNG generates an 8-bit random number.

To take advantage of the 32-bit architecture, we optimize the polynomial multiplication by vectorizing the operation. During the multiplication the highest value one coefficient can have without being reduced in the meantime is $256 \times 255 = 65280$ since there are at most 256 additions of values in $[0, 255]$. Therefore, we can store two coefficients into one data word. In the unvectorized multiplication, we do not actually shift the input polynomial, but adjust the base address of the accumulator instead, e.g. shifting a polynomial by three positions (resp. multiplying it by $x^3$) is realized by increasing the accumulators base address by $3 \times 4 = 12$. Keep in mind that we are dealing with 4-byte data words. Due to the fact that only 4-byte-aligned memory accesses are possible, this technique can only be applied to even shift widths in the vectorized multiplication. E.g., shifting by four positions means increasing the accumulators base address by 8. Shifting by three positions on the other hand would mean increasing the base address by 6, but then we no longer have a valid address for 4-byte memory accesses. To be able to handle odd shift widths as well, we store two variants of the input polynomial, the first one just as described and the second one already shifted by one position. That means we can still shift by three positions by increasing the base address of the accumulator by 4 and adding the input polynomial that is already shifted by one position. The advantage of this technique is that the multiplication is now almost twice as fast as in the unvector-

ized case, but on the downside, we need to compress and decompress the input polynomial before and after the multiplication. Note that the dynamic memory consumption is not increased by this method since we are now storing two input vectors of half-length instead of one vector of full-length. During the encryption both multiplications compute the product of a public key polynomial and a binary polynomial. Therefore, it would be possible to precompute the first rotation of the keys. But since this measure doubles the memory consumption for the key storage and our goal is to create a lightweight cipher for the Internet of Things, we decided to not apply this optimization but instead compute the first rotation on-the-fly.

### 4.3 Atmel AVR

Another low-cost microcontroller is the 8-bit Atmel AVR ATxmega128A1. Random numbers can be obtained from the internal true-random number generator (TRNG). We use the output of the TRNG to initialize the internal AES engine that produces 128 bit of randomness in 375 clock cycles. The AVR offers requires needs for optimization as the use of 7-bit or 8-bit wide coefficients fits already very well to the 8-bit architecture. But since there are already various implementations of encryption schemes on this platform, we evaluate the performance of R-BinLWEEnc for comparison as well. The advantage of an 8-bit architecture is that for $q = 256$ no modular reduction is necessary. Additionally, setting $n = 256$ reduces some loop overhead.

## 5. RESULTS

To evaluate the performance of R-BinLWEEnc on typical IoT devices, we implemented the scheme on an 8-bit Atmel AVR ATXmega128A1 microcontroller using AVR-GCC 4.7 with optimization flag `-Os` and on the ARM Cortex-M0 using armcc V5.06 with optimization flag `-O3`.

Results for our C implementation are given in Table 2. Our ARM implementation includes assembly optimization of the multiplication. For our AVR implementation, an assembly implementation did not provide significant savings due to the very simple nature of the polynomial multiplication algorithm and the straightforward mapping of polynomial coefficients to the `uint8_t` data type. Due to the higher level of optimization, as described in Section 4.2, our ARM implementation runs faster than our AVR implementation. Storing two key coefficients in one 32-bit data word instead of one coefficient in an 8-bit data word also doubles the memory requirement for the key storage.

The trade-off between memory consumption and speed for various ATXmega implementations is given in Figure 1. The points correspond to our implementation, two implementations of an instantiation of Ring-LWE with Gaussian error ($n = 256, q = 7681$, [35, 42]), an RSA-1024 implementation and an elliptic-curve point-multiplication [22]. Table 2 also contains an implementation of a code-based scheme [25], which was omitted in the figure since it is several orders of magnitude less space efficient than the other implementations. For the same reason, we omitted the RSA decryption implementation. Figure 1 highlights the extremely small memory footprint of our implementation.

In [35] two different optimization goals are given. The high-speed implementation outperforms our implementation on the same platform by a factor of 2.5. But this comes to no surprise since our main design goal is a low memory footprint and their high-speed implementation includes large precomputed tables for the number-theoretic transform. Their memory-efficient implementation performs comparable to our implementation (1,532,823 / 673,489 cycles for encryption / decryption) but is still much larger than our implementation (8,5 / 6,0 kBytes for encryption/decryption). The implementation of [42] is 1.8 times faster for encryption than ours (and 3.4 times faster for decryption) but also applies the number-theoretic transform with precomputed twiddle factors and therefore requires much more memory.

Translating the implementation results for RSA and ECC given in [22] to cycle counts, it turns out that an ECC secp160r1 operation requires 6.5 million cycles. RSA-1024 encryption with public key $e = 2^{16}+1$ is the only implementation that outperforms ours in terms of memory consumption, but instead only provides two times slower encryption and nearly 120 times slower decryption with Chinese Remainder Theorem (CRT). Compared to an implementation of a code-based scheme [25] we also achieve a much better performance and require less flash memory.

*Acknowledgement..*

This work has been co-funded by the DFG as part of project P1 within the CRC 1119 CROSSING and by the Commission of the European Communities through the Horizon 2020 program under project number 645622 PQCRYPTO and 644729 SAFEcrypto.

## 6. REFERENCES

[1] M. R. Albrecht, C. Cid, J. Faugère, R. Fitzpatrick, and L. Perret. Algebraic algorithms for LWE problems. *IACR Cryptology ePrint Archive*, 2014:1018, 2014.

[2] M. R. Albrecht, R. Fitzpatrick, and F. Göpfert. On the efficacy of solving LWE by reduction to unique-SVP. In H. Lee and D. Han, editors, *Information Security and Cryptology - ICISC 2013 - 16th International Conference, Seoul, Korea, November 27-29, 2013, Revised Selected Papers*, volume 8565 of *LNCS*, pages 293–310. Springer, 2013.

[3] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Mathematical Cryptology*, 9(3):169–203, 2015.

[4] S. Arora and R. Ge. New algorithms for learning in presence of errors. In *ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I*, pages 403–415, 2011.

[5] A. C. Atici, L. Batina, J. Fan, I. Verbauwhede, and S. B. Örs. Low-cost implementations of NTRU for pervasive security. In *19th IEEE International Conference on Application-Specific Systems, Architectures and Processors, ASAP 2008, July 2-4, 2008, Leuven, Belgium*, pages 79–84. IEEE Computer Society, 2008.

[6] L. Batina, D. Hwang, A. Hodjat, B. Preneel, and I. Verbauwhede. Hardware/software co-design for hyperelliptic curve cryptography (HECC) on the 8051$\mu$p. In J. R. Rao and B. Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *LNCS*, pages 106–118. Springer, 2005.

[7] L. Batina, N. Mentens, K. Sakiyama, B. Preneel, and I. Verbauwhede. Low-cost elliptic curve cryptography for wireless sensor networks. In L. Buttyán, V. D. Gligor, and D. Westhoff, editors, *Security and Privacy in Ad-Hoc and Sensor Networks, Third European Workshop, ESAS 2006, Hamburg, Germany, September 20-21, 2006, Revised Selected Papers*, volume 4357 of *LNCS*, pages 6–17. Springer, 2006.

[8] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. In F. F. Yao and E. M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 435–440. ACM, 2000.

[9] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In P. Paillier and I. Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[10] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 575–584, New York, NY, USA, 2013. ACM.

[11] J. Buchmann, F. Göpfert, R. Player, and T. Wunderer. On the hardness of LWE with binary error. Cryptology ePrint Archive, Report 2016/089, 2016. http://eprint.iacr.org/.

[12] D. Cabarcas, P. Weiden, and J. Buchmann. On the efficiency of provably secure NTRU. In Mosca [39], pages 22–39.

[13] R. Canetti and J. A. Garay, editors. *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *LNCS*. Springer, 2013.

[14] R. de Clercq, S. S. Roy, F. Vercauteren, and I. Verbauwhede. Efficient software implementation of Ring-LWE encryption. *IACR Cryptology ePrint Archive*, 2014:725, 2014.

[15] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal Gaussians. In Canetti and Garay [13], pages 40–56.

[16] T. Eisenbarth, T. Güneysu, S. Heyse, and C. Paar. Microeliece: Mceliece for embedded devices. In C. Clavier and K. Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, volume 5747 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2009.

[17] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. T. M. Shalmani. On the power of power analysis in the real world: A complete break of the keeloqcode hopping scheme. In D. Wagner, editor, *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *LNCS*, pages 203–220. Springer, 2008.

[18] K. Eisentraeger, S. Hallgren, and K. Lauter. Weak instances of PLWE. Cryptology ePrint Archive, Report 2014/784, 2014. http://eprint.iacr.org/.

[19] S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In A. C. Yao, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 230–240. Tsinghua University Press, 2010.

[20] N. Göttert, T. Feller, M. Schneider, J. Buchmann, and S. A. Huss. On the design of hardware building blocks for modern lattice-based encryption schemes. In Prouff and Schaumont [44], pages 512–529.

[21] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Prouff and Schaumont [44], pages 530–547.

[22] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit cpus. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *LNCS*, pages 119–132. Springer, 2004.

[23] M. N. Hassan and M. Benaissa. A scalable hardware/software co-design for elliptic curve cryptography on PicoBlaze microcontroller. In *International Symposium on Circuits and Systems (ISCAS 2010), May 30 - June 2, 2010, Paris, France*, pages 2111–2114. IEEE, 2010.

[24] J. Hermans, F. Vercauteren, and B. Preneel. Speed records for NTRU. In J. Pieprzyk, editor, *Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings*, volume 5985 of *LNCS*, pages 73–88. Springer, 2010.

[25] S. Heyse, I. von Maurich, and T. Güneysu. Smaller keys for code-based cryptography: QC-MDPC McEliece implementations on embedded devices. In G. Bertoni and J. Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, volume 8086 of *LNCS*, pages 273–292. Springer, 2013.

[26] P. S. Hirschhorn, J. Hoffstein, N. Howgrave-Graham, and W. Whyte. Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In M. Abdalla, D. Pointcheval, P. Fouque, and D. Vergnaud, editors, *Applied Cryptography and Network Security, 7th International Conference, ACNS 2009, Paris-Rocquencourt, France, June 2-5, 2009. Proceedings*, volume 5536 of *LNCS*, pages 437–455, 2009.

[27] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In J. Buhler, editor, *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.

[28] N. Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In A. Menezes, editor, *Advances in Cryptology - CRYPTO 2007*, volume 4622 of *LNCS*, pages 150–169. Springer Berlin Heidelberg, 2007.

[29] R. Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, Aug. 1987.

[30] Z. Khan and M. Benaissa. Low area ECC implementation on FPGA. In *Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on*, pages 581–584, Dec 2013.

[31] H. Lenstra and A. Silverberg. Revisiting the Gentry-Szydlo algorithm. In J. Garay and R. Gennaro, editors, *Advances in Cryptology – CRYPTO 2014*, volume 8616 of *LNCS*, pages 280–296. Springer Berlin Heidelberg, 2014.

[32] R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In A. Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer Berlin Heidelberg, 2011.

[33] R. Lindner, M. Rueckert, P. Baumann, and L. Nobach. Lattice challenge. http://www.latticechallenge.org/.

[34] M. Liu and P. Nguyen. Solving BDD by enumeration: An update. In E. Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *LNCS*, pages 293–309. Springer Berlin Heidelberg, 2013.

[35] Z. Liu, H. Seo, S. S. Roy, J. Großschädl, H. Kim, and I. Verbauwhede. Efficient ring-lwe encryption on 8-bit AVR processors. In T. Güneysu and H. Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *LNCS*, pages 663–682. Springer, 2015.

[36] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010.

[37] D. Micciancio and C. Peikert. Hardness of SIS and LWE with small parameters. In R. Canetti and J. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8042 of *LNCS*, pages 21–39. Springer Berlin Heidelberg, 2013.

[38] D. Micciancio and O. Regev. Lattice-based cryptography. In D. J. Bernstein, J. Buchmann, and E. Dahmen, editors, *Chapter in Post-quantum Cryptography*, pages 147–191. Springer, 2009.

[39] M. Mosca, editor. *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, volume 8772 of *LNCS*. Springer, 2014.

[40] C. Peikert. Lattice cryptography for the internet. In Mosca [39], pages 197–219.

[41] T. Pöppelmann and T. Güneysu. Towards practical lattice-based public-key encryption on reconfigurable hardware. In T. Lange, K. E. Lauter, and P. Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *LNCS*, pages 68–85. Springer, 2013.

[42] T. Pöppelmann, T. Oder, and T. Güneysu. High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers. In K. E. Lauter and F. Rodríguez-Henríquez, editors, *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, volume 9230 of *LNCS*, pages 346–365. Springer, 2015.

[43] C. Pöpper, O. Mischke, and T. Güneysu. MicroACP - A fast and secure reconfigurable asymmetric crypto-processor - -overhead evaluation of side-channel countermeasures-. In D. Goehringer, M. D. Santambrogio, J. M. P. Cardoso, and K. Bertels, editors, *Reconfigurable Computing: Architectures, Tools, and Applications - 10th International Symposium, ARC 2014, Vilamoura, Portugal, April 14-16, 2014. Proceedings*, volume 8405 of *LNCS*, pages 240–247. Springer, 2014.

[44] E. Prouff and P. Schaumont, editors. *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *LNCS*. Springer, 2012.

[45] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, Sept. 2009.

[46] S. S. Roy, F. Vercauteren, N. Mentens, D. D. Chen, and I. Verbauwhede. Compact Ring-LWE cryptoprocessor. In L. Batina and M. Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *LNCS*, pages 371–391. Springer, 2014.

[47] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *LNCS*, pages 27–47. Springer, 2011.

[48] W. Stein et al. *Sage Mathematics Software (Version 6.0)*. The Sage Development Team, 2013. `http://www.sagemath.org`.

[49] D. Strobel, B. Driessen, T. Kasper, G. Leander, D. Oswald, F. Schellenberg, and C. Paar. Fuming acid and cryptanalysis: Handy tools for overcoming a digital locking and access control system. In Canetti and Garay [13], pages 147–164.