Cryptanalysis of Chosen Symmetric Homomorphic Schemes

Damian Vizár and Serge Vaudenay

EPFL
CH-1015 Lausanne, Switzerland
http://lasec.epfl.ch

Abstract. Since Gentry's breakthrough result was introduced in the year 2009, the homomorphic encryption has become a very popular topic. The main contribution of Gentry's thesis [9] was, that it has proven, that it actually is possible to design a fully homomorphic encryption scheme. However ground-breaking Gentry's result was, the designs, that employ the *bootstrapping* technique suffer from terrible performance both in key generation and homomorphic evaluation of circuits. Some authors tried to design schemes, that could evaluate homomorphic circuits of arbitrarily many inputs without need of bootstrapping. This paper introduces notion of symmetric homomorphic encryption, analyses the security of four such proposals, published in three different papers ([5], [12], [15]). Our result is a known plaintext key-recovery attack on every one of these schemes.

Keywords: Homomorphic Encryption, Symmetric encryption, Cryptanalysis, Key-recovery

1 Introduction

In 1978, the notion of *homomorphic encryption* has been formally introduced for the first time by Rivest et al. in [13].[1] Homomorphic encryption is a useful tool, for instance if there are some data, that need to be kept confidential, but have to be involved in a computation carried out in a (potentially) insecure environment. A homomorphic encryption function preserves effects of an operation over plaintexts - e.g. an addition of two ciphertexts would decrypt to the result of an addition of corresponding plaintexts.

Since Gentry's breakthrough result in 2009, homomorphic encryption has regained its popularity as a research topic. Currently, there are numerous proposals that followed Gentry's strategy and employed the bootstrapping technique, e.g. [7], [2], [14]. All of these schemes are reasonably secure, but suffer from terrible performances [11].There have also been several proposals ([5], [12], [15]), that use a construction, which is completely different from Gentry's.Their design does not require bootstrapping. It is however based on linear transformations, what puts their security in question. Our goal in this paper is to analyse security of these proposals and show, that they can be easily broken and are not secure.

Our result is a known plaintext key-recovery attack on each of the discussed schemes. Throughout the paper, we show that the construction of these schemes makes them not only efficient, but also vulnerable.

2 Symmetric homomorphic encryption scheme

In this section, the properties of a symmetric homomorphic encryption scheme are deduced and a definition is stated.

To define an encryption scheme ε, which is homomorphic in respect to some operation $\circ$ over plaintexts, we will use notion of *permitted circuits* $\mathcal{C}_\varepsilon$ of ε (this notion was introduced by Gentry in [9]). We can represent every formula over plaintexts as a circuit with $\circ$-gates. Then for such circuit $\mathsf{c}_\circ$, we can define equivalent circuit $\mathsf{c}_{\mathsf{Compose}}$ over ciphertexts by replacing $\circ$-gates by Compose-gates.

Definition 1. *We will denote by $\mathsf{c}_\circ$ an arbitrary circuit over ℓ inputs $m_i \in R$, $1 \leq i \leq \ell$, that consists of $\circ$-gates. Given a $c_\circ$, the notation $\mathsf{c}_{\mathsf{Compose}}$ stands for a circuit over ℓ elements of T, with the same structure as $\mathsf{c}_\circ$ with $\circ$-gates replaced by* Compose*-gates.*

Similarly $\mathsf{c}_{+,\times}$ represents a circuit over ℓ inputs $m_i \in R$, $1 \leq i \leq \ell$ that consists of $+$ and $\times$-gates and $\mathsf{c}_{Add,Mult}$ represents corresponding circuit with replaced gates.

[1] The original name used by Rivest was *privacy homomorphism*.

Since this document deals with symmetric homomorphic encryption, the secret key is used both for encryption and decryption. We may however need to make some parameters public, so that it is possible to carry out the homomorphic operations with the ciphertexts. These will be called pk in our definition. At the same time, the scheme must be secure. To ensure this, we are using the notion of indistinguishability.

Definition 2. *We say that a tuple* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Compose})$ *of probabilistic polynomial-time algorithms is a* secure homomorphic symmetric encryption scheme *over a monoid* $(R, \circ)$ *if:*

1. (homomorphic) *The set of permitted circuits* C_ε *contains every possible circuit* $c_\circ$*. For every circuit* $\mathsf{c}_\circ \in \mathcal{C}_\varepsilon$ *with* ℓ *inputs, any* $m_i \in R$, $1 \leq i \leq \ell$ *and for every possible* $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *we have*

$$\Pr\left[\mathsf{Dec}_{sk}\left(\mathsf{c}_{\mathsf{Compose}}\left(\mathsf{Enc}_{sk}(m_1), \ldots, \mathsf{Enc}_{sk}(m_\ell)\right)\right) = \mathsf{c}_\circ\left(m_1 \ldots, m_\ell\right)\right] \geq 1 - \epsilon$$

   for some ϵ *negligible in security parameter.*
2. (secure) *For every circuit* $\mathsf{c}_\circ \in \mathcal{C}_\varepsilon$ *with* ℓ *inputs, any* $m' \in R$, $m_i \in R$, $1 \leq i \leq \ell$ *and for every possible* $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *the distributions*

$$\left(\mathsf{pk}, \mathsf{c}_{\mathsf{Compose}}\left(\mathsf{Enc}_{sk}(m_1), \ldots, \mathsf{Enc}_{sk}(m_\ell)\right)\right), \left(\mathsf{pk}, \mathsf{Enc}_{sk}(m')\right)$$

   are computationally indistinguishable.[2]

The name of algorithm $\mathsf{Compose}$ can be replaced by Add or Mult, if the considered monoid uses additive or multiplicative notation. We say, that a tuple $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Add}, \mathsf{Mult})$ of probabilistic polynomial-time algorithms is a *fully-homomorphic symmetric encryption scheme* over ring $(R, +, \times)$ if it satisfies definition 2 with a slight modification - circuits $c_{+,\times}$ are used instead of $c_\circ$ and C_ε contains every possible circuit consisting of $+$ and $\times$ gates.

3 The Chan's homomorphic encryption schemes

This section discusses two encryption schemes presented in [5] by Chan. Both schemes are only homomorphic over addition of plaintexts, but both schemes also allow multiplication of ciphertexts by an unencrypted (scalar) value. Both schemes have a simple and clean design, which allows evaluation of circuits of arbitrary depth, but makes it easy to break them. Both of the schemes are non-deterministic.

3.1 Iterated Hill Cipher

The scheme called *Iterated Hill Cipher* is a generalization of the Hill cipher. Thanks to its design, the key space of this scheme is increased to all $\ell \times \ell$ matrices over a ring $\mathbb{Z}_n$ (compared to original Hill cipher, that could only use invertible matrices). The scheme has two parameters - $n, \ell \in \mathbb{N}$. Plaintexts live in $\mathbb{Z}_n^\ell$. The scheme is defined as follows:

$\mathsf{KeyGen}()$**:** Select a secret matrix $\mathbf{A} \xleftarrow{R} \mathbb{Z}_n^{\ell \times \ell}$ and a secret integer $k \xleftarrow{R} \mathbb{N}$. The matrix $\mathbf{A}$ does not need to be invertible. Output $(pk, sk) = (n, (\mathbf{A}, k))$.

$\mathsf{Enc}_{\mathsf{sk}}(\mathbf{x})$**:** Choose a random vector $\mathbf{u} \xleftarrow{R} \mathbb{Z}_n^\ell$, set $\mathbf{x}_{-1} = \mathbf{u}$, $\mathbf{x}_0 = \mathbf{x}$. Compute $\mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i - \mathbf{x}_{i-1}$ for $0 \leq i < k$. Output $(\mathbf{x}_k, \mathbf{x}_{k-1})$.

$\mathsf{Dec}_{\mathsf{sk}}(\mathbf{c}_1, \mathbf{c}_0)$**:** Set $\mathbf{x}_k = \mathbf{c}_1$, $\mathbf{x}_{k-1} = \mathbf{c}_0$. Compute $\mathbf{x}_{i-1} = \mathbf{A}\mathbf{x}_i - \mathbf{x}_{i+1}$ for $0 \leq i < k$. Output $(\mathbf{x}_0)$.

$\mathsf{Add}_{\mathsf{pk}}(\mathsf{c}, \mathsf{c}')$**:** Output $\mathsf{c}^* = (\mathbf{c}_1 + \mathbf{c}'_1, \mathbf{c}_0 + \mathbf{c}'_0)$.

Homomorphic properties: As can be seen, this scheme does not provide the Mult algorithm. It allows just "*scalar multiplication*" (i.e. multiplication by unencrypted value from $\mathbf{Z}_n$), that is performed as multiplication (modulo n) of all ciphertext components by this value. Circuits of arbitrary depth can be evaluated over ciphertexts, and the result is always decryptable.

[2] Note, that the two properties imply semantic security, if we consider a circuit with no gates.

Remark on encryption: The encryption and decryption can be expressed as linear transformation using bigger matrices (from $\mathbb{Z}_n^{2\ell\times 2\ell}$). We have

$$\mathsf{Enc}(\mathbf{x}) = \begin{pmatrix}\mathbf{A} & -\mathbf{1}\\ \mathbf{1} & \mathbf{0}\end{pmatrix}^k \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix} = \begin{pmatrix}\mathbf{x}_k\\ \mathbf{x}_{k-1}\end{pmatrix}$$

$$\begin{aligned}\mathsf{Dec}(\mathbf{x_k}, \mathbf{x_{k-1}}) &= \left(\begin{pmatrix}\mathbf{0} & \mathbf{1}\\ -\mathbf{1} & \mathbf{A}\end{pmatrix}^k \begin{pmatrix}\mathbf{x}_k\\ \mathbf{x}_{k-1}\end{pmatrix}\right)_{1,1} = \left(\begin{pmatrix}\mathbf{0} & \mathbf{1}\\ -\mathbf{1} & \mathbf{A}\end{pmatrix}^k \begin{pmatrix}\mathbf{A} & -\mathbf{1}\\ \mathbf{1} & \mathbf{0}\end{pmatrix}^k \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix}\right)_{1,1}\\ &= \begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix}_{1,1}\end{aligned}$$

Security: The author claims, that if a new random vector $\mathbf{u}$ is chosen randomly for every encryption, this scheme is secure against known plaintext attacks. This claim does not hold, as we will show how to break the scheme, introducing a known-plaintext key-recovery attack, which recovers a decryption key. The data complexity of the attack is $O(\ell)$ PT-CT pairs and the time complexity is $O(\ell^3 \log n)$ arithmetic operations.

This attack is possible, because encryption function of Iterated Hill possesses the linear structure shown above. Main idea behind the attack is to split entries of the secret key (a matrix) in two parts and deal with one part at a time. We first provide an algorithmic description of the attack in algorithm 3.1.

Algorithm 3.1 Key-recovery attack on Iterated Hill Cipher

Input: PT-CT pairs created with secret key sk, parameters ℓ, n.
Output: Matrix $\mathbf{S} \in \mathbb{Z}_n^{\ell\times 2\ell}$, $\forall\,(\mathbf{x}, \mathbf{c} = \mathsf{Enc}_{\mathsf{sk}}(\mathbf{x})) : \mathbf{x} = \mathbf{Sc}$.

Initialize $\mathbf{C}$ as an empty matrix.
while $\mathbf{C}$ does not have ℓ linearly independent columns **do**
    Find $\lambda_i \in \mathbb{Z}_n$ and $(\mathbf{x}_i, \mathbf{c}_i = \mathsf{Enc}_{\mathsf{sk}}(\mathbf{x_i}))$ with $\sum_{i=1}^m \lambda_i \mathbf{x}_i \equiv \mathbf{0} \pmod n$.
    Add column vector $\mathbf{c}' = \sum_{i=1}^m \lambda_i \mathbf{c}_i$ to $\mathbf{C}$.
end while
Solve system $\mathbf{HC} \equiv \mathbf{0}^{\ell\times\ell} \pmod n$, for $\mathbf{H} \in \mathbb{Z}_n^{\ell\times 2\ell}$.
if No solution $\mathbf{H}$ is found **then**
    Return Failure.
end if
Initialize $\mathbf{C}$ and $\mathbf{X}$ as empty matrices.
while $\mathbf{C}$ does not have ℓ linearly independent columns **do**
    Pick a PT-CT pair $(\mathbf{x}, \mathbf{c})$, compute $\mathbf{c}' = \mathbf{Hc}$.
    Add column vector $\mathbf{c}'$ to $\mathbf{C}$ and column vector $\mathbf{x}$ to $\mathbf{X}$.
end while
Solve system $\mathbf{MX} \equiv \mathbf{C} \pmod n$, for $\mathbf{M} \in \mathbb{Z}_n^{\ell\times\ell}$.
if No solution $\mathbf{M}$ is found **then**
    Return Failure.
end if
Compute matrix $\mathbf{S} = \mathbf{M}^{-1}\mathbf{H} \pmod n$.
Output $\mathbf{S}$.

Theorem 1. *By running the algorithm 3.1 at most $O(\log n)$ times, an attacker can compute matrix $\mathbf{S}$, which can be used to decrypt any ciphertext $\mathbf{c}$ by computing $\mathbf{Sc} = \mathbf{x}$.*

Complexity: The time complexity of the attack is dominated by complexity of solving two linear systems of size ℓ. This is bounded by $O(\ell^3)$ arithmetic operations. In case of a composite $n = \sum_{i=1}^k p_i^{r_i}$, the number of systems that need to be solved depends on factorization of n. The upper bound on number of systems that need to be solved can be calculated as $\sum_{i=1}^k r_i = O(\log n)$. So the total time complexity is bounded

$O(\ell^3 \log n)$ arithmetic operations.

The data complexity of the attack is determined by the number of PT-CT pairs needed in two phases of algorithm 3.1. In the first phase, where parity-check matrix $\mathbf{H}$ is computed, we need ℓ linearly independent encryptions of zero (as in equation A.3). Remember, that the encryptions of zero are obtained by finding linear relations $\sum_{i=1}^{m} \lambda_i \mathbf{x}_i \equiv \mathbf{0} \pmod{n}$ among plaintexts and applying coefficients λ_i to corresponding ciphertexts. If we arrange m plaintext vectors as columns of a matrix $\mathbf{X}$ and coefficients λ_i into vector λ, we can express finding these relations as equation

$$\mathbf{X}\lambda \equiv \mathbf{0} \pmod{n}$$

It is well known, that by using 2ℓ random samples in $\mathbb{Z}_\ell$ we obtain ℓ linearly independent ones with high probability. So, the nullspace of the plaintext collection $\mathbf{X}$ has dimension ℓ and we can find ℓ linearly independent relations of the form $\sum_{i=1}^{2\ell} \lambda_i \mathbf{x}_i$.

In the second phase, where matrix $\mathbf{M}$ is computed, encryptions of ℓ linearly independent plaintexts are needed. Fulfilling this requirement is implied if the requirement for first phase is satisfied. We can therefore deduce the data complexity of the attack to be $O(\ell)$ PT-CT pairs.

3.2 Modified Rivest

The scheme called *Modified Rivest* is inspired by one of four *privacy homomorphisms*, that were presented as an example by Rivest et al. in [13] from 1978. The design of Modified Rivest scheme tries to prevent attacks on original scheme in [13]. The scheme has one parameter $k \in \mathbb{N}$ and is defined as follows:

$\mathsf{KeyGen}()$**:** Set $p \xleftarrow{R} \mathbb{P}$, $q \xleftarrow{R} \mathbb{P}$. The value $n = pq$ is the public key (used in homomorphic operations). Choose numbers $\mathbf{r} = (r_1, \ldots, r_k) \xleftarrow{R} (\mathbb{Z}_p^*)^k$ and $\mathbf{s} = (s_1, \ldots, s_k) \xleftarrow{R} (\mathbb{Z}_q^*)^k$. Output $(pk, sk) = (n, (p, q, \mathbf{r}, \mathbf{s}))$.

$\mathsf{Enc}_{\mathsf{sk}}(\mathsf{x})$**:** Choose a vector $(x_1, \ldots, x_k)$, with $\sum_{i=1}^{k} x_i = x \bmod n$. For $1 \le i \le k$ compute the pairs

$$(c_i, c_i') = (r_i x_i \bmod p,\ s_i x_i \bmod q)$$

Output $\mathbf{c} = ((c_1, c_1'), \ldots, (c_k, c_k'))$.

$\mathsf{Dec}_{\mathsf{sk}}(\mathbf{c})$**:** Using CRT compute $x_i \bmod n$, $1 \le i \le k$ with $x_i \equiv c_i r_i^{-1} \pmod{p}$ and $x_i \equiv c_i' s_i^{-1} \pmod{q}$. Output $\sum_{i=1}^{k} x_i \bmod n$.

$\mathsf{Add}_{\mathsf{pk}}(\mathbf{c}^{(1)}, \mathbf{c}^{(2)})$**:** For $0 \le i \le k$ compute

$$\left(c_i^*, c_i^{*\prime}\right) = \left(c_i^{(1)} + c_i^{(2)} \bmod n,\ c_i^{(1)\prime} + c_i^{(2)\prime} \bmod n\right).$$

Output $\mathbf{c}^*$.

Homomorphic properties: The scheme does not provide the algorithm $\mathsf{Mult}_{\mathsf{pk}}$, but a "*scalar multiplication*" (i.e. multiplication by unencrypted value from $\mathbf{Z}_n$) can be performed as multiplication of all ciphertext components by this value. Again, circuits of arbitrary depth can be evaluated without loss of ability to decrypt correctly.

Security: We will show how to break the scheme, introducing a known-plaintext, key-recovery attack with the data complexity $O(k)$ PT-CT pairs and the time complexity $O(k^3)$ arithmetic operations.

The attack is based on the simple observation, that the decryption can be expressed as a linear equation modulo n. The algorithmic description of the attack is presented in algorithm 3.2.
The Following analysis shows that the output of algorithm 3.2 can be used to decrypt messages encrypted under the same key as known PT-CT pairs. We express the idea of the attack in lemma 2 and provide a constructive proof of the lemma, which motivates the algorithm 3.2.

Algorithm 3.2 Key-recovery attack on Modified Rivest

Input: PT-CT pairs created with secret key sk.
Output: Vector $\mathbf{t} = (t_1^a, t_1^b, \ldots, t_k^a, t_k^b)^T \in \mathbb{Z}_n^{2k}$, $\forall\,(x, \mathbf{c} = \mathsf{Enc}_{\mathsf{sk}}(x)) : x = \sum_{i=1}^{k} t_i^a c_i + t_i^b c_i' \bmod n$

```
Set C as an empty matrix and x as empty column vector.
while C does not have 2k linearly independent columns do
    if rows of C are linearly dependent then
        Drop last row in C and last element in x.
    end if
    Select an encryption c = ((c_1, c_1'), ..., (c_k, c_k')) = Enc_sk(x).
    Append c to C as last column and x to x as last element.
end while
Solve system Ct = x (mod n), for t ∈ Z_n^{2k}.
if System not solved because an a, gcd(a, n) > 1 was encountered then
    Output factors gcd(a, n), n / gcd(a, n)
end if
Output t.
```

Lemma 2 *Given an instance* $(\mathsf{pk}, \mathsf{sk})$ *of the Modified Rivest scheme, there exist* $a, b \in \mathbb{Z}_n$ *and* $t_1, \ldots, t_k \in \mathbb{Z}_n$, *such that for all possible* $x, ((c_1, c_1'), \ldots, (c_k, c_k')) = \mathsf{Enc}_{\mathsf{sk}}(x)$

$$x = \sum_{i=1}^{k} t_i(ac_i + bc_i') \bmod n.$$

The proof of lemma 2 can be found in appendix. Note, that $t_i(ac_i + bc_i') = t_i ac_i + t_i bc_i'$ and that the terms at_i, bt_i are constant in every encryption under the same key. Let $t_i^{(a)} = at_i \bmod n$ and $t_i^{(b)} = bt_i \bmod n$ denote the unknowns in equation A.12. With $O(2k)$ PT-CT pairs, we can create a determined linear system, that can be solved for $t_i^{(a)}, t_i^{(b)}$. This is exactly what algorithm 3.2 does. Once these numbers are known, the attacker can decrypt any message using equation A.12. Time complexity of algorithm 3.2 is dominated by complexity of solving a linear system of size $2k$ which is $O(k^3)$ arithmetic operations.

Apart from susceptibility of the scheme to presented attack, it is also worth noting, that the distribution of ciphertexts created by encryption algorithm differs from distribution of ciphertexts created by homomorphic addition. In a freshly encrypted ciphertext $\mathbf{c} = ((c_1, c_1'), \ldots, (c_k, c_k'))$, the probability $\Pr[c_i > p] \approx \Pr[c_i > \sqrt{n}] \approx 0$. A similar relation applies to elements c_i'. In a ciphertext created by homomorphic addition however, this probability would be $\Pr[c_i > p] \approx \left(1 - \frac{1}{\sqrt{n}}\right)$. This means that the Modified Rivest scheme does not have the *secure* property of definition 2.

4 Homomorphic schemes based on linear transformations

This section discusses two schemes, with very similar design, based on matrix multiplication. The design of the discussed schemes is very simple and clean, and they have very nice homomorphic properties - they are homomorphic in two operations, without any need for bootstrapping technique. They claim to be fully homomorphic. As will be shown, however, the way that these schemes achieve the homomorphic properties is also exploitable in simple attacks.

4.1 Simple linear homomorphic encryption

The following scheme called MORE (Matrix Operation for Randomized Encryption) is presented in [12]. It is defined as follows:

$\mathsf{KeyGen}()$**:** Choose $p \xleftarrow{R} \mathbb{P}$, $q \xleftarrow{R} \mathbb{P}$, set $n = pq$. Choose secret key $\mathbf{S} \xleftarrow{R} \{A \in \mathbb{Z}_n^{2\times 2} | \det(A) \in \mathbb{Z}_n^*\}$. Output $(pk, sk) = (n, \mathbf{S})$.

$\mathsf{Enc}_{\mathsf{sk}}(\mathsf{x})$**:** Choose $r \xleftarrow{R} \mathbb{Z}_n$. Output $\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} = \mathbf{S} \begin{pmatrix} x & 0 \\ 0 & r \end{pmatrix} \mathbf{S}^{-1}$.

$\mathsf{Dec}_{\mathsf{sk}}(\mathbf{C})$**:** Output $\left(\mathbf{S}^{-1}\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}\mathbf{S}\right)_{11}$.
$\mathsf{Add}_{\mathsf{pk}}(\mathbf{C}_1, \mathbf{C}_2)$**:** Output $\mathbf{C} = \mathbf{C}_1 + \mathbf{C}_2 \bmod n$.
$\mathsf{Mult}_{\mathsf{pk}}(\mathbf{C}_1, \mathbf{C}_2)$**:** Output $\mathbf{C} = \mathbf{C}_1\mathbf{C}_2 \bmod n$.

Homomorphic properties: This scheme is homomorphic in two operations, its homomorphic properties follow from the definition of encryption, which is a linear transformation in nature - they are performed as standard matrix multiplication and addition of ciphertexts. There is no need for bootstrapping.

Security: The authors prove the security of the scheme based on the following assumptions:

- just ciphertext-only attacks are allowed,
- plaintexts are random and independent.

The assumptions made by the authors of [12] are very strong and do no correspond to real-world applications. It can be deduced, that the MORE scheme does not meet the *secure* property of definition 2. Recall, that the *secure* property of the definition 2 requires, that the output of encryption function is indistinguishable from the output of homomorphic evaluation of circuits for any circuit. This implies semantic security because a circuit with no gates and a single input is in fact direct encryption. So, for a homomorphic encryption scheme ε to be secure, we must have, that for any two plaintexts p_1, p_2 the distributions of $\mathsf{Enc}_{\mathsf{sk}}(p_1)$ and $\mathsf{Enc}_{\mathsf{sk}}(p_2)$ must necessarily be indistinguishable.

The MORE scheme does not meet this requirement. As a counterexample, consider two plaintexts $x_1, x_2 \in \mathbb{Z}_n$ with $x_1 \neq x_2$. We have

$$\Pr[\mathsf{Enc}_{\mathsf{sk}}(x_1) \neq \mathsf{Enc}_{\mathsf{sk}}(x_2)] = 1,$$

because the two ciphertexts must necessarily have characteristic polynomials with different roots. The two distributions are then easily distinguishable - a plaintext is always a root of the characteristic polynomial of its ciphertext.

We will now present a key recovery attack on the MORE scheme, that requires only a side channel information on plaintext, not the knowledge of the plaintexts themselves. More precisely, algorithm 4.1 only requires known polynomial relation between some plaintexts, rather than the plaintext themselves - it uses ciphertexts $\mathbf{C}_1 = \mathsf{Enc}_{\mathsf{sk}}(x_1), \ldots, \mathbf{C}_\ell = \mathsf{Enc}_{\mathsf{sk}}(x_\ell)$ with known $\lambda_1, \ldots, \lambda_\ell \in \mathbb{Z}_n$ and $e_1, \ldots, e_\ell \in \mathbb{Z}_n$ such that

$$\sum_{i=1}^{\ell} \lambda_i x_i^{e_i} \equiv 0 \pmod{n}.$$

Algorithm 4.1 Related plaintext key-recovery attack on MORE

Input: Ciphertexts $\mathbf{C}_1 = \mathsf{Enc}_{\mathsf{sk}}(x_1), \ldots, \mathbf{C}_\ell = \mathsf{Enc}_{\mathsf{sk}}(x_\ell)$ and $\lambda_1, \ldots, \lambda_\ell, e_1, \ldots, e_\ell \in \mathbb{Z}_n$ with $\sum_{i=1}^{\ell} \lambda_i x_i^{e_i} \equiv 0 \pmod{n}$
Output: Eigenvector $\mathbf{v}$ of $\mathbf{C}$, $\forall\,(x', \mathbf{C}' = \mathsf{Enc}_{\mathsf{sk}}(x')) : \mathbf{C}'\mathbf{v} \equiv x'\mathbf{v} \pmod{n}$.
1: Compute $\mathbf{C}^* = \sum_{i=1}^{\ell} \lambda_i \mathbf{C}_i^{e_i} \bmod n$.
2: Find a non-zero solution $\mathbf{v}$ of $\mathbf{C}^*\mathbf{v} \equiv \mathbf{0} \pmod{n}$.
3: **if** System not solved because an $a, \gcd(a, n) > 1$ was encountered **then**
4: Output factors $\gcd(a, n)$, $\frac{n}{\gcd(a,n)}$
5: **end if**
6: Output $\mathbf{v}$.

The ciphertext $\mathbf{C}^* == \sum_{i=1}^{\ell} \lambda_i \mathbf{C}_i^{e_i} \bmod n$ computed in algorithm 4.1 is in fact an encryption of zero. More precisely, we have

$$\mathbf{C}^* \equiv \mathbf{S}\begin{pmatrix} 0 & 0 \\ 0 & y^* \end{pmatrix}\mathbf{S}^{-1} \pmod{n}$$

for some $y^* \in \mathbb{Z}_n$. A non-zero solution of equation $\mathbf{C}^*\mathbf{v} \equiv \mathbf{0} \pmod{n}$ is a vector $\mathbf{v}$ of the form $\mathbf{S}(t,0)^T$, $t \in \mathbb{Z}_n$, which is an eigenvector of arbitrary ciphertext, associated with its plaintext. This can be used to decrypt any ciphertext $\mathbf{C}$ by solving the equation $\mathbf{C}\mathbf{v} = x\mathbf{v}$ for x. The time complexity of the attack is again $O(1)$ arithmetic operations. The data complexity is given by the number of ciphertexts that satisfy the relation obtained by attacker. Relation formed by as little as two ciphertext is already usable.

Ciphertext only decryption attack for small plaintexts:A weak spot in the ciphertext only scenario are encryptions of plaintexts smaller than $\sqrt{n}$. Due to the theorem of Coppersmith in [6], it is possible to find plaintext x, which is a root of ciphertext's $\mathbf{C}$ characteristic polynomial, in time polynomial in $\log(n)$ if we have

$$|x| \leq \frac{1}{2} n^{O(\frac{1}{2})}.$$

This attack works, if the small plaintexts occur with high probability.

4.2 Improved linear homomorphic encryption

Scheme presented in [15] is in fact an improvement of the MORE scheme presented in previous section. The key-recovery attack described by algorithm 4.1 is not applicable to this scheme - a plaintext is still an eigenvalue of corresponding ciphertext, but eigenspace associated to the plaintext has now higher dimensionality and its basis is randomised. The scheme has one parameter $k \in \mathbb{N}$ and is defined as follows:

$\mathsf{KeyGen}()$**:** Choose $p_i \xleftarrow{R} \mathbb{P}$, $q_i \xleftarrow{R} \mathbb{P}$ for $1 \leq i \leq k$. Let $\mathbf{n} = (p_1q_1, \ldots, p_kq_k)$ and $n = \prod_{i=1}^{k} p_iq_i$. Choose secret matrix $\mathbf{K} \xleftarrow{R} \{A \in \mathbb{Z}_n^{4\times 4} | \det(A) \in \mathbb{Z}_n^*\}$.
Output $(pk, sk) = (n, (\mathbf{n}, \mathbf{K}))$.

$\mathsf{Enc_{sk}(x)}$**:** Choose $r \xleftarrow{R} \mathbb{Z}_n$. For $1 \leq i \leq k$ set values a_i, b_i, c_i according to following distribution:

$$\Pr\begin{bmatrix} a_i = x \\ b_i = r \\ c_i = r \end{bmatrix} = 1 - \frac{1}{k+1}, \Pr\begin{bmatrix} a_i = r \\ b_i = x \\ c_i = r \end{bmatrix} = \frac{1}{2(k+1)}, \Pr\begin{bmatrix} a_i = r \\ b_i = r \\ c_i = x \end{bmatrix} = \frac{1}{2(k+1)}$$

Using CRT compute $a, b, c \bmod n$, for which following holds:

$$a \equiv a_i \pmod{p_iq_i},\ b \equiv b_i \pmod{p_iq_i},\ c \equiv c_i \pmod{p_iq_i}.$$

Output $\mathbf{C} = \mathbf{K^{-1}} \begin{pmatrix} x & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & c \end{pmatrix} \mathbf{K}$.

$\mathsf{Dec_{sk}}(\mathbf{C})$**:** Output $\left(\mathbf{KCK}^{-1}\right)_{11}$.
$\mathsf{Add_{pk}}(\mathbf{C_1}, \mathbf{C_2})$**:** Output $\mathbf{C} = \mathbf{C}_1 + \mathbf{C}_2 \bmod n$.
$\mathsf{Mult_{pk}}(\mathbf{C_1}, \mathbf{C_2})$**:** Output $\mathbf{C} = \mathbf{C}_1\mathbf{C}_2 \bmod n$.

Homomorphic properties: This scheme is homomorphic in two operations. Similarly as in previous paper, *homomorphic addition* and *homomorphic multiplication* are performed as standard matrix addition and multiplication mod n, without need of bootstrapping.

Security: The scheme presented in [15] can be broken similarly as the one in [12]. A known plaintext key-recovery attack is presented in algorithm 4.2. The data complexity of the attack can be bounded by $O(1)$ PT-CT pairs and the time complexity of the attack can be bounded by $O(1)$ arithmetic operations.

We will now show, that output of algorithm 4.2 can be used to decrypt any ciphertext encrypted under the same key. The scheme of Xiao et al. [15] can be seen as an improvement of MORE scheme. The scheme was designed, to prevent a simple known-plaintext attack. This is done by increasing dimension of ciphertexts and associating plaintext x with more than just one eigenvector of ciphertext matrix. An attack of similar

Algorithm 4.2 Key-recovery attack on scheme of Xiao et al.

Input: Known PT-CT pairs created with secret key sk.
Output: Eigenvector $\mathbf{v}$, $\forall\,(x', \mathbf{C}' = \mathsf{Enc}_{\mathsf{sk}}(x')) : \mathbf{C}'\mathbf{v} \equiv x'\mathbf{v} \pmod{n}$

```
Pick a PT-CT (x, C) pair and remove it from collection
Solve (C − xI)v' ≡ 0 (mod n), for V = {v'|v' ∈ Z_n^2 \ {0}}.
while dim(V) > 1 do
    Pick a PT-CT (x, C) pair and remove it from collection
    Solve (C − xI)v' ≡ 0 (mod n), for V* = {v'|v' ∈ Z_n^2 \ {0}}.
    if System not solved because an a, gcd(a, n) > 1 was encountered then
        Output factors gcd(a, n), n/gcd(a, n)
    end if
    Set V = V ∩ V*.
end while
Output any v ∈ V*.
```

complexity as the attack on MORE scheme can however be mounted.

We can establish an equation to find eigenvectors associated with plaintext x as follows:

$$(\mathsf{Enc}(\mathsf{x}) - x\mathbf{I})\mathbf{v} = 0$$

If we solve this system for $\mathbf{v}$, we get a set of solutions of the form:

$$\mathcal{V} = \left\{\mathbf{v} = a\mathbf{K}^{-1}(1,0,0,0)^T + b\mathbf{K}{-}1\mathbf{v}^* \middle| a, b \in \mathbb{Z}_n\right\}, \tag{4.1}$$

where $\mathbf{v}^* \in \mathbb{Z}_n^4$ is a random vector with the following distribution:

$$\Pr[\mathbf{v}^* \equiv (0,1,0,0)^T \pmod{p_iq_i}] = 1{-}\frac{1}{k+1},\ \Pr[\mathbf{v}^* \equiv (0,0,1,0)^T \pmod{p_iq_i}] = \Pr[\mathbf{v}^* \equiv (0,0,0,1)^T \pmod{p_iq_i}] = \frac{1}{2(k+1)}$$

If we compute these equations for two PT-CT pairs, we will obtain two eigenspaces $\mathcal{V}_1, \mathcal{V}_2$. Both $\mathcal{V}_1$ and $\mathcal{V}_2$ will contain the eigenline $\{\mathbf{v} \in \mathbb{Z}_n^4 | \mathbf{v} = a\mathbf{K}^{-1}(1,0,0,0)^T, a \in \mathbb{Z}_n\}$, so we have

$$\left\{\mathbf{v} = a\mathbf{K}^{-1}(1,0,0,0)^T \middle| a \in \mathbb{Z}_n\right\} \subset \mathcal{V}_1 \cap \mathcal{V}_2$$

To decrypt an arbitrary ciphertext, we need to identify the set

$$\mathcal{V} = \left\{\mathbf{v} = a\mathbf{K}^{-1}(1,0,0,0)^T \middle| a \in \mathbb{Z}_n\right\}.$$

We this can be obtained as

$$\left\{\mathbf{v} = a\mathbf{K}^{-1}(1,0,0,0)^T \middle| a \in \mathbb{Z}_n\right\} = \mathcal{V}_1 \cap \mathcal{V}_2 \tag{4.2}$$

The event, that equation 4.2 does not hold, can be shown to occur with probability close to e^{-2}. Therefore after $O(1)$ iterations of the loop in algorithm 4.2, we will find desired subspace $\mathcal{V}$ and therefore a vector $\mathbf{v}$ of the form $\mathbf{v} = \lambda\mathbf{K}^{-1}(1,0,0,0)^T,\ \lambda \in \mathbb{Z}_n$. Such a vector can be used to decrypt any ciphertext $\mathbf{C}' = \mathsf{Enc}_{\mathsf{sk}}(x')$ by solving the system $(\mathbf{C}' - x'\mathbf{I})\mathbf{v} = 0$ for x'. Time complexity of the attack is dominated by solving $O(1)$ linear systems of rank 4 which is $O(1)$ arithmetic operations.

5 Conclusion

In this paper, security of four symmetric homomorphic encryption schemes is discussed. The goal of our work was to show, that these schemes are not secure. As we have shown, there is a key-recovery attack on each of the schemes, more precisely:

– the scheme Iterated Hill Cipher presented by Chan in [5] can be broken with $O(\ell)$ PT-CT pairs in time bounded by $O(\ell^3 \log^2(n))$ arithmetic operations,

– the scheme Modified Rivest presented by Chan in [5] can be broken with $O(k)$ PT-CT pairs in time bounded by $O(k^3)$ arithmetic operations,
– the scheme MORE presented in [12] and the presented in [15] can be both broken with $O(1)$ PT-CT pairs in time bounded by $O(1)$ arithmetic operations.

All of the discussed schemes can be seen as an attempt, to construct a (fully) homomorphic scheme, that allows homomorphic evaluation of circuits of arbitrary depth. Moreover, in these cases the ability to evaluate arbitrary circuits is intrinsic - we do not transform a somewhat homomorphic scheme using bootstrapping. This is a very desirable property - thanks to it, the computational overhead introduced by bootstrapping could be avoided. Unfortunately, the construction principles of the discussed schemes have a common feature, and this feature is also the reason of their insecurity. Following from the results of this paper, there are two lessons to be learned.

Firstly, it is not advisable to use linear transformations as the only component of encryption algorithm. As all the analysed schemes demonstrate, the linear transformations are very attractive, when it comes to homomorphic properties. You can evaluate circuits (or formulas) of arbitrary depth efficiently. In case of linear transformations, this property is intrinsic, so no bootstrapping is needed. However, as this paper demonstrates, linear transformations are susceptible to efficient known plaintext attacks, even if they are randomized.

Secondly, it is not advisable, not to base the security of an encryption scheme on a generally accepted security assumption. The presented schemes claim their security, but they are susceptible to simple and efficient attacks. It is because they either do not base the security on any generally accepted security assumption or the the assumptions that the security proofs rely on are unrealistic.

The result of this paper shows, that the discussed schemes are not secure. This does not necessarily mean, that it is not possible to design an intrinsically (fully) homomorphic scheme, that is also secure. This paper points out however, that the construction principles used in discussed schemes may not be appropriate to achieve this goal, at least not in their current form. So, an open question, that remains unanswered is, if there is any way, to design a (symmetric) homomorphic encryption based on a very simple construction principle (e.g. linear transformation) with some additional measures, that would be secure.

References

1. Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in lwe-based homomorphic encryption. In *Public Key Cryptography*, pages 1–13, 2013.
2. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Proceedings of the 31st annual conference on Advances in cryptology*, CRYPTO'11, pages 505–524, Berlin, Heidelberg, 2011. Springer-Verlag.
3. Ernest F. Brickell and Yacov Yacobi. On privacy homomorphisms. In *EUROCRYPT*, volume 304 of *Lecture Notes in Computer Science*, pages 117–125, 1988.
4. Claude Castelluccia, Einar Mykletun, and Gene Tsudik. E.cient aggregation of encrypted data in wireless sensor networks. In *Proceedings of the The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, MOBIQUITOUS '05, pages 109–117, Washington, DC, USA, 2005. IEEE Computer Society.
5. Aldar C-F. Chan. Symmetric-key homomorphic encryption for encrypted data processing. In *Proceedings of the 2009 IEEE international conference on Communications*, ICC'09, pages 774–778, Piscataway, NJ, USA, 2009. IEEE Press.
6. Don Coppersmith. Small solutions to polynomial equations, and low exponent rsa vulnerabilities. *J. Cryptology*, 10(4):233–260, 1997.
7. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
8. Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP J. Inf. Secur.*, 2007:15:1–15:15, January 2007.
9. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford, CA, USA, 2009. AAI3382729.
10. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.

11. Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520, 2010. `http://eprint.iacr.org/`.
12. Aviad Kipnis and Eliphaz Hibshoosh. Efficient methods for practical fully homomorphic symmetric-key encrypton, randomization and verification. *IACR Cryptology ePrint Archive*, 2012:637, 2012.
13. R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.
14. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010.
15. Liangliang Xiao, Osbert Bastani, and I-Ling Yen. An efficient homomorphic encryption protocol for multi-user systems. *IACR Cryptology ePrint Archive*, 2012:193, 2012. informal publication.

A Proofs

A.1 Proof of theorem 1

We will show, that the output of algorithm 3.1 can be indeed used to obtain a decryption key. We will first deal with the case, that n is prime and then cover more general cases.

Case n is prime

If $n \in \mathbb{P}$ then plaintexts and ciphertexts are vectors from proper vector spaces over prime field $\mathbb{Z}_n$.
First, recall that the encryption algorithm can be written as a matrix-vector product in higher dimension. Let $\mathbf{B} \in \mathbb{Z}_n^{2\ell \times 2\ell}$ denote the secret matrix, which is equivalent to secret key $(\mathbf{A}, k)$:

$$\mathbf{B} = \begin{pmatrix} \mathbf{A} & -\mathbf{1} \\ \mathbf{1} & \mathbf{0} \end{pmatrix}^k.$$

The matrix $\mathbf{B}$ always has an inverse (otherwise we could not decrypt), and its rank is therefore $\operatorname{rank}(\mathbf{B}) = 2\ell$. Further on, the encryption is in fact choice of a vector from vector space spanned by $\mathbf{B}$, as a linear combination of columns of $\mathbf{B}$ given by $\mathbf{B}(\mathbf{x}^T\, \mathbf{u}^T)^T$. Notice, that if we split the secret matrix $\mathbf{B}$ into two matrices $\mathbf{B}^{(1)}, \mathbf{B}^{(2)} \in \mathbb{Z}_n^{2\ell \times \ell}$ with $\mathbf{B} = \left(\mathbf{B}^{(1)}\, \mathbf{B}^{(2)}\right)$, we can write the encryption down as

$$\mathsf{Enc}_{\mathsf{sk}}(\mathbf{x}) = \mathbf{B}^{(1)}\mathbf{x} + \mathbf{B}^{(2)}\mathbf{u}. \tag{A.1}$$

It follows, that the encryptions of zero vector, i.e. $\{\mathbf{c} = \mathsf{Enc}_{\mathsf{sk}}(\mathbf{0})\}$, comprise a subspace of the vector space $\langle \mathbf{B} \rangle$:

$$\left\langle \mathbf{B}^{(2)} \right\rangle = \left\{ \mathbf{c} \middle| \mathbf{c} = \mathbf{B} \begin{pmatrix} \mathbf{0} \\ \mathbf{u} \end{pmatrix} = \mathbf{B}^{(2)}(\mathbf{u}) \right\} \tag{A.2}$$

The linear subspace of $\mathbb{Z}_n^{2\ell}$ given by equation A.2 is in fact a linear $[2\ell, \ell]$ code over $\mathbb{Z}_n$ - a consequence of the fact, that the secret matrix $\mathbf{B}$ must have full rank (so a subset of ℓ of its columns is always linearly independent). If we collect sufficient amount of codewords, we can compute a parity-check matrix $\mathbf{H}$ for this code, given by the equation A.3.

$$\forall\, \mathbf{c} \in \left\langle \mathbf{B}^{(2)} \right\rangle : \mathbf{Hc} = \mathbf{0} \tag{A.3}$$

The parity-check matrix $\mathbf{H} \in \mathbb{Z}_n^{\ell \times 2\ell}$ must have $\operatorname{rank}(\mathbf{H}) = \ell$ (it generates linear code dual to $\left\langle \mathbf{B}^{(2)} \right\rangle$, their dimensions add up to 2ℓ). To compute $\mathbf{H}$, a linear system with ℓ linearly independent vectors from $\left\langle \mathbf{B}^{(2)} \right\rangle$ is required. This can be done by finding linear relations of the form $\sum_{i=1}^{m} \lambda_i \mathbf{x}_i \equiv \mathbf{0} \pmod{n}$ among known plaintexts and homomorphically create encryptions of zero $\sum_{i=1}^{m} \lambda_i \mathbf{c}_i$, using the coefficients λ_i with corresponding ciphertexts.

Once computed, the parity-check matrix $\mathbf{H}$ can be used to remove the randomized part from a ciphertext, because $\left\langle \mathbf{B}^{(2)} \right\rangle$ is in fact the null-space of $\mathbf{H}$. If we multiply any ciphertext by $\mathbf{H}$, we get a vector, which is only dependent on plaintext, the submatrix $\mathbf{B}^{(1)}$ and the matrix $\mathbf{H}$, as shown in equation A.4 (recall the encryption equation A.1).

$$\mathbf{Hc} = \mathbf{HB}^{(1)}\mathbf{x} + \mathbf{HB}^{(2)}\mathbf{u} = \mathbf{HB}^{(1)}\mathbf{x} \tag{A.4}$$

Let $\mathbf{M} = \mathbf{HB}^{(1)}$ (note that $\mathbf{M} \in \mathbb{Z}_n^{\ell \times \ell}$). The matrix $\mathbf{M}$ is always invertible, because $\mathbf{M}$ always has full rank. The matrix $\mathbf{M}$ is defined as product of matrices $\mathbf{H}$, $\mathbf{B}^{(1)}$, both of rank ℓ. It follows from Sylvester's inequality

$$\operatorname{rank}(\mathbf{A}) + \operatorname{rank}(\mathbf{B}) - \ell \leq \operatorname{rank}(\mathbf{AB}) \leq \min\{\operatorname{rank}(\mathbf{A}), \operatorname{rank}(\mathbf{B})\}$$

(and fact, that $\left\langle \mathbf{B}^{(1)} \right\rangle$ is not null-space of $\mathbf{H}$) that $\mathbf{M}$ must have rank ℓ.

We can therefore always find a matrix $\mathbf{S} = \mathbf{M}^{-1}\mathbf{H}$. This matrix is constant for any product of the form $\mathbf{Hc}$, so computing the product $\mathbf{Sc} = \mathbf{M}^{-1}\mathbf{Hc} = \mathbf{x}$ is equivalent to decryption. Matrix $\mathbf{M}$ can be computed if we collect ℓ linearly independent equations given in equation A.4.

Case $n = p^r$, with p prime
We start by solving the problem modulo p. Once we have computed $\mathbf{H}$ modulo p, we can "lift" it to a solution modulo p^r using a technique similar to Hensel's lifting lemma. Then, we compute $\mathbf{M}$ and invert it (since it is invertible modulo p, it must also be invertible modulo p^r).

We will now describe the lifting process from solutions modulo p^α to solutions modulo $p^{\alpha+1}$. Recall, that in algorithm 3.1 we need to solve the linear system

$$\mathbf{HC} \equiv \mathbf{0} \pmod{p^{\alpha+1}}. \tag{A.5}$$

The matrix $\mathbf{C}$ is known (it is a collection of encryptions of zero). Suppose that we have found $\mathbf{H}_0$, such that $\mathbf{H}_0\mathbf{C} \equiv \mathbf{0} \pmod{p^\alpha}$. A solution $\mathbf{H}' \in \mathbb{Z}_{p^{\alpha+1}}$ will satisfy $\mathbf{H}' \equiv \mathbf{H}_0 \pmod{p^\alpha}$. This means, that we can assume that $\mathbf{H}' = \mathbf{H}_0 + p^\alpha\mathbf{H}_1$ for some $\mathbf{H}_1 \in \mathbb{Z}_p$. Thus we can substitute $\mathbf{H}$ in equation A.5 by $\mathbf{H}_0 + p^\alpha\mathbf{H}_1$ and obtain following system:

$$\mathbf{H}_1\mathbf{C} \equiv \frac{\mathbf{0} - \mathbf{H}_0\mathbf{C}}{p^\alpha} \pmod{p} \tag{A.6}$$

Equation A.6 represents a linear system modulo p, which can be solved using standard methods, as for example Gaussian elimination. To compute a solution modulo p^r (starting with solution modulo p), $r-1$ iterations of the lifting process are needed.

General case $n \in \mathbb{Z}$
Let's assume that $n = \prod_{i=1}^{k} p_i^{r_i}$. We solve the problem modulo $p_i^{r_i}$ to get $\mathbf{S}_i$, such that $\mathbf{x} \equiv \mathbf{S}_i\mathbf{c} \pmod{p_i^{r_i}}$ for $1 \leq i \leq k$. Using CRT we compute $\mathbf{S}$, such that

$$\mathbf{S} \equiv \mathbf{S}_i \pmod{p_i^{r_i}}$$

for $1 \leq i \leq k$ and have $\mathbf{x} \equiv \mathbf{Sc} \pmod{n}$. □

A.2 Proof of lemma 2

The proof consists of two parts. In the first part, we prove the existence of numbers $a, b \in \mathbb{Z}_n$ and then in second part we prove the existence of numbers $t_1, \ldots, t_k \in \mathbb{Z}_n$.

We start by pointing out, that the rings $\mathbb{Z}_n$ and $\mathbb{Z}_p \times \mathbb{Z}_q$ are isomorphic. It is well known, that an isomorphism $f : \mathbb{Z}_p \times \mathbb{Z}_q \to \mathbb{Z}_n$ can be constructed as shown in equation A.7.

$$f((c_1, c_1')) = q(q^{-1} \bmod p)c_i + p(p^{-1} \bmod q)c_i' \bmod n \tag{A.7}$$

Let $a = q(q^{-1} \bmod p) \bmod n$ and $b = p(p^{-1} \bmod q) \bmod n$. We consequently define c_i^*:

$$c_i^* = ac_i + bc_i' \bmod n. \tag{A.8}$$

Following equation A.8, every ciphertext $((c_1, c_1'), \ldots, (c_k, c_k'))$ can be mapped to a unique tuple $(c_1^*, \ldots, c_k^*)$. The elements a, b are constant for any pair c_i, c_i'. This concludes the first part of the proof. Note, that equation A.8 is in fact evaluation of isomorphism f defined in A.7.

Now we will show, that the decryption can be expressed as a single equation modulo n. Recall, that every ciphertext $\mathbf{c} = ((c_1, c_1'), \ldots, (c_k, c_k'))$ satisfies $c_i = m_i r_i \bmod p$ and $c_i' = m_i s_i \bmod q$. Therefore, for every c_i^* defined by equation A.8, we have

$$c_i^* \equiv c_i \equiv r_i x_i \pmod{p},$$
$$c_i^* \equiv c_i' \equiv s_i x_i \pmod{q}.$$

Among elements of $\mathbb{Z}_n$, we can always find $t_1, \ldots, t_k \in \mathbb{Z}_n$, such that

$$t_i \equiv r_i^{-1} \pmod{p}, \tag{A.9}$$
$$t_i \equiv s_i^{-1} \pmod{q}. \tag{A.10}$$

The elements $t_1, \ldots, t_k$ are constant for a fixed secret key. If we then observe the products $t_i c_i^*$ modulo p and q, for $1 \leq i \leq k$ we have

$$t_i c_i^* \equiv r_i^{-1} r_i x_i \equiv x_i \pmod{p},$$
$$t_i c_i^* \equiv s_i^{-1} s_i x_i \equiv x_i \pmod{q}.$$

It follows, that the decryption can be indeed expressed as a linear equation modulo n using the coefficients t_i specified by equations A.9 and A.10, as shown in equation A.11.

$$x = \sum_{i=1}^{k} t_i c_i^* \bmod n \tag{A.11}$$

Finally, substituting c_i^* in equation A.11 by right side of equation A.8 gives rise to equation A.12.

$$x = \sum_{i=1}^{k} t_i (a c_i + b c_i') \bmod n \tag{A.12}$$

□