Alexander Rostovtsev, Alexey Bogdanov and Mikhail Mikhaylov
St. Petersburg state Polytechnic University
rostovtsev@ssl.stu.neva.ru

Secure evaluation of polynomial using privacy ring homomorphisms

Method of secure evaluation of polynomial $y = F(x_1, \ldots, x_k)$ over some rings on untrusted computer is proposed. Two models of untrusted computer are considered: passive and active. In passive model untrusted computer correctly computes polynomial F and tries to know secret input $(x_1, \ldots, x_k)$ and output y. In active model untrusted computer tries to know input and output and tries to change correct output y so that this change cannot be determined.

Secure computation is proposed by using one-time privacy ring homomorphism $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$, $n = pq$, generated by trusted computer. In the case of active model secret check point $v = F(u_1, \ldots, u_k)$ is used. Trusted computer generates polynomial $f(z) = (z - t)(z + t)$, $t \in \mathbb{Z}/n\mathbb{Z}$ and input $X_i(z) \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ such that $X_i(t) \equiv x_i \pmod n$ for passive model, and $f(z) = (z - t_1)(z - t_2)(z - t_3)$, $t_i \in \mathbb{Z}/n\mathbb{Z}$ and input $X_i(z) \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ such that $X_i(t_1) \equiv x_i \pmod n$, $X_i(t_2) \equiv u_i \pmod n$ for active model. Untrusted computer computes function $Y(z) = F(X_1(z), \ldots, X_k(z))$ in the ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$. For passive model trusted computer determines secret output $y \equiv Y(t) \pmod n$. For active model trusted computer checks that $Y(t_2) \equiv v \pmod n$, then determines correct output $y \equiv Y(t_1) \pmod n$.

1. Introduction

Sometimes it is necessary to execute some program with secret data on a computer which cannot be considered as trusted one. Untrusted computer can have low reliability, it can give wrong results, it doesn't keeps data in secret, etc. Usually we can assume that untrusted computer computes function $y = F(x_1, \ldots, x_k)$ for input $(x_1, \ldots, x_k)$ and gives output y.

Some examples of such problems are.

1. Computing salary, taxes on untrusted computer in such a way that real salary and taxes stay unknown and user can check that they are computed correctly.
2. Public key encryption of secret plaintext on untrusted computer in such a way that terminal cannot understand the plaintext and cannot give incorrect ciphertext.

Assume that untrusted computer is controlled by adversary. Usually two general models of untrusted computer (or adversary) are used. In passive model untrusted computer correctly executes the program and tries to find secret input x and secret output y. In active model untrusted computer tries to find x, y and tries to change correct output in such a way that it is hard to recognize such changing.

It is obvious that security can be obtained if one uses trusted computer, that by invertible way transforms input (and possibly function F), gives transformed input to untrusted computer, which computes transformed output and gives it to trusted computer. Trusted computer applies inverse transform and obtains correct output y.

Let φ is the transform, and $\mathsf{x} = \varphi(x)$, $\mathsf{y} = \varphi(y)$, $\mathsf{F} = \varphi(F)$. Such protection method is correct if equalities $\mathsf{y} = \mathsf{F}(\mathsf{x})$ and $y = F(x)$ hold simultaneously. Notice that φ can differently act on set of inputs and outputs and on set of algorithms F. Such protection method is practical if map φ can be computed easily.

Security can be obtained if next conditions hold.

1. For given x, y, F and possibly F it is hard to compute x, y.
2. In the case of active model texts (x, y) must contain redundancy, which allows to recognize true and wrong outputs.
3. For given pairs (x_1, y_1), …, (x_m, y_m) and given x_{m+1} it is hard to compute such incorrect output y_{m+1}, that pair (x_{m+1}, y_{m+1}) will be not rejected.

Suitable way for constructing secure computations is using homomorphisms of algebraic structures (groups, rings). Notice that since all field homomorphisms are injective, such homomorphisms are non-common.

Using equations $\mathsf{y} = \mathsf{F}(\mathsf{x})$ and $y = F(x)$ we can be write $\varphi(y) = \varphi(F)(\varphi(x))$ or in multiplicative notations $\varphi(Fx) = \varphi(F)\varphi(x)$. In the case of groups last equation corresponds to group homomorphism. Here data x and algorithm F must be members of the same group, which is non-common. Sometimes it is possible to apply homomorphism φ to input x only. Finite Abelian group is isomorphic to direct sum of cyclic groups [7]. If homomorphism of Abelian groups maps each generator of cyclic group to generator of group of the same order, it is invertible. For example, well-known RSA cryptosystem [10] maps $x \to x^e \pmod{n}$ for composite $n = pq$ with secret factorization and uses privacy homomorphisms of finite Abelian group $(\mathbb{Z}/n\mathbb{Z})^*$.

If we want to provide secure computation on untrusted computer, ring homomorphisms seem more common then group ones. Here algorithm F can be considered as sequence of ring operations (additions, multiplications, inversions and divisions if it is possible). It is known that computable function F can be represented as composition of computable Boolean functions (or normal algebraic form polynomials). So secure computation can be obtained if one can build privacy ring homomorphism of normal algebraic form. This problem is far from being solved. Well known privacy group homomorphism $(\mathbb{Z}/n\mathbb{Z})^* \to \{0, 1\}^+$, $n = pq$, for input x such that Jacobi symbol $\left(\frac{x}{n}\right)=1$, has image 0 if x is a square modulo n (in this case $\left(\frac{x}{p}\right)=\left(\frac{x}{q}\right)=1$), it has image 1 if x is not a square modulo n (in this case $\left(\frac{x}{p}\right)=\left(\frac{x}{q}\right)=-1$). But this homomorphism does not hold under multiplication and hence cannot be used for normal algebraic form polynomials.

Usually elements of rings can be defined by polynomials. For passive model secure polynomial evaluation is oblivious polynomial evaluation (untrusted computer cannot have any information about value of y for given value of variable x). Problem of oblivious polynomial evaluation over field is connected with problem of noisy polynomial reconstruction [8]. Theoretically secure polynomial evaluation protocol, required pre-distributed data, was performed in [11]. Oblivious polynomial evaluation technique based on zero-knowledge proofs was performed in [4]. Such approaches cannot provide large computation scale and are not practical. Secure polynomial evaluation based on privacy ring homomorphisms seems to be more practical.

Large class of computable function can be described using ring of integers $\mathbb{Z}$. There are well-known ring homomorphisms $\mathbb{Z}[z] \to \mathbb{Z}$, computed by substituting $z = t$ for some $t \in \mathbb{Z}$. Secure input and output are polynomials: $\varphi(x)$, $\varphi(y) \in \mathbb{Z}[z]$. If algorithm F takes some multiplications, degree of output polynomial as element of $\mathbb{Z}[z]$ can be very large, and computations become too slow. Degree of output polynomial can be limited if instead of $\mathbb{Z}[z]$ one uses $\mathbb{Z}[z]/(f(z))$, where polynomial $f(z)$ has at least one root in $\mathbb{Z}$. Ring homomorphism $\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}$ is computed substituting some root of polynomial $f(z)$ instead z. Hence computation of the ring homomorphism is equivalent to computation of roots of polynomial $f(z)$. Polynomials over finite fields can be easily factored [3]. This factorization can be lifted to ring of integers $\mathbb{Z}$ by Hensel's lemma. Hence this homomorphism is not secure. Sometimes ring of algebraic integers $R = \mathbb{Z}[w]/(g(w))$ for irreducible polynomial $g(w)$ is considered instead of $\mathbb{Z}$. Polynomials over R also can be factored. It is sufficient to consider polynomials over finite field, given by polynomial $g(w)$, find its roots and lift them to ring R.

The goal of this paper is description of privacy homomorphisms of ring $\mathbb{Z}/n\mathbb{Z}$ and its application to secure evaluation of polynomials over some rings on untrusted computer both for passive and active models.

2. Mathematical background and privacy homomorphisms

Let $p \neq q$ are large integers and $n = pq$. Ring $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \oplus \mathbb{Z}/q\mathbb{Z}$ is widely used in public key cryptosystems. Set of such cryptosystems includes RSA [10], Fiat-Shamir [2], public-key cryptosystem based on permutation polynomials [1]. Paillier cryptosystem [9] and its elliptic curve analog [3] are defined over rings $\mathbb{Z}/n^2\mathbb{Z} \cong \mathbb{Z}/p^2\mathbb{Z} \oplus \mathbb{Z}/q^2\mathbb{Z}$. Security of those cryptosystems is based on complexity of integer factorization problem.

There are some well-known number theory problems connected with factorization problem.

1. Computing Euler function $\phi(n) = (p - 1)(q - 1)$ is equivalent to factoring. Indeed, if p, q are known, then $\phi(n)$ can be computed. Back, if n, $\phi(n)$ are known, then p, q are roots of polynomial $x^2 - (n + 1 - \phi(n))x + n$.
2. Quadratic equation solving $x^2 - D \equiv 0 \pmod{n}$ is equivalent to factoring. The congruence holds if and only if hold both $x^2 - D \equiv 0 \pmod{p}$, $x^2 - D \equiv 0 \pmod{q}$. Last two equations have 2 roots, so equation $x^2 - D \equiv 0 \pmod{n}$ has 4 roots. If those roots t_i are known, then $p = \mathrm{GCD}(n, t_i + t_j)$. Back, if p, q are known, then equations $x^2 - D \equiv 0 \pmod{p}$, $x^2 - D \equiv 0 \pmod{q}$ can be easily solved, roots of these equations give 4 roots modulo n.
3. If class number h of imaginary quadratic order of discriminant $D = -n$ or $D = -4n$ can be computed (since n is composite, h is even), then number n can be factored. Indeed, let $h = h_1 2^k$, h_1 is odd. Take arbitrary reduced quadratic form (a, b, c), $b^2 - 4ac = D$, $a > 1$ and exponent it to power h_1 and then successively square it. With probability ≈ 0.5 one obtains reduced form (a_1, a_1, c_1) and $p = \mathrm{GCD}(n, a_1)$, or form $(a_1, b_1\ a_1)$ and $p = \mathrm{GCD}(n, 2a_1 - b_1)$. Inverse reduction is not true: for given large primes p, q it is hard to compute class number of imaginary quadratic order of discriminant $D = -n$ or $D = -4n$.

Consider privacy ring homomorphisms for passive and active models. Computing of such homomorphism doesn't take knowledge of factorization of n.

Algorithm 1. Generating ring homomorphism $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$, $x \mapsto X(z)$ for passive model.

Input: $n = pq$, $x \in \mathbb{Z}/n\mathbb{Z}$.

Output. polynomial $f(z)$, $X(z) \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$.

Method.

1. Generate random $t \in \mathbb{Z}/n\mathbb{Z}$ and compute $D \equiv t^2 \pmod{n}$.
2. $f(z) = z^2 - D$.
3. Compute $X(z) = z - t + x$.
4. Return: $f(z)$, $X(z)$.

Polynomial $f(z)$ not necessary has degree 2 for passive model. Sometimes it is common to use polynomials of more degree.

Protocol 2. Secure computation $y = F(x)$ for passive model.
Entities: trusted computer, untrusted computer.
Secret input of trusted computer: x.
Common input of trusted and untrusted computers: number n with unknown (to untrusted computer) factorization.
Output: $y = F(x)$.
Method.

1. Trusted computer generates secret ring homomorphism $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$, $x \mapsto X(z)$ using algorithm 1 and sends $f(z)$, $X(z)$ and algorithm F to untrusted computer.
2. Untrusted computer computes $Y(z) = F(X(z))$ in the ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ and sends $Y(z)$ to trusted computer.
3. Trusted computer computes $y \equiv Y(t) \pmod{n}$.

Algorithm 3. Generating ring homomorphism $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$, $x \mapsto X(z)$ for active model.

Input: $n = pq$, $x \in \mathbb{Z}/n\mathbb{Z}$, check point (u, v), $v = F(u)$.

Output. polynomial $f(z)$, $X(z) \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$.

Method.

1. Generate random $t_1, t_2 \in \mathbb{Z}/n\mathbb{Z}$.
2. $f(z) = (z - t_1)(z - t_2)(z + t_1 + t_2) = z^3 - (t_1^2 + t_1t_2 + t_2^2)z + t_1t_2(t_1 + t_2)$.
3. Compute $a = t_1 + t_2 + \dfrac{u - x}{t_1 - t_2} \pmod{n}$, $b = t_1t_2 - \dfrac{t_1u - t_2x}{t_1 - t_2} \pmod{n}$, $X(z) = z^2 + az + b \pmod{n}$ (these equations satisfy conditions $X(t_1) \equiv x \pmod{n}$, $X(t_2) \equiv u \pmod{n}$).
4. Return: $f(z)$, $X(z)$.

Polynomial $f(z)$ not necessary has degree 3 for active model. Sometimes it is common to use polynomials of more degree.

Protocol 4. Secure computation $y = F(x)$ for active model.

Entities: trusted computer, untrusted computer.

Secret inputs of trusted computer: x; check point (u, v) for $v = F(u)$.

Common input of trusted and untrusted computers: number n with unknown (to untrusted computer) factorization.

Output: $y = F(x)$.

Method.

1. Trusted computer generates secret ring homomorphism $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$, $x \mapsto X(z)$ using algorithm 3 and sends $f(z)$, $X(z)$ and algorithm F to untrusted computer.
2. Untrusted computer computes $Y(z) = F(X(z))$ in the ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ and sends $Y(z)$ to trusted computer.
3. Trusted computer checks equality $Y(t_2) \equiv v \pmod{n}$. If it holds, output is probably true.
4. Trusted computer computes $y \equiv Y(t) \pmod{n}$.

Homomorphisms of ring $\mathbb{Z}/n^2\mathbb{Z}$ and secure computation in this ring can be obtained similarly.

Theorem 1. Let $f(z) = (z - t_1)\ldots(z - t_d)$, $t_i \in \mathbb{Z}/n\mathbb{Z}$ for $1 \le i \le n$, is square-free polynomial and t is a root of $f(z)$. Then rings $\mathbb{Z}/n\mathbb{Z}[t]/(f(t)$ and $\mathbb{Z}/n\mathbb{Z}$ coincide.

Proof. Let $g(z)$, $h(z) \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ and $g(t)$, $h(t) \in \mathbb{Z}/n\mathbb{Z}$ — their images under substitution z **a** t. Then image of $g(z) + h(z)$ is $g(t) + h(t) \in \mathbb{Z}/n\mathbb{Z}$, image of $g(z)h(z)$ is $g(t)h(t) \in \mathbb{Z}/n\mathbb{Z}$. So substitution z **a** t is ring homomorphisms $\mathbb{Z}/n\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}/n\mathbb{Z}$ and $\mathbb{Z}/n\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}/n\mathbb{Z}[t]/(f(t)) \subseteq \mathbb{Z}/n\mathbb{Z}$. Kernel of the first homomorphism is ideal $(z - t)$. Kernel of the second homomorphism is also ideal $(z - t)$. According to ring homomorphism theorem if domains and kernels of homomorphism coincide, images of homomorphism are isomorphic. For arbitrary c homomorphic images of polynomial $cz - ct + 1$ are 1 for both homomorphisms. Under addition group $\mathbb{Z}/n\mathbb{Z}$ is cyclic with generator 1. Hence rings $\mathbb{Z}/n\mathbb{Z}[t]$ and $\mathbb{Z}/n\mathbb{Z}[t]/(f(t)$ are homomorphic images of ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ and coincide. **n**

Hence surjective ring homomorphism $\mathbb{Z}/n\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}/n\mathbb{Z}[t]$ is given by substitution z **a** t. Composition of homomorphic embedding $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ according to Algorithm 1, 3 and homomorphism $\mathbb{Z}/n\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}/n\mathbb{Z}$ is identity map of $\mathbb{Z}/n\mathbb{Z}$. Indeed, if $X(z)$ is the image of x under embedding, then second homomorphism gives $X(t) \equiv x \pmod{n}$.

Ring $\mathbb{Z}/n\mathbb{Z}$ has important property: factoring of square free polynomials over this ring is hard problem. So ring homomorphism $\mathbb{Z}/n\mathbb{Z}[z]/(f(z)) \to \mathbb{Z}/n\mathbb{Z}[t]/(f(t))$ is hard to compute if element t as a root of $f(z)$ is unknown. Other common rings such as $\mathbb{Z}$, $\mathbb{Q}$, finite fields do not possess this property. Polynomial over finite field can be easily factored [5]. Polynomials over $\mathbb{Z}$, $\mathbb{Q}$ can be factored using algorithm [5] and p-adic lifting according to Hensel's lemma.

Theorem 2. If square free polynomial $f(z) \in \mathbb{Z}/n\mathbb{Z}[z]$, $n = pq$, factors over $\mathbb{Z}/n\mathbb{Z}$: $f(z) = \prod_{i=1}^{d}(z - t_i)$, $d \ge 2$, and $\mathrm{GCD}(t_i, t_j)$ is invertible element in $\mathbb{Z}/n\mathbb{Z}$ for any $t_i \ne t_j$, then polynomial $f(z)$ has d^2 different roots in $\mathbb{Z}/n\mathbb{Z}$.

Proof. According to Chinese remainder theorem $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \oplus \mathbb{Z}/q\mathbb{Z}$. Hence polynomial $f(z)$ has d different roots $u_1, \ldots, u_d$ in $\mathbb{Z}/p\mathbb{Z}$ and has d different roots $v_1, \ldots, v_d$ in $\mathbb{Z}/q\mathbb{Z}$. Any pair u_i, v_j of roots by Chinese remainder theorem defines element $z_{i,j} \in \mathbb{Z}/n\mathbb{Z}$ which is a root of $f(z)$ modulo n, different pairs define different $z_{i,j}$. There are d^2 such different pairs and d^2 roots $z_{i,j}$ of polynomial $f(z)$ over $\mathbb{Z}/n\mathbb{Z}$. **n**

Theorem 3. Let $f(z)$ is polynomial as in theorem 2. Next claims are equivalent.

1. Factorization $n = pq$ is known.
2. It is possible to compute or to guess $d + 1$ roots of $f(z) \in \mathbb{Z}/n\mathbb{Z}[z]$.
3. It is possible to compute or to guess all d^2 roots of $f(z) \in \mathbb{Z}/n\mathbb{Z}[z]$.

Proof. 1 $\Rightarrow$ 2, 3. Let factorization $n = pq$ is known. By theorem 2 d^2 roots of polynomial $f(z)$ over $\mathbb{Z}/n\mathbb{Z}$ are defined by its roots over finite fields $\mathbb{Z}/p\mathbb{Z}$, $\mathbb{Z}/q\mathbb{Z}$. Roots of polynomials over finite fields can be computed effectively [5]. Then using Chinese remainder theorem, one can compute all d^2 roots over $\mathbb{Z}/n\mathbb{Z}$ and hence $d + 1$ roots.

2 $\Rightarrow$ 1, 3. Let $d + 1$ roots $z_1, \ldots, z_{d+1}$ of polynomial $f(z) \in \mathbb{Z}/n\mathbb{Z}[z]$ are known. Define $z_i \equiv u_i \pmod{p}$, $z_i \equiv v_i \pmod{q}$. Since $\deg(f) = d$, there exists a pair $z_i, z_j \in \mathbb{Z}/n\mathbb{Z}$ such that $z_i - z_j \equiv 0 \pmod{p}$ and $z_i - z_j \neq 0 \pmod{q}$. Then $\mathrm{GCD}(n, z_i - z_j) = p$ and $q = n/p$. If factorization of n is known, all d^2 roots of $f(z)$ over $\mathbb{Z}/n\mathbb{Z}$ can be computed as above.

Implication 3 $\Rightarrow$ 2 is trivial. **n**

3. Applications and limitations

Some public key cryptosystems are defined in terms of ring $\mathbb{Z}/n\mathbb{Z}$, its finite extension or ring of polynomials over $\mathbb{Z}/n\mathbb{Z}$ ([1], [2], [3], [9], [10]). These cryptosystems can be realized by direct application of algorithms, described in the section 2.

Ring $\mathbb{Z}/n\mathbb{Z}$ with proposed homomorphisms admits next operations: addition/subtraction, multiplication, inversion modulo n if it is possible. Representing element $z \in \mathbb{Z}/n\mathbb{Z}$ as absolutely minimal residue allows interpret computation in $\mathbb{Z}/n\mathbb{Z}$ as computation in subset of integers $\mathbb{Z}$ that absolutely do not exceed $n/2$. Computing in the field $\mathbb{Q}$ can be defined as computing over integers and hence over $\mathbb{Z}/n\mathbb{Z}$ for numerators and denominators. Computing in the ring $\mathbb{Z}_n$ of n-adic integers can be represented by computing in $\mathbb{Z}/n\mathbb{Z}$ according to Hensel's lemma. Proposed ring homomorphisms can be easily transformed to finite extension of initial ring: to ring $\mathbb{Z}/n\mathbb{Z}[w]/(g(w))$ or to ring $\mathbb{Z}[w]/(g(w))$, where g is a polynomial. Hence proposed method can be applied for secure evaluation of polynomials over those rings.

But there are some limitations if polynomials over ordered rings are used. For example, in ring $\mathbb{Z}$ inequality holds $1 > 0$, hence this ring is ordered. Ring of algebraic integers also is ordered by norm map. Ring $\mathbb{Z}/n\mathbb{Z}$ has characteristic n: $0 = 1 + \ldots + 1$ (n times), this ring is non-ordered. Indeed, if we assume that $1 > 0$, then $1 + 1 = 2 > 0$ and hence $n = 0 > 0$ — contradiction.

Absolutely small elements of $\mathbb{Z}$ and their images in quotient ring $\mathbb{Z}/n\mathbb{Z}$ as absolutely minimal residues coincide. Hence small positive and negative integers as elements of $\mathbb{Z}/n\mathbb{Z}$ can be recognized. But their images under privacy homomorphism can be not small. So generally it is impossible to recognize when element of ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ is positive, negative or zero. If we protect secret data with proposed privacy homomorphism, then branching programs with protected data such as "If argument is positive, do step A, else do step B" are impossible. Ring $\mathbb{Z}$ is ordered and Euclidean, it admits division, but in ring $\mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ there is no division. This property limits possible applications of proposed privacy homomorphisms. For example, exponentiation in prime finite field or exponentiation on elliptic curve over prime finite field cannot be realized with proposed homomorphisms.

Consider some cryptographic protocols based on proposed privacy homomorphism.

Protocol 5. RSA public key encryption [10] on untrusted computer (active model).

It is required to encrypt secret plaintext on untrusted computer in such a way that the computer cannot know the plaintext.

Entities: trusted computer and untrusted computer.

Common input: composite number $n = pq$, public exponent e.

Secret input of trusted computer: plaintext x, check point (u, v), $v \equiv u^e \pmod{n}$.

Output: ciphertext $y \equiv x^e \pmod{n}$.

Method.

1. Trusted computer generates secret homomorphism $(f(z), X(z))$ using algorithm 3 and sends it to untrusted computer.
2. Untrusted computer computes ciphertext $Y = X^e \in \mathbb{Z}/n\mathbb{Z}[z]/(f(z))$ and sends it to trusted computer.
3. Trusted computer checks that $Y(t_2) \equiv v \pmod{n}$ and if it is so, finds ciphertext $y \equiv Y(t_1) \pmod{n}$.

Consider small example. Let $p = 57$, $q = 79$, $n = 47 \cdot 79 = 3713$, $\phi(n) = 3588$. Let secret exponent is $e = 101$, check point is $u = 1002$, $v = u^e = 164$, plaintext is $x = 1234$.

Compute ring homomorphism. Choose $t_1 = 502$, $t_2 = 2233$ and find $f(z) = z^3 + 1110z + 3058$. Compute coefficients of the image of plaintext according to algorithm 3: $a = 255$, $b = 3659$, $X(z) = 3659 + 255z + z^2$.

Encrypt image of plaintext. $Y(z) = 2995 + 1425z + 2417z^2$.

Test that computation has no errors. $Y(t_2) \equiv v \equiv 164 \pmod{n}$, hence result likes probably true.

Find the true ciphertext $y \equiv Y(t_1) \equiv 32 \pmod{n}$. Its coincides with $x^e \pmod{n}$ computed by common method.

Map $\mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}$, $z \ \mathbf{a}\ z^e (\mathrm{mod}\, n)$, $\mathrm{GCD}(e, \phi(n)) = 1$, is invertible. Hence this map defines permutation of elements of $\mathbb{Z}/n\mathbb{Z}$ and monomial z^e is permutation polynomial over $\mathbb{Z}/n\mathbb{Z}$. There are some other classes of permutation polynomials over $\mathbb{Z}/n\mathbb{Z}$ [1]: Lucas polynomials, Dickson polynomials, elliptic curve division polynomials (elements of affine coordinate ring of corresponding elliptic curve), etc. Permutation polynomial has inverse under polynomial composition. If $g(h(z)) = z$, then $h(g(h(z))) = h(z)$ and applying permutation, inverse to permutation defined by $h(z)$ we obtain $h(g(z)) = z$. So polynomials $g(z)$ and $h(z)$ are mutually inverse under composition. If polynomial $g(z)$ defines encryption, then polynomial $h(z)$ defines decryption. Analogs of RSA cryptosystem can be obtained changing monomial z^e by some permutation polynomial. Proposed privacy ring homomorphism can be applied to any polynomial over $\mathbb{Z}/n\mathbb{Z}$ and hence to any class of permutation polynomials and protocol 5 can be easily transformed to public key encryption defined by some permutation polynomial over $\mathbb{Z}/n\mathbb{Z}$. Notice that due to ring isomorphism $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \oplus \mathbb{Z}/q\mathbb{Z}$, permutation polynomials can be computed over prime finite fields, Chinese remainder theorem gives permutation polynomial over $\mathbb{Z}/n\mathbb{Z}$.

Consider Paillier elliptic curve encryption/decryption [3]. This protocol is similar to [9] and exploits elliptic curve $E(\mathbb{Z}/n^2\mathbb{Z})$: $V^2W = U^3 + AUW^2 + BW^3 \pmod{n^2}$ with number of points mn (number of points $\#E(\mathbb{Z}/n\mathbb{Z}) = m$ is secret key) and fixed point $Q \in E(\mathbb{Z}/n^2\mathbb{Z})$ of order m, $\mathrm{GCD}(m, n) = 1$. This point can be computed from random point $Q' \in E(\mathbb{Z}/n^2\mathbb{Z})$, $Q = nQ'$. Since elements of $\mathbb{Z}/n\mathbb{Z}$ form ideal $\mathbb{Z}/n^2\mathbb{Z}$, elliptic curve $E(\mathbb{Z}/n^2\mathbb{Z})$ has reduction modulo n and number of points of E modulo n is m. Point at infinity $(0, 1, 0) \in E(\mathbb{Z}/n\mathbb{Z})$ can be lifted to point $(xn, 1, 0) \in E(\mathbb{Z}/n^2\mathbb{Z})$ for any x, $0 \le x < n$. Note that multiplying point $(n, 1, 0) \in E(\mathbb{Z}/n^2\mathbb{Z})$ by exponent x one obtains point $(xn, 1, 0)$.

Paillier elliptic curve encryption of plaintext x takes next operations.

1. Generating random number r, $1 \le r \le n$.
2. Computing over ring $\mathbb{Z}/n^2\mathbb{Z}$ point rQ, computes point $S = (xn, 1, 0) + rQ$. Ciphertext is point S.

Paillier public key decryption of ciphertext S takes elliptic curve point multiplication by m. Then $mS = mrQ + m(xn, 1, 0) = (mxn, 1, 0)$. Dividing U-coordinate of this point by n, one obtains $mx \pmod{n}$ and finds $x \equiv m^{-1}mx \pmod{n}$.

Since untrusted computer computes point rQ for random number r, using check point is uncommon. So only passive model can be considered.

Secure decryption of ciphertext can be executed by next protocol.

Protocol 6. Paillier elliptic curve encryption on untrusted computer (passive model).

It is required to decrypt ciphertext on untrusted computer in such a way that the computer cannot know the plaintext.

Entities: trusted computer and untrusted computer.

Common input: a square of composite number, $n^2 = (pq)^2$, elliptic curve $E(\mathbb{Z}/n^2\mathbb{Z})$: $V^2W = U^3 + AUW^2 + BW^3 \pmod{n^2}$, point $Q \in E(\mathbb{Z}/n^2\mathbb{Z})$.

Secret input of trusted computer: plaintext x, point $Q \in E(\mathbb{Z}/n^2\mathbb{Z})$ of order m.

Output: ciphertext y.

Method.

1. Trusted computer generates secret homomorphism $(f(z), X(z), Q(z))$ using algorithm 1 or algorithm 3 for arbitrary check input, $\deg(f) \geq 3$, and sends it to untrusted computer (it is possible to use original point Q instead of $Q(z)$).
2. Untrusted computer generates random number r, $1 \leq r \leq n$, computes point $rQ(z)$ (or point rQ), computes point $Y(z) = (X(z)n, 1, 0) + rQ(z)$ and sends it to trusted computer.
3. Trusted computer finds $y \equiv Y(t) \pmod{n}$.

Consider small example. Let $p = 37$, $q = 53$, $n = 37 \cdot 53 = 1961$. Elliptic curve $V^2W = U^3 + 2UW^2 + 8W^3$ has $m = 1581$ points over $\mathbb{Z}/n\mathbb{Z}$ and has mn points over $\mathbb{Z}/n^2\mathbb{Z}$. Point $Q = (517462, 1, 1733727)$ has order m. Plaintext is $x = 132$.

Trusted computer computes ring homomorphism for the case when point Q is unchanged. It chooses $t_1 = 502$, $t_2 = 1000$, finds $f(z) = 281884 + 2091517z + z^3$, $X(z) = 655492 + 1882651z + z^2$ and sends $f(z)$, $X(z)$, Q to untrusted computer.

Untrusted computer computes point $(nX(z), 1, 0)$, generates random number $r = 1174$, computes point $Y(z) = rQ + (nX(z), 1, 0) = (3151615 + 370629z + 2074728z^2, 1, 753386+727531z + 1078660z^2)$ and sends point $Y(z)$ to trusted computer.

Trusted computer finds true ciphertext $y \equiv Y(t_1) \pmod{n^2} = (3275158, 1, 3559577)$. Decryption of this ciphertext gives the plaintext.

5. Security

Problem of computing homomorphism between two general rings is in complexity class AM $\cap$ co-AM and is not NP-complete [6].

Semantic security of proposed scheme for passive model depends on hardness of finding a root of polynomial $f(z)$ over $\mathbb{Z}/n\mathbb{Z}$. Given homomorphic image $X(z)$ can correspond to arbitrary $x \equiv X(t) \pmod{n}$ with the same probabilities.

Proposed ring homomorphism seems to be secure, at least for passive model. Indeed, if $f(z) = z^2 - D$, then computing its root $t \equiv \pm\sqrt{D} \pmod{n}$ takes computing $\sqrt{D} \pmod{n}$ and hence computing $\varphi(n)$ or factorization of number n. Other algorithms for computing $\sqrt{D} \pmod{n}$ are unknown.

Theorem 4. If there exists polynomial-time algorithm that computes one of four possible values of $\sqrt{D} \pmod{n}$ for any D, which is a square in $\mathbb{Z}/n\mathbb{Z}$, then there exists probabilistic polynomial-time algorithm of factorization of n with probability arbitrary near to 1.

Proof. Choose arbitrary $t \in (\mathbb{Z}/n\mathbb{Z})^*$, compute $D \equiv t^2 \pmod n$. Applying algorithm of computing square root, find $w \equiv \sqrt{D} \pmod n$. Since there are 4 different values $\sqrt{D} \pmod n$, with probability 0,5 we have $w \neq \pm t \pmod n$. If this condition holds, then GCD $(n, t + w)$ is divisor of n. Else take another t and repeat the procedure. After k attempts probability of factorization of n is $1 - 2^{-k}$. Is S is complexity of computing $\sqrt{D} \pmod n$, then after k attempts complexity of factorization is $O(k(S + (\log n)^2))$. ∎

Privacy homomorphism with check point cannot provide security if it is twice. Indeed let untrusted computer obtains two inputs $X_1(z)$, $X_2(z)$ and correct output $Y_1(z)$ corresponding to input $X_1(z)$. Then computing GCD$(X_1(z), X_2(z)) = (z - t_2)$, untrusted computer can find true check result $Y(t_2)$ and can change true result so that trusted computer cannot recognize the change. Hence, privacy homomorphism can be attacked using birthday paradox. But if length of number n is about 1000 bits, this attack is impossible in practice.

Besides of that privacy homomorphism with check point is secure only if the check point (input, output and intermediate data) is secret. Indeed, if untrusted computer knows check input u and check output v, represented by polynomials $X(z)$, $Y(z)$, then $X(z) - u \equiv 0 \pmod{(z - t_2)}$, $Y(z) - v \equiv 0 \pmod{(z - t_2)}$. Hence with probability near to 1 equality holds $z - t_2 = \mathrm{GCD}(X(z) - u, Y(z) - v)$. This allows computing t_2, and changing true result $Y(z)$ by any $Y(z) \equiv v \pmod{(z - t_2)}$ so that trusted computer cannot recognize the change. Instead of u, v intermediate data can be used.

Cardano formulas for computing roots of polynomial $f(z) = z^3 + Az + B$ over $\mathbb{Z}/n\mathbb{Z}$ takes computing $Q = \sqrt{\frac{A^3}{27} + \frac{B^2}{4}} \pmod n$, $w_1 = \sqrt[3]{\frac{-B}{2} + Q}$, $w_2 = \sqrt[3]{\frac{-B}{2} - Q}$, $\sqrt{-3} \pmod n$ The roots of the polynomial are:

$$t_1 = w_1 + w_2 \pmod n,$$

$$t_2 = -\frac{w_1 + w_2}{2} + \frac{w_1 - w_2}{2}\sqrt{-3} \pmod n,$$

$$t_3 = -\frac{w_1 + w_2}{2} - \frac{w_1 - w_2}{2}\sqrt{-3} \pmod n.$$

Known algorithms of computing a root of cubic polynomial $f(z)$ takes computing $\phi(n)$ and hence factorization of n.

Bibliography

1. Castagnos V. and Vergnaud D. Trapdoor permutation polynomials over Z/nZ and public key cryptosystems // Proceedings of ISC 2007, pp. 333–350.
2. Fiat A., Shamir A. How to prove yourself: Practical solutions to identification and signature problems // Advances in Cryptology — CRYPTO′85. LNCS, v. 263, Springer-Verlag, 1990, pp. 186–194.
3. Galbraith S. Elliptic curve Paillier schemes // Journal of cryptology, v. 15, No. 3, 2001, pp. 129–138.
4. Hazay C. and Lindell Y. Efficient oblivious polynomial evaluation with simulation-based security // International association for Cryptologic research, e-print IACR archive, 2009-459.
5. Kaltofen E. and Shoup V.. Subquadratic-time factoring of polynomials over finite fields // Mathematics of computation, v. 67, Number 223, 1998, pp. 1179–1197.
6. Kayal N. and Saxena N. Complexity of ring morphism problems // Computational Complexity archive, v. 15, issue 4, 2006, pp. 342–390.
7. Lang S. Algebra, Addison-Wesley publishing company, 1965.
8. Naor M. and Pinkas B. Oblivious polynomial evaluation // SIAM J. of computing, v. 35, number 5, 2006, pp. 1254–1281.
9. Paillier P. Public key cryptosystem based on composite degree residousity classes // EUROCRYPT'99, LNCS, v. 1592, Springer-Verlag, 1999, pp. 223–238.
10. Rivest R. L., Shamir A. and Adleman L. A method for obtaining digital signatures and public-key cryptosystems // Communications of the ACM, v. 21, № 2, 1978, pp. 120–126.
11. Tonicelli R., Dowsley R., Hanaoka G., Imai H., Muller-Quadi J., Otsuka A. and Nascimento A.C.A. Sequentially composable information theoretically secure oblivious polynomial evaluation // International association for Cryptologic research, e-print IACR archive, 2009-270.