# DoubleMod and SingleMod: Simple Randomized Secret-Key Encryption with Bounded Homomorphicity

Dhananjay S. Phatak,[*] Qiang Tang,[†] Alan T. Sherman,[*]
Warren D. Smith,[‡] Peter Ryan,[†] Kostas Kalpakis[*]

April 2011 (revised February 6, 2014)

**Abstract**

An encryption relation $f \subseteq \mathbb{Z}\times\mathbb{Z}$ with decryption function $f^{-1}$ is *"group-homomorphic"* if, for any suitable plaintexts $x_1$ and $x_2$, $x_1+x_2 = f^{-1}(f(x_1)+f(x_2))$. It is *"ring-homomorphic"* if furthermore $x_1x_2 = f^{-1}(f(x_1)f(x_2))$; it is *"field-homomorphic"* if furthermore $1/x_1 = f^{-1}(f(1/x_1))$. Such relations would support oblivious processing of encrypted data.

We propose a simple randomized encryption relation $f$ over the integers, called *DoubleMod*, which is "bounded ring-homomorphic" or what some call "somewhat homomorphic." Here, "bounded" means that the number of additions and multiplications that can be performed, while not allowing the encrypted values to go out of range, is limited (any pre-specified bound on the operation-count can be accommodated). Let $R$ be any large integer. For any plaintext $x \in \mathbb{Z}_R$, DoubleMod encrypts $x$ as $f(x) = x + au + bv$, where $a$ and $b$ are randomly chosen integers in some appropriate interval, while $(u, v)$ is the secret key. Here $u > R^2$ is a large prime and the smallest prime factor of $v$ exceeds $u$. With knowledge of the key, but not of $a$ and $b$, the receiver decrypts the ciphertext by computing $f^{-1}(y) = (y \bmod v) \bmod u$.

DoubleMod generalizes an independent idea of van Dijk *et al.* 2010. We present and refine a new CCA1 chosen-ciphertext attack that finds the secret key of both systems (ours and van Dijk *et al.*'s) in linear time in the bit length of the security parameter. Under a known-plaintext attack, breaking DoubleMod is at most as hard as solving the *Approximate GCD (AGCD)* problem. The complexity of AGCD is not known.

We also introduce the *SingleMod* field-homomorphic cryptosystems. The simplest SingleMod system based on the integers can be broken trivially. We had hoped, that if SingleMod is implemented inside non-Euclidean quadratic or higher-order fields with large discriminants, where GCD computations appear difficult, it may be feasible to achieve a desired level of security. We show, however, that a variation of our chosen-ciphertext attack works against SingleMod even in non-Euclidean fields.

[*]Cyber Defense Lab, CSEE Department, University of Maryland, Baltimore County (UMBC), Baltimore, MD 21250 USA. {phatak, sherman, kalpakis}@umbc.edu

[†]Computer Science and Communications Research Unit, University of Luxemborg, 6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg. {qiang.tang, peter.ryan}@uni.lu

[‡]warren.wds@gmail.com

**Keywords.** Approximate GCD problem (AGCD), cryptanalysis, cryptography, cryptology, DoubleMod, homomorphic encryption, lattice algorithms, non-Euclidean algebraic number fields, randomized encryption, SingleMod.

## 1 Introduction

For decades, cryptographers have sought *homomorphic encryption systems*[1] because of their potential to support oblivious processing of encrypted data. In his dissertation, Gentry [18, p. 6] views homomorphic encryption as "glove box" operations on digital data, wherein an untrusted entity can perform useful calculations on data without touching (seeing) the plaintext. The utility of the glove box depends on its efficiency, cryptographic strength, and the types of operations it supports. We present and analyze two new symmetric-key randomized encryption systems over the integers—*DoubleMod* and *SingleMod*—that enjoy homomorphic properties, and in comparison with most previously-proposed homomorphic encryption systems, are simpler. They support encryption of multiple bits without bootstrapping. Unfortunately, they (and all other integer homomorphic systems we are aware of) turn out to be insecure against CCA1 chosen-ciphertext attack[2] that finds the secret key.

Phatak and Sherman discovered DoubleMod in spring 2011. By supporting the encryption of mulitple bits, DoubleMod generalizes an independent idea of van Dijk *et al.* [55] published in 2010, which encrypts one bit at a time. Independently from us, Pisa *et al.* [39] also discovered DoubleMod and published it in 2012. Our paper includes a new CCA1 chosen-ciphertext attack on DoubleMod.

An encryption relation $f \subseteq \mathbb{Z}\times\mathbb{Z}$ with decryption function $f^{-1}$ is *"group-homomorphic"* if, for any plaintexts $x_1$ and $x_2$, $x_1 + x_2 = f^{-1}(f(x_1) + f(x_2))$. It is *"ring-homomorphic"* if furthermore $x_1 x_2 = f^{-1}(f(x_1)f(x_2))$. For an introduction to homomorphic encryption, see Wikipedia [2] and Hayes [30].

In 2009, Gentry [19, 20, 21] proposed the first ring-homomorphic system, based on ideal lattices. But many researchers, including Micciancio [36], noted serious practical obstacles for this approach. Since then several variants—including that of van Dijk—have been proposed that require less message expansion. For example, improvements over van Dijk include techniques by Coron *et al.* [14, 15] and Cheon *et al.* [9]. These systems, however, are unattractive to implement from a practical engineering perspective. For example, the system of van Dijk *et al.* [55] encrypts one bit at a time and, as is true for other homomorphic systems, requires a cumbersome circuit translation that encodes the algebraic relationships among the inputs. To encrypt multiple bits, these systems depend on Gentry's complex bootstrapping idea. By contrast, DoubleMod is attractive for its simplicity, even if it does not accomplish its hope for lower bandwidth expansion.

Defined by its encryption relation $f$, DoubleMod adds random multiples of two secret integers $u$ and $v$ to the plaintext. DoubleMod has *bounded ring-homomorphicity*, meaning that it preserves a bounded number of additions and/or multiplications expressed as a bounded-depth circuit (many authors call this property "somewhat homomorphic;" we prefer our more descriptive terminology). Unbounded ring-homomorphicity would be nicer, but given the maximum depth of such circuits in advance, the parameters of DoubleMod

[1]We avoid the ambiguous term "*fully homomorphic*." Some authors have used this term to mean that *both* addition and multiplication are preserved. Others have used this term to mean that arbitrary polynomials can be evaluated.

[2]CCA1 refers to a limited ("lunchtime") adaptive chosen-ciphertext attack [1, 24], in which the adversary can adaptively query a decryption oracle for a limited period of time. By contrast, in a CCA2 attack, the adversary has unlimited access to the decryption oracle.

can be chosen to accommodate any circuit of depth up to the specified maximum. DoubleMod is remarkably simple and fast: Encryption uses just two integer multiplications and two additions, while decryption is accomplished solely by two modular reductions.

Defined by its encryption relation $g$, SingleMod adds to the plaintext random multiples of a secret integer $u$. SingleMod is *field-homomorphic*, meaning that it is ring-homomorphic and, for any suitable plaintext $x$, $1/x = g^{-1}(g(1/x))$. When implemented over the integers, SingleMod is even simpler and faster than DoubleMod, but SingleMod over the integers can be trivially broken. We attempted to find an alternative algebraic setting in which SingleMod might offer significant cryptographic strength, exploring non-Euclidean number fields, where GCD computations appear to be difficult. Although this strategy might work for some other cryptosystem, we found an chosen-ciphertext attack that breaks SingleMod even in non-Euclidean fields, without computing GCDs.

As *randomized* encryption systems [25, 45], our proposals inherit some advantages (*e.g.*, cryptographic strength—including that separate encryptions of the same plaintext yield different ciphertexts) and disadvantages (*e.g.*, bandwidth expansion) of this class of encryption systems.

Skeptics may rightfully wonder whether such simple transformations can be secure. Indeed, as is also true for all previously proposed ring-homomorphic systems over the integers, with a known-plaintext attack, breaking DoubleMod is at most as hard as solving the AGCD problem—which perhaps might better be called the *Noisy GCD* problem. There are promising lattice techniques for AGCD (*e.g.*, Chen [8]).

Even worse, we present a refinement of a 2012 CCA1 chosen-plaintext attack by Tang that breaks DoubleMod and all known integer homomorphic systems, by finding the secret key without solving AGCD. This attack uses a decyption oracle to facilitate a binary search foreshadowed in 1978 by Rivest, Adleman, and Dertouzos [43]. On the other hand, insecurity against CCA1 attack is not surprising in light of the fact that most homomorphic systems are at best CPA secure.

Insecurity against a chosen-ciphertext attack is a notably undesirable property (especially when the attack finds the secret key), but there might possibly be some applications (*e.g.*, outsourced computations in the cloud where the adversary does not have access to a decryption oracle) for which this weakness is not necessarily a showstopper. Neither Gentry's homomorphic system [20], its variation by Gentry and Halevi [22], nor the system of Brakerski and Vaikuntanathan [7], is secure against such chosen-ciphertext attack. Furthermore, no homomorphic system can be secure against a CCA2 chosen-ciphertext attack that simply decrypts ciphertexts: Given any ciphertext $y_1 = f(x_1)$, using one oracle call to decrypt a ciphertext different from $y_1$, the adversary can compute the unknown plaintext $x_1$ as $x_1 = f^{-1}(y_1 + y_2) - x_2$, where $(x_2, y_2)$ is any matching plaintext-ciphertext pair $y_2 = f(x_2)$.

Another limitation of DoubleMod is that operations on ciphertext further expand the ciphertext length. As noted in Section 5.4, this expansion leaks some information about which algebraic operations are performed. Van Dijk *et al.*'s system does not have these limitations.

Although our security assessments of DoubleMod and SingleMod are mostly negative (especially against CCA1 chosen-ciphertext attack), the simple, natural, and intriguing structure of our systems deserves examination, and we hope that our analyses will be helpful to others in the understanding and pursuit of practical secure homomorphic systems.

Contributions of our paper include:

- Presentation and initial analysis of two randomized secret-key encryption systems called DoubleMod and SingleMod.

- DoubleMod has bounded ring-homomorphicity. In comparison with previous integer homomorphic systems, DoubleMod is simpler and supports encryption of multiple bits.
- SingleMod is field-homomorphic. SingleMod is weak when implemented over integers. Nevertheless, it is an intriguing proposition to base cryptography in non-Euclidean number fields where GCD computations appear to be difficult, and it is instructive to learn from our unsuccessful attempt to do so for SingleMod.
- New CCA1 chosen-ciphertext key-finding attacks against DoubleMod, SingleMod, and the integer homomorphic system of van Dijk *et al.* [55]. Each attack exploits the piecewise linearity of the encryption function by applying a decryption oracle to facilitate a binary search for short vectors that form a nearly orthonormal basis for an underlying lattice of low dimension. These attacks do not need GCD calculations.

The rest of this paper explains and analyzes our encryption systems. Section 2 motivates homomorphic encryption by analogy with "glove box" operations. Section 3 reviews additional previous and related work. Section 4 introduces the DoubleMod system with bounded ring-homomorphicity. Section 5 discusses the security of DoubleMod, including a new powerful chosen-ciphertext attack. Section 6 suggests some parameter choices for DoubleMod. Section 7 introduces the field-homomorphic SingleMod system. Section 8 revisits SingleMod in alternative algebraic settings (*e.g.*, non-Euclidean number fields) for the purpose of finding an alternative setting in which it might have significantly higher levels of cryptographic strength. Section 8 also presents a new chosen-ciphertext attack against SingleMod. Section 9 summarizes our findings and outlines some open problems.

We assume the reader is familiar with the basics of cryptology and number theory, as explained, for example, by Stinson [51], Goldreich [23, 24], Bach-Shallit [5], Hardy & Wright [27], and Niven *et al.* [38]. Our companion paper [49] provides a short tutorial for cryptologists on non-Euclidean number fields and related topics.

## 2 Homomorphic Encryption as "Glove Box" Operations

Homomorphic encryption is an information-processing analogue of the physical "glove box" [18], a device enabling humans to manipulate lethal viruses and other biohazards safely. With homomorphic encryption, untrusted data processing entities can carry out computations by operating on encrypted data without gaining access to associated plaintexts. Whereas a physical glove box aims to protect the handler from exposure to biohazards, a homomorphic digital glove box aims to protect the privacy of manipulated data from an untrusted handler. For example, in cryptographic voting systems using homomorphic encryption, an election authority can verifiably tally encrypted ballots without learning or revealing how anyone voted. Similarly, with homomorphic encryption, processing services in the cloud can analyze a database of medical records without revealing any personal patient information. The more *kinds* of operations can thus be supported, and the more *efficiently* and *securely* they can be implemented, the more useful and interesting the digital glove box becomes.

***Efficiency.*** To be practical, a homomorphic digital glove box should introduce at most reasonable performance penalties. Thus, operations on encrypted data should suffer at most a polynomially-bounded performance penalty versus performing the corresponding operations on plaintext. We shall refer to this requirement as our "efficiency requirement." In addition, the encryption and decryption costs should be polynomially bounded, as should be the costs of transporting the data to and from the glove box user.

***Security.*** For a digital glove box to be secure, we would like it to be super-polynomially difficult for any attacker to break the encryption by determining plaintexts or secret keys, starting from information to which the attacker might reasonably be expected to gain access, including known-plaintext and chosen-ciphertext attacks. Unfortunately, we present chosen-ciphertext attacks that break DoubleMod, SingleMod in non-Euclidean fields, and other recently proposed plain-integer homomorphic encryption methods ([55] and their derivatives).

***Kinds of Operations.*** The more types of operations a digital glove box supports, the more useful it can be, but there are fundamental limits on what kinds of operations can be securely supported. It has been recognized since the 1970s that some public-key encryption systems based on elliptic curve groups ([34]) or modular exponentiation ([44]) enjoy group-homomorphic properties. In 1978, Rivest, Adleman, and Dertouzos [43] observed that secure group-homomorphic encryption is impossible, if encryption permits both efficient *comparisons* and access to encrypted arbitrary constants, because this capability would allow binary search on plaintext values. Thus, while the untrusted entity can perform addition (and we shall show certain other operations such as multiplication can be adjoined to the repertoire), we cannot permit "if-then-else" branching.

## 3 Additional Previous and Related Work

Our DoubleMod relation $f$ grew out of Phatak's [40] recent research on *Residue Number Systems (RNSs)* [35],[3] in which ring-homomorphic properties follow from the fact that addition and multiplication are performed component-wise, independently in each of the modular channels. We informally view the relation $f$ as a dual construction of a two-moduli RNS system. Initially, Phatak and Sherman thought of SingleMod, and then devised DoubleMod as a more secure alternative for the plain integers. DoubleMod represents a different approach from our earlier method to approximate oblivious data processing in cloud computing, based on splitting the moduli in an RNS system across multiple cloud service providers [41].

After Phatak and Sherman discovered the relation $f$ in spring 2011, we subsequently became aware of the independent work of van Dijk *et al.* [55], who apply Gentry's lattice techniques to a simple relation $E$ to construct an unbounded ring-homomorphic relation. Their relation $E$, which they use to encrypt one bit, is the special case of our DoubleMod $f$ with $u = 2$ and $v = p$, where $p$ is a large prime integer. Assuming an unproven computational hypothesis (the sparse subset sum hardness assumption of the bootstrapped scheme), van Dijk *et al.* prove that breaking their system is as hard as solving AGCDs—but only under a limited adversarial model that does not permit chosen ciphertext attacks. Van Dijk *et al.* also review three prior variants of their relation mentioned by others. Some consider van Dijk *et al.*'s single-bit system part of the "cryptographic folklore;" Halevi [26] mentioned it in his 2011 tutorial. Pisa *et al.* [39] discovered DoubleMod independently from us.

The reductions of van Dijk *et al.* can also be viewed as a CCA1 chosen-ciphertext key-finding attack against their system. They show that distinguishing plaintexts in their ciphertext-only model implies breaking AGCD, and that breaking AGCD implies a key-finding attack on their system. Because a CCA1 oracle can distinguish plaintexts in their ciphertext-only model, their reductions establish a CCA1 key-finding attack on their system.

Cohn and Heninger [13] analyze a multivariate generalization of the Howgrave-Graham [31] algorithm for computing AGCDs. Chen and Nguyen [8] present some faster algo-

[3]In a RNS, each integer is represented as a sequence of residues modulo some set of pairwise co-prime moduli.

rithms. These lattice algorithms appear to be the currently best known *known-plaintext* attacks against our and van Dijk *et al.*'s cryptosystems.

In early December 2012, the UMBC group learned of and refined a powerful *chosen-ciphertext* attack against DoubleMod by Tang (see Section 5), which does not solve AGCD. This attack also breaks the integer system of van Dijk *et al.* [55]. Independently, Zhang [57, 58], developed separate attacks on van Dijk *et al.*, which also do not solve AGCD.

In 2011, Gentry and Halevi [22] proposed a variation of Gentry's method, which bears some similarities with SingleMod. Loftus [32] *et al.* show that this system is not IND-CCA1 secure but that a modification of it is. In 2013, Blass *et al.* [16] devised a system based on the hidden modular group order and showed that it is IND-CPA secure; their system too has similarities with DoubleMod.

Vaikuntanathan [54] surveys recent results in homomorphic encryption. For additional references, see Goldwasser, Kalai, and Popa [48].

## 4 The DoubleMod Relation $f$

We define a randomized symmetric-key encryption relation $f$ and its decryption mapping $f^{-1}$ and show that $f$ is ring-homomorphic (preserves addition and multiplication). In this system, $f$ encrypts each plaintext by adding to it random multiples of secret integers $u$ and $v$. Decryption removes this random noise by moding out by $v$ and then $u$.

The parameters $v > u$ must satisfy certain bounds to be explained, and these bounds depend on the "complexity" $K$ (to be defined) of the multivariate polynomial computed by glove box users on encrypted data. The complexity $K$ is related to the maximum permitted total degree $\kappa$ of this polynomial, which establishes an upper bound on the number of permitted cascaded multiplications.

We represent each plaintext message as an integer in $\mathbb{Z}_R$, where $R$ is any large integer. Thus, each plaintext can be represented with at most $\lfloor \lg R \rfloor + 1$ bits.[4] Each ciphertext is an integer in $\mathbb{Z}_T$ where the integer $T$ depends on the system parameters (see Section 6).

Let $u > R^2$ be any large prime, and let $v > u$ be any integer whose smallest prime factor exceeds $u$. These conditions force $v$ to be relatively prime to $u$.

We define the encryption relation $f \subseteq \mathbb{Z}_R \times \mathbb{Z}_T$ in terms of an associated function $F : \mathbb{Z}_R \times \mathbb{S} \to \mathbb{Z}_T$, where $\mathbb{S} = \mathbb{Z}_{R_a} \times \mathbb{Z}_{R_b}$ denotes the set of possible random choices made by $f$, where $R_a > 2$ and $R_b > 2$ are bounds on the random choices $a$ and $b$, respectively, to be explained. For any $x \in \mathbb{Z}_R$, and for any $(a, b) \in \mathbb{S}$, let

$$(x, F(x, a, b)) \in f, \tag{1}$$

where

$$F(x, a, b) = x + au + bv. \tag{2}$$

The relation $f$ defines a randomized encryption system, say to be used from Alice to Bob. To encrypt any plaintext $x$, Alice selects $(a, b) \in \mathbb{S}$ at random; computes $f(x) = F(x, a, b)$; and discards $(a, b)$. The secret key is $(u, v)$.

To decipher any ciphertext $y \in \mathbb{Z}_T$, Bob computes

$$f^{-1}(y) = (y \bmod v) \bmod u. \tag{3}$$

The injectivity of $f$ follows from the facts that $x < u$ and the fact that (due to demands about $v$ we have not yet described) $x + au < v$.

[4] lg denotes the function $\log_2$.

A cost of the homomorphic and randomized properties is that $f$ expands the message length significantly: each ciphertext requires up to $\lfloor \lg T \rfloor + 1$ bits.

To prevent an adversary from exhaustively trying all possible values of $a$ and $b$, we require that they be drawn from a sufficiently large set. In Section 6 we suggest parameter values for $R$, $R_a$, $R_b$ and calculate the resulting ciphertext length.

We now prove that, with appropriate parameter choices, $f$ is ring-homomorphic. We begin with the case $\kappa = 1$, when only one glove box multiplication is permitted. Next, we handle the case of more than one glove box multiplication. Finally, we outline strategies by which the communicants can establish an appropriate parameter $\kappa$.

Earlier we had considered the possibility of reducing the size of ciphertexts by moding out $F$ by $m = uv$. Assuming factoring $m$ is computationally infeasible, one could share the modulus $m$ with the glove box user enabling her to reduce the size of her computations by working in $\mathbb{Z}_m$. We strongly oppose this possibility: working in $\mathbb{Z}_m$ can invalidate the injectivity condition, and exposing $m$ facilitates certain attacks, including attacks that attempt to compute $u$ (see Section 5.2) and possibly lattice attacks.

However, a variation of DoubleMod that mods out by $m = vw$ (where $w > v$ is another secret prime) may be worthy of consideration.

## 4.1 DoubleMod with at Most One Glove Box Multiplication

We assume the following condition to ensure the injectivity of $f$.

*Injectivity Condition 1:* $u > R^2 \geq 4$ and $v > [R_M(u + 1)]^2$, where $R_M = \max\{R, R_a\}$.

Proposition 1 proves that $f$ is ring-homomorphic.

**Proposition 1.** *Let $f$, $R$, $u$, $v$, $R_a$, $R_b$, $R_M$ be defined as above, and assume Condition 1. For any $x_1, x_2 \in \mathbb{Z}_R$, and for any random choices made by $f$, $x_1 + x_2 = f^{-1}(f(x_1) + f(x_2))$ and $x_1x_2 = f^{-1}(f(x_1)f(x_2))$.*

**Proof.** Let $x_1, x_2 \in \mathbb{Z}_R$ be any plaintexts. First, we prove the additive condition. By definition, $f(x_1)+f(x_2) = (x_1+x_2)+(a_1+a_2)u+(b_1+b_2)v$, where $a_1, a_2 \in \mathbb{Z}_{R_a}$ and $b_1, b_2 \in \mathbb{Z}_{R_b}$ are the random choices made by $f$. It suffices to show that $(x_1 + x_2) + (a_1 + a_2)u < v$ and $x_1 + x_2 < u$. First, since $R^2 < u$ and $R \geq 2$, it follows that $x_1 + x_2 < R^2 < u$. Second, by Condition 1, $(x_1 + x_2) + (a_1 + a_2)u < u + R_M^2 u = (R_M^2 + 1)u < v$. Consequently, $[(f(x_1) + f(x_2)) \bmod v] \bmod u = [(x_1 + x_2) + (a_1 + a_2)u] \bmod u = x_1 + x_2$.

Second, we prove the multiplicative condition. By definition of $f$, $f(x_1)f(x_2) = x_1x_2 + (a_1x_2 + a_2x_1)u + a_1a_2u^2 + v(\text{remaining terms})$. It suffices to show that $x_1x_2 + (a_1x_2 + a_2x_1)u + a_1a_2u^2 < v$ and $x_1x_2 < u$. First, $x_1x_2 < R^2 < u$. Second, by Condition 1, $x_1x_2 + (a_1x_2 + a_2x_1)u + a_1a_2u^2 < R_M^2 + 2R_M^2 u + R_M^2 u^2 = R_M^2(u + 1)^2 < v$. Therefore, $[f(x_1)f(x_2) \bmod v] \bmod u = [x_1x_2 + (a_1x_2 + a_2x_1)u + a_1a_2u^2] \bmod u = x_1x_2$. **Q.E.D.**

## 4.2 DoubleMod with More Than One Glove Box Multiplication

To handle more than one glove box algebraic operation, we require a constant upper bound on the number of such operations to be specified. From this bound, we now derive lower bounds on $u$ and $v$ sufficient to permit the specified number of operations to be performed ring-homomorphically. In Section 4.3 we describe methods to establish the specified bound.

***About polynomial degree.*** The glove box user via $+$, $-$, and $\times$ operations will necessarily compute a multivariate polynomial function of her inputs. Let $\kappa$ be the total degree of this polynomial. For example degree$(xy^2z^3 + 3x^5) = 6$.

Unfortunately, $\kappa$ can be as large as $2^j$ for a computation involving $j$ multiplications—consider repeated squaring. For a *depth-1* arithmetic circuit with $w$ internal "wires" interconnecting $+$, $-$, and $\times$ gates (unboundedly large fan-in and fan-out permitted), the degree obeys $\kappa \le w$ and this bound is achievable. More generally, for a *depth-d* (feed-forward only) arithmetic circuit with $w$ internal wires, the degree obeys $\kappa \le (w/d)^d$ and this bound is achievable.

It is this potential *exponential* degree-growth with glove box circuit depth that causes DoubleMod to disobey our efficiency requirement *i.e.*, operations on encrypted data can suffer exponentially great performance penalty versus the corresponding sequence of operations on plaintext data.

However, for *bounded-depth* circuits, wherein $d$ is guaranteed never to exceed some constant, $\kappa$ enjoys an upper bound that is a *polynomial* function of the circuit size (described by the number $w$ of internal wires). Indeed, if $d$ is small, the bound $\kappa$ is a polynomial of low degree, which as we shall see causes our efficiency requirement to be fully satisfied. This enjoyable situation could still happen even for some (but not all) unbounded-depth circuits; the "interval" and "simulation" approaches described in Section 4.3 exploit benefits of all such cases.

***Definition of "complexity" K.*** For any depth-$d$ (feed-forward only) arithmetic circuit with $w$ internal wires, define its "complexity" to be $K = (w/d)^d$.

***Bounds on $u$ and $v$ in terms of $K$.*** We now derive bounds on $u$ and $v$ sufficient for $f$ to stay ring-homomorphic up through any glove box computation with complexity $K$. Generalizing Condition 1, these bounds ensure that the DoubleMod encryption function $f$ remains injective. Assume the following.

*Injectivity Condition 2:* $u > R^{K+1}$ and $v > [R_m(u + 1)]^{K+1}$.

The requirement $u > R^{K+1}$ ensures that $u$ will exceed the product of any $K+1$ plaintexts (using $K$ multiplications). Additions could also be involved but are less destructive than are multiplications.

Consider the expressions that can result from $K$ cascaded multiplications of encrypted plaintexts. The second part of Condition 2 ensures that $v$ will be larger than the sum of all the terms in the expansion of $(x + au + bv)^{K+1}$ that do not contain $v$ as a factor. Letting $\zeta = x + au$, these terms of interest are those in the binomial expansion of $(\zeta + bv)^{K+1}$ that do not contain $v$ as a factor—namely, $\zeta^{K+1}$. Therefore, $v > [R_m(u + 1)]^{K+1}$ ensures the desired bound.

Thus, the number of bits required to represent a ciphertext computed with complexity $K$ is approximately $\lg v > (K + 1)\lg R_M + (K + 1)^2 \lg R$. This polynomially-bounded bit-count expansion yields a polynomially-bounded runtime expansion, verifying our efficiency requirement.

## 4.3 Establishing Bounds

The *encryptor* and *glove box user* can negotiate bounds on $u$ and $v$ using a variety of methods. Underlying each method is the essential requirement, for unique decipherability, that the encrypted value $x + au + bv$ stay within the "permitted region" defined by $\overline{x} < u$ and $\overline{x} + \overline{au} < v$, where $\overline{x}$ is an upper bound on $x$, and $\overline{au}$ is an upper bound on $au$.

We now describe three methods for negotiating bounds. These methods differ mainly in who performs certain bound calculations when. Alternatively, the encryptor could calculate and tell the glove box user bounds on the number of multiplications and additions that may be performed.

***On-Line Interval Method.*** Along with the encrypted data, the encryptor provides to the glove box user $R$, $R_a$, $R_b$, and interval bounds $[u_L, u_H]$ and $[v_L, v_H]$ for $u$ and $v$, respectively. As she manipulates the encrypted data, the glove box user also performs "interval arithmetic" to keep track of upper bounds on the magnitudes of her calculations. While these computed bounds stay within the permitted region defined above, the homomorphic property remains valid (the glove box user is guaranteed that DoubleMod decryption will restore the correctly-manipulated plaintext values). However, if the glove box user does too many (or a bad choice of) arithmetic operations, causing her interval bounds to extend outside the permitted region, then it is possible that decryption will instead produce garbage. In that case, the glove box user could *complain* to the encryptor, who could respond by re-encrypting using suitably-larger $u$ and $v$, hopefully large enough for the glove box user then to be able to complete whatever computation she wanted to do. If not, a further complaint could be issued. Because the interval arithmetic can likely be performed using single-precision floating point arithmetic, the computational cost of the interval arithmetic will likely be negligible.

***Pre-Computed Simulation Method.*** Before receiving any encrypted data, the glove box user *simulates* her desired calculations, performing the aforementioned interval calculations without the encrypted data. If the simulation terminates successfully, then the glove box user could announce that the proposed $u$ and $v$ intervals would be adequate for her needs; otherwise, she could complain preemptively.

***Pre-Specified Ciruit Bounds.*** The glove box user first informs the encryptor what sort of computation she will be doing, for example by providing degree bounds, operation-count bounds, or arithmetic-circuit size and depth bounds. The encryptor uses these bounds to compute how large his $u$ and $v$ need to be, so that the glove box user will be assured of never leaving the permitted region. The encryptor finds suitable $u$ and $v$ and encrypts the data using them.

## 5 Attacks on DoubleMod

We now discuss some security properties of the relation $f$. After summarizing our adversarial model, we discuss some attack strategies, beginning with a CCA1 chosen-ciphertext attack by Tang (as refined by Phatak) for determining the secret key. We also discuss known-plaintext attacks, including lattice attacks to solve the *Approximate Greatest Common Divisor (AGCD)* problem [8, 13].

### 5.1 Adversarial Model

The main goal of the attacker is to determine the secret key $(u, v)$. We assume the adversary can mount adaptive chosen-plaintext and chosen-ciphertext attacks, requesting the ciphertexts of some limited number of plaintexts of her choice, and requesting the plaintexts of some limited number of ciphertexts of her choice. We say the system is secure if all such successful attacks require at least $\Omega(2^\lambda)$ steps, where $\lambda$ is a security parameter.

## 5.2 A Chosen-Ciphertext Attack that Finds $u$

In December 2012, Tang[5] suggested the basic ideas of the following chosen-ciphertext attack, which finds part of the key, $u$, with at most $1 + \lg u$ calls to a decryption oracle. It does so without solving AGCD. We now present a refinement of this attack.

First, by exhaustive seach using chosen plaintexts $2^1, 2^2, 2^3, \ldots$, find the smallest positive integer $i$ such that $f^{-1}(2^i) \neq 2^i$; let this minimal value be denoted $t$. Since $2^i < u < v$ implies $f^{-1}(2^i) = 2^i$, it follows that $2^t > u > 2^{t-1}$. It follows that $2^t \bmod u = 2^t - u$. This linear search requires at most $\lg u$ calls to the decryption oracle. If the adversary already knows the approximate bit length of $u$ (as Tang had originally assumed), then this search takes even fewer steps.

Second, given any known and matching plaintext-ciphertext pair $(x_1, y_1)$, where $y_1 = x_1 + a_1 u + b_1 v$, the adversary computes the difference $\delta_1 = y_1 - x_1 = a_1 u + b_1 v$. Next, the attacker asks for a decryption of $y = 2^t + \delta_1$ and suppose that it is $x$. Then, $u = 2^t - x$ because

$$\begin{aligned} x &= [(2^t + \delta_1) \bmod v] \bmod u = [(2^t + a_1 u + b_1 v) \bmod v] \bmod u \qquad (4) \\ &= 2^t - u. \end{aligned}$$

Section 5.3 shows how to find $v$.

## 5.3 A Chosen-Ciphertext Attack that Finds $v$ Given $u$

Tang also outlined how to find the rest of the secret key, $v$, from $u$. The idea is to represent $v$ as a polynomial in $u$, and to use the decryption oracle as a means of comparing plaintext values to permit binary searches for the coefficients of this polynomial. The task is an easy version of a knapsack problem similar to attacks on iterated knapsacks [3]. We now present a slightly refined version of this process.

First, by exhaustive search using chosen ciphertexts $u^2, u^3, u^4, \ldots$, find the smallest positive integer $i$ such that $f^{-1}(u^i) \neq 0$; let this minimal value be denoted $k$. Since $u^i < v$ implies $f^{-1}(u^i) = 0$, it follow that $u^k > v > u^{k-1}$. Therefore, $v$ can be represented as a polynomial in $u$ of degree at most $k-1$; *i.e.*, $v = c_0 + c_1 u + c_2 u^2 + \cdots + c_{k-1} u^{k-1}$, for some nonnegative integers $c_0, c_1, \ldots, c_{k-1}$ less than $u$. This "forward pass" search for $k$ requires at most $\log_u v$ calls to the decryption oracle.

Second, in a "reverse pass," determine the coefficients $c_j$ one at a time, in decreasing value of $j$, from $j = k-1$ downto $j = 0$. For each $c_j$, perform a binary search in the interval $[0, u-1]$. At each step of this binary search, let $C_L$ denote the lower endpoint of the current search interval, and let $C_H$ denote the upper endpoint. Let $C_M = \lfloor (C_L + C_H)/2 \rfloor$ denote the midpoint. Initially, $C_L = 0$ and $C_H = u - 1$.

We perform $k$ "phases;" one for each value of $j$, where $0 \leq j < k$. In Phase $j$, we will determine the coefficient $c_{k-j}$. For each $j$, at the start of Phase $j$, we have already determined the coefficient of each term with degree higher than $k - j$. Let this known "suffix" $\hat{v}_j$ of $v$'s polynomial be $\hat{v}_j = c_{(k-j)+1} u^{(k-j)+1} + \cdots + c_{k-1} u^{k-1}$. We seek the largest integer $c_{k-j}$ such that $c_{k-j} u^{k-j} + \hat{v}_j \leq v$. To this end, let $y_j = c_{k-j} u^{k-j} + \hat{v}_j$, and let $x_j = f^{-1}(y_j)$. It is true that $x_j = 0$ if and only if $c_{k-j} u^{k-j} + \hat{v}_j \leq v$.

Thus, in each step of Phase $j$ of the binary search for $c_{k-j}$, let $y_j = C_M u^{k-j} + \hat{v}_j$ and $x_j = f^{-1}(y_j)$. If $x_j = 0$, then continue the search in the new interval $[C_M, C_H]$; otherwise, if $x_j \neq 0$, then continue the search in the new interval $[C_L, C_M]$.

[5]Personal correspondence dated December 6, 2012.

In the reverse pass, there are $k$ phases, each of which requires at most $\lg u$ calls to the decryption oracle. Therefore, the total number of oracle calls in this attack is $\log_u v + k \lg u$. In the binary search, the oracle calls are adaptive in that the inputs to subsequent calls depend on the answers returned from previous calls.

Because the underlying cryptographic transformation in van Dijk *et al.* [55] is a special case of DoubleMod (with $u = 2$ and $v = p$, where $p$ is a large integer), this attack also breaks their system.

Finally, we note that adding a third secret prime-multiple addend ($cw$) in the encryption computation of what we call TripleMod ($x + au + bv + cw$) would not mitigate the security limitations of DoubleMod, as the attack described here would still apply. More generally, if any number of additive secrets of this form are used, our attack can "peel off" these secrets one-at-a-time, starting with the smallest value.

Even if in practice the enemy does not have access to a decryption oracle, we feel that these chosen-ciphertext attacks reflect poorly on the security on DoubleMod (and the system of van Dijk *et al.*): the existence of weaknesses under chosen-ciphertext attack increases the likelihood that weaknesses also exist under other attacks.

## 5.4 Some Approaches for Known-Plaintext Attacks

A generic known-plaintext attack (that subsumes several other attacks) is to compute differences

$$\delta_i = f(x_i) - x_i = b_i v + a_i u \tag{5}$$

for some $1 \leq i \leq t$ known or chosen plaintexts $x_i$, and then to apply an algorithm (*e.g.*, Chen and Nguyen [8]) for solving the resulting AGCD problem to determine $v$, as we will explain in Section 5.5. In this attack, for each $1 \leq i \leq t$, let $r_i = a_i u$ denote the "perturbations" of the multiples $b_i v$ of $v$. Once $v$ is determined, it is easy to compute $u$, because $u = \gcd(r_i, r_j)$, for any $i \neq j$, where $r_i = \delta_i \bmod v$, and $r_j = \delta_j \bmod v$.

The attacker could, in principle, search over all possible values of $v$ and test each candidate choice by checking the consistency of $\gcd(r_i, r_j)$, for $i \neq j$, with a sufficient number of differences $\delta_i$. Thus, the system offers only computational (and not information-theoretic) security. It is important that $v$ be large enough to make this attack computationally infeasible.

It is important that the adversary be unable to determine the coefficients $a_i$ or $b_i$. (When we said we wanted them to be "random uniform" and "discarded," we meant it.) For example, suppose the adversary learns the $a_1, a_2, a_3, a_4$ used to compute $\delta_1 = b_1 v + a_1 u$; $\delta_2 = b_2 v + a_2 u$; $\delta_3 = b_3 v + a_3 u$; and $\delta_4 = b_4 v + a_4 u$. Then $(a_2\delta_1 - a_1\delta_2) = (a_2 b_1 - a_1 b_2)v$ and $(a_4\delta_3 - a_3\delta_4) = (a_4 b_3 - a_3 b_4)v$, when $v = \gcd(a_2\delta_1 - a_1\delta_2, a_4\delta_3 - a_3\delta_4)$. Similarly, given $b_1, b_2, b_3, b_4$, the adversary could compute $u$. Therefore, it is important that $R_a$ and $R_b$ be chosen large enough to make it infeasible for the adversary to determine their values by exhaustive search.

A slight disadvantage of DoubleMod is that it leaks some information about the algebraic operations performed by the glove box user: multiplications expand ciphertext length more than do additions. In this sense, and unlike van Dijk *et al.*'s method, DoubleMod does not achieve Gentry's [19] notion of "circuit privacy."

Repeatedly encrypting some special value (*e.g.*, zero) offers no advantage over using Equation 5. Moreover, and unlike some other methods (*e.g.*, [55]), in our system no special steps are needed to encrypt zero.

## 5.5 Lattice Attacks Computing Approximate GCDs

In this section we explain the approximate GCD problem, review the best known algorithm for solving it, and explain how it can be applied to attack $f$. In Section 6 we make recommendations for system parameters that aim to render this attack computationally infeasible.

A general version of the approximate GCD problem can be summarized as follows, as explained by Cohn and Heninger [13].

**Problem AGCD**. *Given $t$ approximate multiples of an integer $p$ which are of the form $\alpha_i = q_i p + r_i$, for $i = 1, \cdots, t$, where the "perturbations" $r_i$ satisfy the upper bounds $|r_i| \leq X_i$, find $p$, which is the GCD of the $\alpha_i$ with the perturbations removed.*

In 2011 (after the work of van Dijk *et al.* [55]), Cohn and Heninger [13] analyzed a multivariate generalization of the Howgrave-Graham [31] algorithm for computing AGCDs, yielding the following observation. We now summarize this heuristic observation, which Cohn and Heninger call "Heuristic Theorem 2."

**Semi-Empirical observation about computing approximate common divisors.** *Let $N$ and $t$ be any positive integers, and let $\alpha_1, \cdots, \alpha_t$ be any positive integers such that, for all $1 \leq i \leq t$, $\alpha_i \approx N$. Let $X$ and $\beta$ be bounds (that may depend on $N$) such that, as $N$ grows asymptotically, $\beta \gg 1/\sqrt{\log N}$ and $\lg X < (C_l + O(1))\beta^{t/(t-1)} \lg N$, where, $C_l \approx 1 - (\log t)/t$. For any fixed integer $l$, it is possible to find in polynomial time all $r_1, \cdots, r_t$ such that $\gcd(\alpha_1 - r_1, \cdots, \alpha_t - r_t) \geq N^\beta$ and $|r_i| \leq X$.*

We can apply this heuristic to break $f$ as follows. (1) Given any $t$ matching plaintext-ciphertext pairs $(x_i, y_i)$, for each $1 \leq i \leq t$, calculate the difference $\delta_i = y_i - x_i = b_i v + a_i u$, as explained in Equation 5. (2) For each $1 \leq i \leq t$, regard $\delta_i$ as a multiple $b_i v$ of the secret integer $v$ to which a perturbation $r_i = a_i u$ is added. (3) Apply Cohn and Heninger's heuristic observation to determine the secret $v$. (4) For each $1 \leq i \leq t$, compute $r_i = \delta_i \bmod v$. (5) Compute $u = \gcd(r_i, r_j)$, for any $i \neq j$.

Thus, under a known-plaintext attack, breaking $f$ is at most as hard as solving AGCD.

## 6 Recommended Parameter Choices

We now recommend some parameter choices. In any cryptographic system, it is a crucial engineering decision to select parameter values that yield an acceptable level of cryptographic strength while not unduely increasing the time and memory costs to encrypt plaintext and to represent the resulting ciphertext. In each of the proposed ring-homomorphic systems, one of the major costs is bandwidth expansion—the increase in the number of bits required to represent each ciphertext beyond those required to represent the corresponding plaintext.

Following commonly-used notation (*e.g.*, [55], [13], [15]), we shall adopt the following symbols to denote certain important bit lengths:

$\gamma$ denotes the length of each ciphertext in bits; that is, for any plaintext $x$, $\gamma = \lfloor \lg f(x) \rfloor + 1$.

$\eta$ denotes the length of $v$ in bits; that is, $\eta = \lfloor \lg v \rfloor + 1$. ($v$ corresponds to the solution to the AGCD problem.)

$\rho$ denotes the length of the perturbation term $r_i = a_i u$; that is, $\rho = \lfloor \lg r_i \rfloor + 1 = \lfloor \lg(a_i u) \rfloor + 1$.

We will now suggest values for the parameters $R$, $R_a$, $R_b$, $\gamma$, $\eta$, and $\rho$. We will do so consistently for the choice of security parameter $\lambda = 72$, as selected by van Dijk *et al.* [55]. That is, we will select parameters with the goal of requiring the adversary to expend at least $\Omega(2^{72})$ work to determine the secret key $(u, v)$. Using $\lambda = 72$ enables us easily to compare the bandwidth expansion of our system with those of other ring-homomorphic systems for comparable levels of cryptographic strength. We recognize, however, that this choice is near the edge of what many adversaries may be able to compute today and hence does not offer any significant margin of safety for the future. A similar decision process could be made for larger security parameters. In what follows, we shall assume that the number of cascaded multiplications is $\kappa = 1$.

Let $R = 2^{64}$, which means that each plaintext is a 64-bit integer. Select $R_a = 2^{72}$, so that an exhaustive search of the coefficient space for the $a$'s would require $\Omega(2^{\lambda})$ steps. These choices imply $R_m = \max(R, R_a) = 2^{72}$. Condition 1 then implies the constraints $u > R^2 = 2^{128}$ and $v > [(R_M + 1)u]^2 > [(2^{72} + 1)2^{128}]^2 \approx 2^{400}$. Thus, the bit-length of $v$ is $\eta \approx 400$ and the bit-length of the perturbation term $a_i u$ is $\rho \approx (128 + 72) + 1 = 200$.

To select the parameter $R_b$ which bounds the size of the coefficients $b$, we follow a process used by van Dijk *et al.* [55, p. 15], which aims to ensure that the cryptanalytic effort will be at least $2^{\lambda}$. Setting $\gamma/\eta^2 = \lambda = 72$ yields $\gamma = \eta^2\lambda \approx 400^2(72) = 11,520,000 \approx$ 11.5 Mb (megabits) $\approx$ 1.4 MB (megabytes).[6] Thus, the bit-length of $b$ is approximately $\lg R_b \approx \gamma - \eta = 11,520,000 - 400 = 11,519,600 \approx 11.5$ Mb.

For these parameter choices our method needs only approximately 1.4 MB $\approx$ 11.5 Mb of ciphertext when encrypting one 64-bit block of plaintext. Therefore, our system uses approximately $11,520,000/64 = 180,000 = 180$ Kb $= 22.5$ KB of ciphertext per encrypted plaintext bit.

The practical bandwidth expansion of DoubleMod (with multiple cascaded multiplications) is worse than suggested by these calculations because operations on encrypted data further expand the ciphertext length. Previous systems with bootstrapping do not have this limitation.

For possible additional strength, we suggest that the parameters be further adjusted in two ways to violate the hypotheses of Cohn and Heniger's heuristic observation from Section 5.5. First, their observation requires that, for all $1 \le i \le t$, $\alpha_i \approx N$. The $\alpha_i$ in Theorem 2 correspond to the terms $b_i v$ in $f$. Varying the length of the coefficients $b_i$ will cause this hypothesis to be violated. For example, one might vary the bit-length of the $b_i$ from about 11.5 Mb to 23 Mb.

Second, their observation also requires that $\beta \gg 1/\sqrt{\log N}$ to find $v \ge N^{\beta}$. For our suggested parameter choices, since $\lg N \approx \gamma$, the inequality $v \ge N^{\beta}$ implies $\beta \le (\lg v)/\lg N \approx \eta/\gamma \approx 400/11,520,000 \approx 0.000034722$. By contrast, $1/\sqrt{\log N} \approx 1/\sqrt{11,520,000} \approx 1/3394.1125 \approx 0.0002946$. Thus, for our suggested parameter choices, the heuristic observation does not apply. We recognize, however, that violating the hypotheses of the heuristic observation does not imply that the resulting cryptanalytic problem is hard.

## 7 Introducing the SingleMod Encryption System

We now introduce a simple and intuitive randomized secret-key encryption system, which we call *SingleMod*. In this section we discuss its plain-integer version, which is the first system we considered. The insecurity of this plain-integer version led us to devise the related

[6] 1 KB = $10^3$ bytes, 1 MB = $10^6$ bytes, and 1 GB = $10^9$ bytes, where 1 byte = 8 bits. Similarly, 1 Kb = $10^3$ bits, 1 Mb = $10^6$ bits, and 1 Gb = $10^9$ bits.

DoubleMod system. In Section 8, we revisit SingleMod in alternative algebraic settings (including non-Euclidean algebraic number fields, where GCD computations appear hard), where we had hoped there might exist higher levels of security.

In the plain version of SingleMod, we represent each plaintext message $x$ as an integer in $\mathbb{Z}_u$, where $u$ is any large secret prime. The secret key is $(u, v)$, where $v > u$ is a larger secret prime. We publicize the product $m = uv$. The encryption relation $g \subseteq \mathbb{Z}_u \times \mathbb{Z}_m$ is defined by

$$g(x) = (x + au) \bmod m, \tag{6}$$

whenever $x \in \mathbb{Z}_u$, where $a$ is chosen uniformly at random from $\mathbb{Z}_m$. After encrypting, the sender Alice discards $a$. To decipher any ciphertext $y \in \mathbb{Z}_m$, the receiver Bob simply computes

$$g^{-1}(y) = y \bmod u. \tag{7}$$

***Insecurity.*** An attacker can determine the secret key using a GCD calculation in a variety of ways. Given $m$ and a plaintext-ciphertext difference $\delta = y - x$, the attacker can compute $u = \gcd(\delta, m)$. With access to a few plaintext-ciphertext pairs $(x_j, y_j)$, the attacker can compute $u$ as the GCD of the differences $\delta_j = y_j - x_j$. If she instead had access to two ciphertexts $y_1$, $y_2$ arising from the same unknown plaintext $x$, then $u = \mathrm{GCD}(y_2 - y_1, m)$. Finally, an attacker armed with a quantum computer could efficiently deduce $(u, v)$ by factoring $m$.

***Field Homomorphism.*** Let $x_1, x_2, x \in \mathbb{Z}_u$. One may readily verify $(x_1 + x_2) \bmod u = g^{-1}(g(x_1) + g(x_2) \bmod m)$ and $(x_1 x_2) \bmod u = g^{-1}(g(x_1)g(x_2) \bmod m)$. Let "$(1/x) \bmod u$" denote the multiplicative inverse of $x$ modulo $u$. We have $(1/x) \bmod u = g^{-1}((1/g(x)) \bmod m)$ subject to certain caveats. First, 1/0 does not exist. This issue arises in every field and is not really an issue. Second, $1/x \bmod m$ fails to exist if $x > 0$ is divisible by either $u$ or $v$. This is a genuine problem because it can prevent the glove box user (*i.e.*, the entity operating on encrypted data) from dividing by $x$, even though if she were operating on plaintext she could have performed that division. However, the only possible situations in which this failure could occur have $\mathrm{GCD}(x, m)$ being a nontrivial factor (*i.e.*, $u$ or $v$) of $m$. Thus, the user will never experience this kind of failure unless she could crack the cryptosystem by factoring $m$ anyhow, which we assume is so hard that this event will not happen during the lifetime of the universe. So for practical purposes $g$ is field homomorphic.

***Efficiency.*** With SingleMod, encryption and operating on encrypted data are very fast, requiring only a very small number of elementary modular operations. SingleMod encryption requires only one multiplication, one addition, and one modular reduction. Decryption takes only one modular reduction. Performing addition on encrypted data requires only one addition and one modular reduction, and the latter can be implemented as "subtract $m$ if value $\geq m$" with only constant-factor slowdown. Performing multiplication on encrypted data requires only a multiplication and a modular reduction, again with only constant factor slowdown (division with remainder takes at most a constant times the amount of time required to multiply integers [4]). Finally, evaluating a multiplicative inverse $1/x$ on encrypted data can be done via an extended GCD computation (*e.g.*, Bach & Shallit [5]) for $x$ and $m$, whereas for plaintext it would be done for $x$ and $u$. Using algorithms such as Schönhage-Strassen multiplication and Schönhage's fast extended GCD algorithm [4], these operations run in $O(N(\log N)^c)$ steps on $N$-bit data for various constants $c < 2.01$, *i.e.*, optimal speedup to log factors.

## 8 SingleMod Revisited in Alternative Algebras

We now revisit the SingleMod encryption system in alternative algebras, with the ultimate goal to find a suitable algebraic setting in which GCD computations are difficult, thereby enhancing the security of SingleMod. First, to explore the essential algebraic structures needed for SingleMod, we consider the Gaussian and Eisenstein Integers. Second, we consider non-Euclidean quadratic number fields, where GCD computations seem difficult. Also, we briefly consider necessary requirements to avoid possible factorization-based attacks. Although we found a chosen-ciphertext attack against SingleMod in non-Euclidean fields (see Section 8.4), we hope others may benefit from our explorations into the intriguing possibility of basing cryptography on the difficulty of computing GCDs in certain non-Euclidean fields.

### 8.1 SingleMod Using Gaussian or Eisenstein Integers

SingleMod can be implemented in a variety of algebraic settings, including for example, the Gaussian or Eisenstein complex 2-dimensional "integers" rather than plain integers.[7] If one would prefer the underlying field's multiplication operation to be noncommutative (these more commonly are called "division algebras"), then one could use the (4-dimensional, $D_4$ lattice) Hurwitz quaternions. For a noncommutative and also nonassociative multiplication, one could use the (8-dimensional, $E_8$ lattice) "integral Cayley octaves" octonions. Each of these algebras has a multiplicative nonnegative-integer-valued "norm," "mod" operation, efficient GCD algorithm, and unique factorization theorem (up to certain equivalence operations—see Rehm [42] and Coan and Perng [11]).

Abstractly, the essential structural ingredient that makes SingleMod work is any ring containing a hard-to-find subfield. In each example above, the reason the subfield is hard to find is because of the assumption that factoring integers is hard. For example, it is not difficult to show that factoring quaternions is of equivalent hardness to factoring ordinary integers since their integer norms automatically also get factored.[8] In each of these cases, however, the existence of an efficient GCD algorithm makes breaking the system easy for any attacker with access either to plaintext-ciphertext pairs, or to two ciphertexts arising from the same plaintext.

### 8.2 In Search of a More Secure SingleMod in Non-Euclidean Quadratic Number Fields

Since the insecurity of plain-integer SingleMod stems from the existence of efficient GCD algorithms, it is natural to consider the possibility of implementing SingleMod in an underlying number field that has no efficient GCD algorithm. This section summarizes our initial thoughts along this quest. We begin with a review of some relevant algebra.

SingleMod readily generalizes to arbitrary algebraic number fields; for simplicity, here we consider only quadratic fields. Hardy and Wright [27] give a nice introduction to quadratic number fields, though they use the word "Euclidean" to describe what we nowadays call "norm-Euclidean." For example, $Q[\sqrt{D}]$ denotes the field of rational numbers $Q$

[7]The "Gaussian integers" are $Z[i]$ where $i = \sqrt{-1}$. The "Eisenstein integers" are $Z[\rho]$ where $\rho = e^{2\pi i/3} = (-1 + \sqrt{-3})/2$.

[8]*E.g.*, use the well known fact that any integer $n > 0$ can be efficiently represented as a sum of four squares $n = a^2 + b^2 + c^2 + d^2$ (Rousseau [46]), then to factor $n$, factor the quaternion $a + ib + jc + kd$. The other hardness direction is trivial.

extended by adjoining the irrational $\sqrt{D}$ and then closing under field operations. If $D < 0$ this is an "imaginary quadratic field" but if $D > 0$ it is "real." The "integer" subring within this field, which we shall denote $Z[\sqrt{D}]$, consists (by Theorem 238 in [27]) of the numbers $x + y\sqrt{D}$ if $D \equiv 2$ or 3 (mod 4) and of the numbers $x + (\sqrt{D} - 1)(y/2)$ if $D \equiv 1$ (mod 4), where $x, y$ are integers. The "norm" of a field element $\xi = r + s\sqrt{D}$, where $r$ and $s$ are rational, denoted $N(\xi)$, is given by $N(\xi) = r^2 - Ds^2$. This norm is "multiplicative," *i.e.*, $N(\xi)N(\gamma) = N(\xi\gamma)$. For imaginary fields $N(\xi) = |\xi|^2 \geq 0$ but for real fields it can have either sign.

A normed field F is "norm-Euclidean" if for $\alpha, \beta \in F$ there exist $q, r \in F$ with $\alpha = \beta q + r$ with $|N(r)| < |N(\beta)|$. Clark [10] defines it to be "Euclidean" if this is true with some other integer-valued "fake norm" function $\phi(x)$ substituted for the usual multiplicative norm $|N(x)|$; this fake norm must obey $\phi(0) = 0$ and $\phi(x) > 0$ if $x \neq 0$. Each such field has a Euclid-style GCD algorithm running in time bounded by a polynomial function of the bit lengths of the input field elements.

It is known that $Q[\sqrt{D}]$ is norm-Euclidean (for a squarefree integer $D$) exactly for these 21 cases

$$D \in \{-11, -7, -3, -2, -1, 2, 3, 5, 6, 7, 11, 13, 17, 19, 21, 29, 33, 37, 41, 57, 73\}. \tag{8}$$

There are also at least two examples ($Z[\sqrt{69}]$ and $Z[\sqrt{14}]$) of Euclidean, but not norm-Euclidean, quadratic fields—see Clark [10], Niklasch [37], and Harper [28, 29].

All Euclidean number fields enjoy "unique factorization into primes" theorems. But an infinite set of quadratic number fields (namely those with class number > 1) have nonunique factorization: for example, $6 = 2\cdot 3 = (1+\sqrt{-5})(1-\sqrt{-5})$ in $Z[\sqrt{-5}]$, and $81 = 3\cdot 3\cdot 3\cdot 3 = (5+2\sqrt{-14})(5-2\sqrt{-14})$ in $Z[\sqrt{-14}]$. Interestingly, there are many quadratic number fields that do not have a norm-Euclidean GCD algorithm but still enjoy unique factorization. For example, it is known (from the Heegner-Stark-Baker Theorem [50] and Williams [56]) that, for for $D < 0$, $Q[\sqrt{D}]$ has unique factorization but no Euclidean GCD algorithm (whether based on the norm or any fake-norm) exactly in the cases $D \in \{-19, -43, -67, -163\}$. For $D > 0$, there is a conjecturally-infinite set of $D$ beginning

$$14, 22, 23, 31, 38, 43, 46, 47, 53, 59, 61, 62, 67, 69, 71, 77, 83, 86, 89, 93, 94, 97, \ldots, \tag{9}$$

for which we have unique factorization but no norm-Euclidean algorithm.

Behrbohm and Redei [6] show that a real quadratic field can enjoy unique factorization only if the radicand $D > 0$ has at most two different prime factors. The Cohen-Lenstra [12] heuristics predict that about 75.45% of prime $D$ yield unique factorization, a prediction confirmed by te Riele and Williams [53] through computer investigations. It has been conjectured that each of these unique-factorable real cases is Euclidean for appropriate fake-norm functions, though as far as we know, this has been proven only for $D = 14$ and 69. So far, those fake-norms have been difficult to find and use.

Decryption requires the existence of a "mod $u$" operation, and one is readily defined (and efficiently computable) by subtracting away the closest point within the "multiples of $u$" sublattice of the 2D lattice that is the number field. It is well known that finding nearest lattice points is fast and easy for 2D lattices [47]. We also need to be able to find prime field elements $u$ quickly. That is easily done using Theorem 241 of Hardy and Wright [27] that $u$ is prime in a quadratic number field if $|N(u)|$ is a plain-integer prime.

For those who want a noncommutative multiplication, one can consider algebraic extensions of the quaternions. As a start in that direction, Fitzgerald [17] recently determined every "norm-Euclidean quaternion order."

As cryptologists, we now face several puzzles:

1. That the norm-Euclidean GCD algorithm fails to exist might not rule out the existence of some other kind of efficient GCD algorithm. Indeed, Kaltofen and Rolletschek [33] construct a GCD algorithm in any quadratic number field (even ones with non-unique factorization), with running time bounded by a polynomial function of the number of bits describing the field-elements, provided $D$ is fixed. But their running time grows proportionally to a power of $D$, *i.e.*, exponentially in the number of bits in $D$. Therefore, to provide security, the underlying quadratic number field must have a discriminant $D$ that is a polynomially-large number of bits long.
2. It might be that only some quadratic fields yield security—in which case we want to know which ones, how common they are, how to find them quickly, and how to estimate their security levels.
3. There might still be some efficient way to break SingleMod even without having any GCD algorithm, and as we show in Section 8.4, this is true under a chosen-ciphertext attack.

We do not know the answers to puzzles 1–2.

## 8.3 Seeking Immunity to Factoring Attacks

The original plain-integer version of SingleMod can be broken by factoring $m = uv$. If somebody built a quantum computer or found an efficient integer-factoring algorithm, SingleMod (and other cryptosystems including RSA) would fall to factoring attacks. It would be interesting to explore the possibility of finding a suitable algebraic setting in which SingleMod might survive such attacks.

In 1989, Kaltofen and Rolletschek [33] showed that polynomial-time integer factoring would automatically yield polynomial-time algorithms for factoring in arbitrary quadratic number fields with unique factorization. Hence, to achieve immunity against integer factoring, it is necessary to work in fields of degree greater than quadratic, or featuring highly-non-unique factorization, or both.

## 8.4 A Chosen-Ciphertext Attack on SingleMod in Non-Euclidean Number Fields

We now summarize Smith's chosen-ciphertext attack against *SingleMod in non-Euclidean number fields (NESM)*. Like Tang's algorithm which finds the DoubleMod key without solving AGCD, Smith's algorithm finds the SingleMod key without computing GCDs. Smith devised this algorithm in early January 2013, originally as a possible attack against the 2011 bounded ring-homomorphic system of Gentry and Halevi [22].

Both the *Gentry-Halevi (GH) system* and NESM can be viewed in the context of lattices. In GH, there is a (high) $n$-dimensional lattice, which creates an "ideal lattice" by regarding each lattice point as the vector of coefficients of a high-degree univariate polynomial modulo $x^n - 1$. These polynomials form a ring. The secret key is a "nice" (*e.g.*, near-orthonormal) basis for the lattice. The plaintext is a not-too-long, not-too-short integer $n$-vector. To encrypt, add a random lattice vector. To decrypt, subtract the nearest lattice vector. This system can be viewed as based on the "cyclotomic" algebraic number field based on the $n$th root of unity, and is of high algebraic degree $n$.

By contrast, NESM is based on a low-degree algebraic number (degree 2 to 5) and hence involves a lattice of dimension at most 5. Each response from the decryption oracle returns a member of the "multiples of $u$" principal ideal, which may be regarded as a sublattice of the entire algebraic number ring. The low dimension of NESM gives SingleMod unbounded field homomorphicity, whereas the high dimension of GH may offer some increased cryptographic strength. Generally speaking, lattice problems are hard only in high dimensions. Furthermore, in low dimension lattices, it is known how to find nearest vectors in polynomial time.

Smith's probabilistic algorithm finds a nice basis for the underlying lattice. Both Tang's algorithm and Smith's algorithm exploit the facts that DoubleMod and SingleMod are piecewise linear in some appropriate ring or field.

**Smith's Algorithm**

1. Find the shortest nonzero vector, denoted $L_1$, in the lattice. Do so by decrypting random short vectors, expanding their length via "binary search," until the nearest lattice vector is no longer 0.
2. For $k = 2, 3, \dots, n$, find the shortest nonzero vector not in the sublattice generated by the preceding vectors $L_1, L_2, \dots, L_{k-1}$. Do so by decrypting short random vectors in the subspace orthogonal to the aforementioned sublattice.
3. The vectors $L_1, L_2, \dots, L_n$ found in this way are probably a fairly nice lattice basis.

We believe there are many ways to extend Smith's algorithm to perform other oracle-aided searches.

## 9 Conclusion

We have proposed and analyzed SingleMod and DoubleMod—two new symmetric secret-key randomized encryption systems that enjoy homomorphic properties: DoubleMod has bounded ring homomorphicity; SingleMod is field homomorphic. Each system can be implemented in a variety of algebraic settings. Although each system suffers from substantial security weaknesses (plain-integer SingleMod can be easily broken with GCD calculations, and DoubleMod can be broken under a CCA1 chosen-ciphertext attack that finds the secret key), there is value in understanding the attractive properties and limitations of these simple, fast, intuitive systems as part of the quest to develop practical homomorphic encryption transformations.

DoubleMod has a close relationship with the AGCD problem, and SingleMod has a close relationship with the GCD problem. Coarsely speaking, DoubleMod is at most as hard as AGCD, and SingleMod is at most as hard as GCD. The complexity of AGCD remains unknown, but in light of known lattice techniques, many cryptologists consider AGCD a risky problem on which to base cryptographic strength. Over the plain integers, GCD is easily computed in logarithmic time (*e.g.*, by Euclid's algorithm), but in certain non-Euclidean algebraic number fields with non-unique factorization, there is no known efficient GCD algorithm. For these reasons, we returned to SingleMod seeking a possible algebraic setting for which SingleMod might achieve higher levels of cryptographic strength. Unfortunately, as explained in Section 8.4, we discovered a variation of the chosen-ciphertext attack against DoubleMod that also works against SingleMod in non-Euclidean fields.

Although we were unable to find a secure version of SingleMod, it remains an intriguing open problem to find new cryptographic functions that derive their security from the apparent difficulty of computing GCDs in certain non-Euclidean number fields.

The precise relationships between breaking DoubleMod and solving AGCD, and between breaking SingleMod and solving GCD, are sensitive to the precise assumptions of the adversarial model. For example, under a known-plaintext attack, breaking DoubleMod is at most has hard as AGCD. Under an adaptive chosen-ciphertext attack, breaking DoubleMod appears strictly much easier than AGCD. And under a ciphertext-only attack, it may be that breaking DoubleMod is harder than AGCD. There are similar relationships between SingleMod and GCD.

The CCA1 chosen-ciphertext attack against DoubleMod also defeats the integer ring-homomorphic system of van Dijk *et al.* [55] and its derivates [36], [15]. This attack, which does not appear to solve AGCD, does not necessarily contradict the proof by van Dijk *et al.* that breaking their system is as hard as solving AGCD, since their adversarial model does not permit chosen-ciphertext attacks. It is an open problem to work out these relationships more precisely with mathematical proofs. In particular, the security proofs of van Dijk *et al.* depend on transmitting one bit at a time, whereas DoubleMod encrypts a block at a time.

Currently we are exploring ideas suggested by Warren Smith for a possible bounded-error polynomial-time quantum algorithm for solving AGCD, given an oracle for computing DoubleMod. The approach is to compute the DoubleMod function with plaintext 0 using the quantum superposition of the random integers $a, b$ (wihtout revealing these random choices to the adversary), and then to compute the period (i.e., the secret key $u$) of this approximately periodic sequence by applying a quantum Fourier transform.

Chosen-ciphertext attacks can be very powerful attacks. The attack against DoubleMod discovers the key by making calls to a decryption oracle to facilitate a binary search. In particular, the attack exploits magnitude comparisons implicit in the remaindering operation over the integers. It appears that variations of this attack will work in any algebraic setting in which magnitude comparisons (or equivalent norm operations) are possible in the encrypted domain. It is an open question whether there are general security limitations of all ring-homomorphic systems under such chosen-ciphertext attacks.

DoubleMod suffers exponential space and time performance penalties in the worst case (repeated squarings) for manipulating encrypted versus plaintext data. However, if the data-manipulator agrees to perform only computations on encrypted data that are describable by an arithmetic circuit with any fixed-depth bound (we permit unbounded fan-in and fan-out), then each of these performance penalties becomes bounded by a low-degree polynomial. While bounded-depth arithmetic circuits constitute a very limited class of algorithms, they nevertheless might be expressive enough to allow some interesting applications. For example, many neural nets (*e.g.*, for optical character recognition), are describable as bounded-depth feed-forward circuits. More generally, any three-layer network of threshold or sigmoidal components (with unbounded fan-in and unbounded fan-out) can represent any single-valued function [52].

Our work explores two new encryption systems with field-homomorphicity and bounded ring-homomorphicity. While we conclude that each of these intuitive systems has substantial cryptographic weaknesses, our exploration points out useful insights and raises intriguing open questions concerning homomorphic encryption.

## Acknowledgments

We thank Ronald Rivest, Marten van Dijk, Nadia Heninger, Travis Mayberry, Alexander Morrow, Abu Mohammad Omar Shehab Uddin Ayub and reviewers for helpful comments. we also thank Shafi Goldwasser for pointing out reference [55]. Sherman was supported in part by the Department of Defense under IASP Grants H98230-10-1-0359, H98230-11-1-0473, H98230-12-1-0454, and by the National Science Foundation under SFS Grant 1241576.

## References

[1] Chosen-ciphertext attack, Wikipedia (Accessed June 2013).

[2] Homomorphic encryption, Wikipedia (Accessed September 2011).

[3] A. M. Odlyzko, The rise and fall of knapsack cryptosystems, in: (ed.), Cryptology and Computational Number Theory, vol. 42, AMS, 1990, Proceedings Symposium Applied Mathematics.

[4] A. Aho, J. Hopcroft, J. Ullman, Design and Analysis of Computer Algorithms, Addison-Wesley, 1974.

[5] E. Bach, J. Shallit, Algorithmic Number Theory, Volume 1: Efficient Algorithms, MIT Press, 1996.

[6] H. Behrbohm, L. Redei, Der Euklidischen Algorithmus in quadratischen Körpern, Journal für die Reine und Angewandte Mathematik 174 (1936) 192–205.

[7] Z. Brakerski, V. Vaikuntanathan, in: P. Rogaway (ed.), Advances in Cryptology—Crypto 2011, vol. LNCS 6841, Springer-Verlag.

[8] Y. Chen, P. Q. Nguyen, Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers, in: Pointcheval, Johansson (eds.), Advances in Cryptology: Eurocrypt 2012, vol. LNCS 7237, Springer, 2012.

[9] J. H. Cheon, J.-S. Coron, J. Kim, M. Lee, T. Lepoint, M. Tibouchi, A. Yun, Batched fully homomorphic encryption over the integers, in: Johansson, Nguyen (eds.), Advances in Cryhptology: Eurocrypt 2013, vol. LNCS 7881, Springer, 2013.

[10] D. A. Clark, A quadratic field which is Euclidean but not norm-Euclidean, Manuscripta Mathematica 83 (3–4) (1994) 327–330.

[11] B. Coan, C. Perng, Factorization of Hurwitz quaternions, International Mathematical Forum 7 (43) (2012) 2143–2156.

[12] H. Cohen, Hermite and Smith normal form algorithms over Dedekind domains, Mathematics of Computation 65 (216) (1996) 1681–1699.

[13] H. Cohn, N. Heninger, Approximate common divisors via lattices, `http://eprint.iacr.org/2011/437.pdf` (August 2011).

[14] J. Coron, A. Mandal, D. Naccache, M. Tibouchi, Fully homomorphic encryption over the integers with shorter public keys, in: P. Rogaway (ed.), Advances in Cryptology—Crypto 2011, vol. LNCS 6841, Springer-Verlag, 2011.

[15] J. Coron, D. Naccache, M. Tibouchi, Public key compression and modulus switching for fully homomorphic encryption over the integers, in: Advances in Cryptology: Eurocrypt 2012, 2012.

[16] T. M. Erik-Oliver Blass, Agnes Hui Chan, J. Trostle, Towards practical somewhat homomorphic encryption, College of Computer and Information Sceince, Northeastern University (November 2013).

[17] R. W. Fitzgerald, Norm-Euclidean quaternionic orders, Integers 12 (2) (2012) 197–208.

[18] C. Gentry, A fully homomorphic encryption system, PhD dissertation, Department of Computer Sceince, Stanford University (September 2009).

[19] C. Gentry, Fully homomorphic encryption using ideal lattices, in: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, ACM, 2009.

[20] C. Gentry, Computing arbitrary functions of encrypted data, Communications of the ACM 53 (3) (2010) 97–105.

[21] C. Gentry, Toward basing fully homomorphic encryption on worst-case hardness, in: H. Gilbert (ed.), Advances in Cryptology—Crypto 2010, Springer-Verlag, 2010.

[22] C. Gentry, S. Halevi, Implementing Gentry's fully-homomorphic encryption scheme, `http://eprint.iacr.org/2010/520.pdf` (February 2011).

[23] O. Goldreich, Foundations of Cryptography, Volume I: Basic Tools, Cambridge University Press, 2001.

[24] O. Goldreich, Foundations of Cryptography, Volume II: Basic Applications, Cambridge University Press, 2004.

[25] S. Goldwasser, S. Micali, Probabilistic encryption, Journal of Computer and System Sciences 28 (2) (1984) 270–299.

[26] S. Halevi, Homomorphic encryption tutorial, http://people.csail.mit.edu/shaih/pubs/Homomorphic-Encryption.tutorial.pdf.

[27] G. Hardy, E. Wright, An Introduction to the Theory of Numbers, Oxford Science Publications, 1979.

[28] M. Harper, $Z[\sqrt{14}]$ is Euclidean, Canad. J. Maths. 56 (1) (2004) 55–70.

[29] M. Harper, M. R. Murty, Euclidean rings of algebraic integers, Canad. J. Maths. 56 (1) (2004) 71–76.

[30] B. Hayes, Alice and Bob in cyber space, American Scientist 100 (September-October 2012) 362–367.

[31] N. Howgrave-Graham, Approximate integer common divisors, in: Cryptography and Lattices, vol. LNCS 2146, Springer, 2001.

[32] J. Loftus, A. May, N. P. Smart, and F. Vercauteren, in: A. Miri and S. Vaudenay (ed.), Proceedings of the 18th international conference on Selected Areas in Cryptography (SAC 2011), vol. LNCS 7118, Springer-Verlag.

[33] E. Kaltofen, H. Rolletschek, Computing greatest common divisors and factorizations in quadratic number fields, Maths. of Computation 53 (188) (1989) 697–720.

[34] N. Koblitz, Ellipitc curve cryptosystems, Mathematics of Computation 48 (177) (1987) 203–209.

[35] I. Koren, Computer Arithmetic Algorithms, 2nd ed., A. K. Peters, Ltd., Natick, MA, 2001.

[36] D. Micciancio, A first glimpse of cryptography's Holy Grail, Communications of the ACM 53 (3) 96.

[37] G. Niklasch, On the verification of Clark's example of a Euclidean but not norm-Euclidean number field, Manuscripta Mathematica 83 (3-4) (1994) 443–446.

[38] I. Niven, H. S. Zuckerman, H. L. Montgomery, An Introduction to the Theory of Numbers, 5th ed., John Wiley, 1991.

[39] M. A. Pedro Silveira Pisa, O. C. M. B. Duarte, Somewhat homomorphic encryption scheme for arithmetic operations on large integers, in: (ed.), Global Information Infrastructure and Networking Symposioum (GIIS), 2012, IEEE Explore, 2012, .

[40] D. S. Phatak, RNS-ARDSP: A novel, fast residue number system using approximate rational domain scaled precomputations, UMBC CSEE Dept. Technical Reports TR-CS-10-01 through TR-CS-10-03 (October 2010).

[41] D. S. Phatak, A. T. Sherman, J. Pinkston, A new paradigm to approximate Oblivious Data Processing (ODP) for data confidentiality in cloud computing, in: Proceedings of the IEEE Seventh World Congress on Services (Services 2011), IEEE, 2011, paper 4065, 8 pages.

[42] H. P. Rehm, Prime factorization of integral Cayley octaves, Ann. Fac. Sci. Toulouse Math 6 (2,2) (1993) 271–289.

[43] R. Rivest, L. Adleman, M. Dertouzos, On data banks and privacy homomorphisms, in: Foundations of Secure Computation, 1978, pp. 169–178.

[44] R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 36 (2) (1978) 120–126.

[45] R. L. Rivest, A. T. Sherman, Randomized encryption techniques, in: Advances in Cryptology: Proceedings of Crypto 82, Plenum Press, 1983.

[46] G. Rousseau, On a construction for the representation of a positive integer as the sum of four squares, L'Enseignement Math. 2 (33,3-4) (1987) 301–306.

[47] C. Schnorr, A hierarchy of polynomial time lattice basis reduction algorithms, Theoretical Computer Science 53 (2–3) (1987) 201–224.

[48] R. A. P. V. V. N. Z. Shafi Goldwasser, Yael Kalaim, Reusable garbled circuits and succinct functional encryption, MIT CSAIL (March 2013).

[49] W. D. Smith, A. T. Sherman, D. Phatak, A short tutorial for cryptologists on abstract algebra, algebraic number fields, non-unique factorization, and GCDs, 18 pages (June 2013).

[50] H. Stark, On the gap in the theorem of Heegner, Journal of Number Theory 1 (1969) 16–27.

[51] D. R. Stinson, Cryptography: Theory and Practice, 3rd ed., Chapman & Hall/CRC, Boca Raton, FL, 2006.

[52] T. Sasao, OR-AND-OR three-level networks, in: T. Sasao and M. Fujita (ed.), Representations of Discrete Functions, Kluwer Academic Publishers, 1996.

[53] H. te Riele, H. Williams, New computations concerning the Cohen-Lenstra heuristics, Experimental Mathematics 12 (1) (1999) 99–113.

[54] V. Vaikuntanathan, Computing blindfolded: New developments in fully homomorphic encryption, Dept. of Computer Science, University of Toronto, `http://www.cs.toronto.edu/~vinodv/FHE-focs-survey.pdf` (2011).

[55] M. van Dijk, C. Gentry, S. Halevi, V. Vaikuntanathan, in: Advances in Cryptology: Eurocrypt 2010, vol. LNCS 6110, Springer-Verlag.

[56] K. S. Williams, Note on principal ideal domains, 48 (May-June) (1975) 176–177.

[57] Z. Zhang, T. Plantard, W. Susilo, On the CCA-1 security of somewhat homomorphic encryption over the integers, Information Security Practice and Experience (2012) 353–368.

[58] Z. Zhang, T. Plantard, W. Susilo, Reaction attack on outsourced computing with fully homomorphic encryption schemes, Information Security and Cryptology-ICISC 2011 (2012) 419–436.

## Contents

1 Introduction 2

2 Homomorphic Encryption as “Glove Box” Operations 4

3 Additional Previous and Related Work 5

4 The DoubleMod Relation $f$ 6
- 4.1 DoubleMod with at Most One Glove Box Multiplication . . . . . . . . . . 7
- 4.2 DoubleMod with More Than One Glove Box Multiplication . . . . . . . . 7
- 4.3 Establishing Bounds . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

5 Attacks on DoubleMod 9
- 5.1 Adversarial Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
- 5.2 A Chosen-Ciphertext Attack that Finds $u$ . . . . . . . . . . . . . . . . . . 10
- 5.3 A Chosen-Ciphertext Attack that Finds $v$ Given $u$ . . . . . . . . . . . . . . 10
- 5.4 Some Approaches for Known-Plaintext Attacks . . . . . . . . . . . . . . . 11
- 5.5 Lattice Attacks Computing Approximate GCDs . . . . . . . . . . . . . . 12

6 Recommended Parameter Choices 12

7 Introducing the SingleMod Encryption System 13

8 SingleMod Revisited in Alternative Algebras 15
- 8.1 SingleMod Using Gaussian or Eisenstein Integers . . . . . . . . . . . . . 15
- 8.2 In Search of a More Secure SingleMod in Non-Euclidean Quadratic Number Fields . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
- 8.3 Seeking Immunity to Factoring Attacks . . . . . . . . . . . . . . . . . . . 17
- 8.4 A Chosen-Ciphertext Attack on SingleMod in Non-Euclidean Number Fields 17

9 Conclusion 18

References 20