The Uneasy Relationship Between Mathematics and Cryptography

Neal Koblitz

During the first six thousand years—until the invention of public key in the 1970s—the mathematics used in cryptography was generally not very interesting. Well into the twentieth century cryptographers had little use for any of the concepts that were at the cutting edge of mathematics. Indeed, mathematicians looking at cryptography in those years might have found justification for Paul Halmos' infamous title "Applied Mathematics Is Bad Mathematics."

There were some exceptions. In the 1940s Alan Turing, the father of computer science, worked extensively in cryptography and, in particular, showed how to use sophisticated statistical techniques to crack a code; and Claude Shannon, the father of information theory, worked on the foundations of cryptography.

In the same decade G. H. Hardy wrote in *A Mathematician's Apology* that "both Gauss and lesser mathematicians may be justified in rejoicing that there is one science [number theory] at any rate, and that their own, whose very remoteness from ordinary human activities should keep it gentle and clean." In Hardy's day most applications of mathematics were military, and as a pacifist he was pleased that number theory was studied not for its practical uses, but only for its intrinsic aesthetic appeal.

This image of number theory as "gentle and clean" took a big hit in 1977 when three computer scientists at the Massachusetts Institute of Technology—Ron Rivest, Adi Shamir, and Len Adleman—invented a radically new cryptographic system. An article in *Scientific American* by Martin Gardner described the RSA idea, explained its significance, and caused a sudden upsurge in popular interest in both cryptography and number theory.

In those years RSA was the most important way to achieve what came to be called "public key cryptography". Earlier systems for scrambling messages worked well in military or diplomatic applications, where there was a fixed hierarchy of people who were authorized to know the secret keys. But by the 1970s, with large sections of the economy rapidly becoming computerized, the limitations of classical cryptography were coming to the fore. For example, suppose that a large network of banks wants to be able to exchange encrypted messages authorizing money transfers. In traditional cryptography any pair of banks must have its own secret set of keys that they agree on and exchange using a trusted courier. The number of possible pairs of banks could easily be in the hundreds of millions. So the earlier type of cryptography, called "private key" (or "symmetric key"), becomes extremely unwieldy.

In public key cryptography, the key needed to scramble a message is public information. Each user of the system (for example, each bank) has its own public key, which is listed in a directory much like someone's phone number. Anybody can encrypt a message using the public key. However, the unscrambling process requires knowledge of a totally different key, which the user keeps secret. The procedure for scrambling a message is called a "trapdoor one-way function". This means that once we look up the bank's public key it is computationally easy (with the help of a computer) for us to send it an encrypted message. If, however,

Neal Koblitz is professor of mathematics at the University of Washington, Seattle. His email address is koblitz@math.washington.edu.

This article is based on an Invited Address given at the AMS meeting at the Stevens Institute of Technology in Hoboken, NJ, on April 14, 2007. Parts of it are taken from the cryptography chapter of his forthcoming book Random Curves: Journeys of a Mathematician*, to be published by Springer-Verlag.*

we want to go the other way—unscramble the message—this is computationally infeasible unless we possess an additional bit of information, namely the secret key.

Rivest, Shamir, and Adleman devised a clever—but also simple—way to make a trapdoor one-way function using elementary number theory. Their construction is based on multiplication of two large prime numbers p and q to get a composite number $N = pq$. One has to assume that this is a one-way process in the sense that factoring N to get p and q is very hard.

Thus, the security of RSA cryptography was entirely dependent on the presumed difficulty of factoring large integers. For this reason the invention of RSA gave a tremendous stimulus to the study of methods to factor integers, as well as methods to generate large random primes. During the early 1980s the highlights of mathematical cryptography were for the most part in this area—for example, Carl Pomerance's development of improved sieving techniques for index-calculus factorization algorithms, and the Adleman-Pomerance-Rumely deterministic nearly-polytime primality proof by means of Jacobi sums.

In a somewhat different vein Don Coppersmith devised an algorithm that could find discrete logarithms in the multiplicative group of $\mathbb{F}_{2^n}$ in time $\exp\left(n^{1/3+\epsilon}\right)$, which was much faster than earlier index-calculus methods. This also had cryptographic significance, because ElGamal had proposed an alternative to RSA encryption that was based on the presumed difficulty of inverting the function $x \mapsto g^x$ (where g is fixed) in a finite field.

In 1984 Hendrik Lenstra distributed a one-page description of a new method he had developed for factoring large integers using elliptic curves. The clever and elegant algorithm was simple enough that I could understand it from the one-page outline, although a detailed analysis of its running time took many more pages. This was the first time that elliptic curves had been used in cryptography, and when I read the page that Lenstra had sent me I felt that at one stroke he had raised the mathematical sophistication in cryptography to a whole new level.

Shortly after that I left for a semester in the Soviet Union, where no one worked openly on cryptography. I continued to think about the subject, though, and soon it occurred to me that it should be possible to use elliptic curves in an entirely different way from what Lenstra had done, namely, to construct systems based on the hard problem of finding logarithms on the curve. Since I knew no one in the Soviet Union I could talk with about this, I wrote a letter to Andrew Odlyzko, then at Bell Labs, describing my idea for using the elliptic curve group to construct a cryptosystem. Odlyzko was one of the few mathematicians at that time who had done major work in both theoretical and practical areas. Nowadays it's not so unusual to bridge pure and applied mathematics, but in the mid-1980s Odlyzko was unique in this respect among the mathematicians whom I knew personally.

Email didn't yet exist, and letters between the U.S.S.R. and the U.S. took a couple of weeks in each direction. So it wasn't until a month later that I received a reply from Odlyzko. He said that my idea for a new type of cryptography was a good one, and in fact at the same time Victor Miller of IBM was proposing exactly the same thing. The appeal of elliptic curve cryptography (ECC) was that the elliptic curve discrete logarithm problem appeared (and still appears twenty-two years later) to be a substantially more difficult problem than integer factorization.

At first neither Victor nor I imagined that ECC would be of commercial importance; rather, we saw it as a nice theoretical construction to think about. In retrospect, what was surprising was not that I had no notion of commercializing the idea, but that Victor Miller, who worked at IBM, wasn't thinking in practical terms. He didn't even apply for a patent, although then as now IBM's policy was to strongly encourage all its employees to get patents for everything they possibly could, even on the flimsiest of grounds. So the question of turning ECC into a commercial product would wait until other people became interested in it.

After I returned to the U.S., I started attending cryptography conferences. The most important were the annual Crypto meetings held each August in Santa Barbara, California. In the 1980s I found the atmosphere at Crypto to be refreshing and stimulating. It was a truly multidisciplinary meeting, with people from industry, government, and academia in fields ranging from math and computer science to engineering and business.

There was an element of "forbidden fruit" in the first decade of the Crypto conferences. At the beginning of the 1980s the National Security Agency (NSA) had made a heavy-handed (but unsuccessful) attempt to restrict open research in cryptography. Thus, the founding of the Crypto conferences in 1981 was itself an act of defiance.

The free-spirited tone of the meetings in those years reflected the colorful and eccentric personalities of some of the founders of and early researchers in public key cryptography. One such person was Whit Diffie, a brilliant, offbeat, and unpredictable libertarian who in 1976 had coauthored (with Martin Hellman) the most famous paper in the history of cryptography. Diffie used to run the "rump session", where informal, irreverent, and

often humorous presentations were the norm. There was heckling, and at one point Whit had to impose some restrictions on what could be thrown at a speaker (empty beer cans were okay, but not full ones).

The corporate influence was much weaker then. There was a long lag between the invention of public key cryptography and its acceptance in the commercial world; until the late 1980s businesses generally had little interest in the issue of data security. Most researchers in cryptography had never signed a "nondisclosure agreement" limiting what they could say publicly—in fact, most of us had never heard of such a thing.

It was at Crypto where I met Scott Vanstone, a mathematician at the University of Waterloo who led a multidisciplinary group that had implemented improved algorithms for arithmetic in finite fields. With that experience they were well equipped to work on ECC. Vanstone, along with two other Waterloo professors, one in math and one in engineering, formed a company, now called the Certicom Corporation, to develop and market ECC.

Elliptic curves are not the only kind of curves that can be used in cryptography. In 1989, I proposed using the jacobian groups of hyperelliptic curves. In recent years a lot of research, especially in Germany, has been devoted to hyperelliptic curve cryptosystems.

In early September 1998, a few days before I was to leave for a year's sabbatical at the University of Waterloo, I received an email from Joe Silverman, a mathematician at Brown University who had written an excellent two-volume graduate textbook on elliptic curves. His message outlined a new algorithm he was proposing to solve the elliptic curve discrete log problem—in other words, to break elliptic curve cryptography.

Silverman called his algorithm "xedni calculus" because that's "index" spelled backwards. His general idea was to perform steps that are similar to those in index-calculus algorithms, but in the reverse order.

The reason Silverman thought that his algorithm might possibly be efficient was based on a deep and difficult relationship called the Birch and Swinnerton-Dyer Conjecture. Ironically, in a book titled *Algebraic Aspects of Cryptography* that I had published just a few months before, I had included a discussion of this conjecture in a section that I called "Cultural Background". My tone was apologetic to my readers for taking their time with mathematics that, while of great interest to theoreticians, was unlikely, I said, ever to be applied to cryptography. Then within a year I was intensively studying Silverman's attack on ECC that was based precisely on the idea behind that conjecture. This shows that it is unwise to predict that a certain type of mathematics will *never* be used in cryptography.

Scott Vanstone and the others at Certicom were extremely worried about Joe Silverman's algorithm, because they feared that ECC skeptics and competitors—especially people at the RSA company—would seize upon it as an argument against the use of elliptic curves.

The first few months of my sabbatical year were devoted to a thorough analysis of the Silverman algorithm. In October I found a theoretical argument, using the concept of the "height" of points, that showed that for very, very large elliptic curve groups the xedni approach would be extremely inefficient. However, with this general line of reasoning I couldn't be specific about the sizes for which the algorithm would be impractical. It was conceivable, although I thought it unlikely, that the algorithm would not be totally infeasible for curves in the size range that's used in cryptography.

It is important to understand that an asymptotic result—such as my theoretical argument that established the inefficiency of xedni in the limit as the size of the group increases—cannot be relied upon as any kind of guarantee of security. Rather, one must analyze the algorithm for elliptic curves of the size employed in cryptography. The asymptotic argument might be helpful as a guide—and certainly it made us hopeful that we would be able to show that xedni is impractical for the curves used in the real world—but it cannot serve as a substitute for a concrete security analysis. It turned out to be much harder and more time-consuming to carry out this analysis than it had been to come up with the theoretical argument for the asymptotic result.

In order to answer the crucial question of efficiency of xedni for elliptic curves in the practical range, I worked with a multidisciplinary group of young mathematicians and computer scientists at the Centre for Applied Cryptographic Research at Waterloo, especially Edlyn Teske, Andreas Stein, and Michael Jacobson. We were in constant communication with Joe Silverman, who gave us suggestions on how best to test his algorithm. Finally, by mid-December enough computations were in, and Silverman agreed that his algorithm was impractical. In fact, that's an understatement — it turned out that his algorithm was probably the *slowest* one that had ever been thought up to find elliptic curve discrete logarithms.

Nevertheless, it was an elegant idea, and our study of xedni was a stimulating project. Silverman's attempted attack on elliptic curve cryptography illustrated the increasing use of arithmetic algebraic geometry in public key cryptography.

In the 1990s another example of the greater sophistication of mathematical cryptography was the proposal of Gerhard Frey to use Weil descent

to find discrete logs on elliptic curves. Subexponential algorithms for discrete logs on high-genus hyperelliptic curves had already been developed by Adleman and Huang, and Frey's idea was to transfer the discrete log problem on an elliptic curve to one on a high-genus hyperelliptic curve. Frey's proposal was studied by Galbraith, Gaudry, Hess, Menezes, Smart, Teske, and others, and was shown to lead to a faster algorithm in a small number of cases.

Progress was also made in finding very quick methods to count the number of points on a randomly generated elliptic curve. The first step in this direction was taken in a 1985 paper by Schoof, who used division polynomials. Subsequently, better algorithms were devised using modular forms and p-adic techniques.

One indication of the amount of research devoted to cryptographic applications of elliptic curves in recent years is the annual ECC conference series, which is now in its eleventh year (see `http://www.cacr.math.uwaterloo.ca`).

A whole new type of elliptic curve cryptography was developed starting in about 2000, following ideas of Antoine Joux, Dan Boneh, and Matt Franklin. It turned out that the Weil and Tate pairings on elliptic curves could be used to achieve cryptographic functionality that had not been possible before (or had been done inefficiently), notably, identity-based encryption (where one's public key is, say, one's email address) and extra-short digital signatures. Pairing-based cryptography has been an active area of research; in July 2007 the first of a series of conferences devoted entirely to this type of elliptic curve cryptography was held in Japan.

Despite these wonderful examples of applications of interesting mathematics to cryptography, there has also been a downside—in fact, two downsides. That will be the subject of the remainder of this article.

First of all, there has been a bandwagon effect. Once in the 1990s the Canadian Natural Sciences and Engineering Research Council sent me a large proposal to review from a group that was led by a prominent mathematician who claimed that the proposed research would be important for cryptography. After reading the project description, it was clear to me that (1) the proposal was strong from a mathematical standpoint, and (2) they didn't know beans about cryptography. It was sad that some mathematicians seemed to feel pressured into portraying their research as being somehow related to cryptography.

In the late 1980s NSA realized that it had erred in antagonizing the mathematical community several years before, and it wanted to patch up relations. In academia, the best way to mend fences is to give out money. So they set up a system of grants that has become a major source of funding in certain fields, such as number theory.

For the most part it's good when more money comes into mathematics—whatever the motives of the donor. However, there can be subtle negative effects as well. Many years ago William Thurston and others warned us of the dangers of excessive reliance on military funding. And last year in the *Notices* David Eisenbud wrote what I thought was an eloquent rebuttal of the argument (based on the supposed advantages for fund-raising) in favor of an AMS Fellows program.

In the early 1990s I received a proposal for NSA funding for a conference on Drinfeld modules. The conference seemed like a good idea, and my review was generally positive. However, the tone of part of the proposal bothered me. In a section on "the effect of the conference on the competitiveness of American mathematics," the writers had attempted to divide the field between American and "non-American" mathematics and argue for the conference on the grounds that it would increase the competitive standing of the former. I commented:

> Mathematics is perhaps the most international of intellectual disciplines. Interaction and joint work easily cross national boundaries. Thus, it is usually impossible to determine—and serves no useful purpose to try to determine—the proportion of credit to be attributed to each country. Such a chauvinistic tone is not in keeping with the cooperative and international spirit of the mathematical profession…[W]hether they wrote this section out of sincerely felt concern for the "competitiveness of American mathematics" or to cater to what they guessed would be the mindset at NSA, I really hope that in the future they delete such nonsense from project proposals.

Apparently the availability of money from NSA had had a corrupting effect on some mathematicians, who started to think in nationalistic and jingoistic terms so that they could write their proposal in a way that they thought would appeal to NSA.

At the same time that mathematicians were trying to jump on the crypto bandwagon, cryptographers were discovering the power that an aura of mathematical certainty can have in competitive situations. They began to prove mathematical theorems that supposedly guaranteed the security of their system—the idea being to convince outsiders that their system was 100% safe. This is the second "dark side" of the relationship

between math and cryptography that developed as each group was looking for ways to exploit the status of the other group in order to advance its interests. Before explaining this use (or misuse) of mathematics in more detail, I'd like to comment on a clash of research cultures between math and cryptography.

In 1996 I was the program chair of Crypto. To someone trained in mathematics this was an unsettling experience. About two-thirds of the submissions arrived by courier mail within 48 hours of the final deadline. Many had obviously been rushed and were full of typesetting errors. One author had sent me only the odd-numbered pages. A few had violated the requirement of anonymity (there was a policy of double-blind reviews). Several had disregarded the guidelines that had been sent to them. And in many cases the papers had little originality; they were tiny improvements over something the same authors had published the year before or a minor modification of someone else's work.

In some ways the situation has gotten even worse with electronic submissions. Alfred Menezes, the program chair for Crypto 2007, told me that of the 197 submissions, 103 arrived within eleven hours of the deadline and 35 arrived within the very last hour.

Mathematical publishing works differently. In the first place, most articles appear in journals, not conference proceedings—and journals don't have deadlines. In the second place, people in mathematics tend to have a low opinion of authors who rush into print a large number of small articles—the derogatory term is LPU (*least publishable unit*)—rather than waiting until they are ready to publish a complete treatment of the subject in a single article.

Math departments usually believe the

Conjecture. *For the development of mathematics it is better for someone to publish one excellent paper in n years than n nearly worthless papers in one year.*

In certain other fields of science—including, unfortunately, computer science and cryptography—the analogous conjecture, while most likely true, is not widely believed.

Cryptography has been heavily influenced by the disciplinary culture of computer science, which is quite different from that of mathematics. Some of the explanation for the divergence between the two fields might be a matter of time scale. Mathematicians, who are part of a rich tradition going back thousands of years, perceive the passing of time as an elephant does. In the grand scheme of things it is of little consequence whether their big paper appears this year or next. Computer science and cryptography, on the other hand, are influenced by the corporate world of high technology, with its frenetic rush to be the first to bring some new gadget to market. Cryptographers, thus, see time passing as a hummingbird does. Top researchers expect that practically every conference should include one or more quickie papers by them or their students.

In recent years Alfred Menezes and I have written a series of papers that critique the subfield of cryptography known as *provable security*. (See `http://eprint.iacr.org/2004/152.pdf`, `http://eprint.iacr.org/2006/229.pdf`, and `http://eprint.iacr.org/2006/230.pdf`.) Although the papers have been widely downloaded and most of the reaction has been favorable, our work in this area has not been welcomed by everyone. Many specialists in theoretical cryptography have resented our intrusion into their field.

In the 1980s it seemed that all cryptographers were glad to see the influx of mathematicians. Twenty years later, however, I have the impression that some of them wish that we would just go away.

The idea of "provable security" is to give a mathematically rigorous proof of a type of conditional guarantee of the security of a cryptographic protocol. It is *conditional* in that it typically has the form "our protocol is immune from an attack of type X provided that the mathematical problem Y is computationally hard."

Here the word "protocol" means a specific sequence of steps that people carry out in a particular application of cryptography. From the early years of public key cryptography it has been traditional to call two users A and B of the system by the names "Alice" and "Bob." So a description of a protocol might go as follows: "Alice sends Bob…, then Bob responds with…, then Alice responds with…," and so on.

The form that proofs of security take is what is known as a *reduction*. Reductions from one problem to another occur implicitly throughout mathematics; in computer science, reductions are the main tool used to compare and classify problems according to their difficulty.

In provable security papers the authors try to prove that a mathematical problem that is widely believed to be computationally hard, such as factoring large integers or finding elliptic curve discrete logs, *reduces* to a successful attack of a prescribed type on their cryptographic protocol. This means that anyone who could break their cryptosystem could also, with only a little extra effort, solve the supposedly hard math problem. Since that is assumed not to be possible, the conclusion is that the protocol is *provably* secure.

For mathematicians who study the provable security literature, as Menezes and I did, there are several reasons to be uneasy. Most obviously, a provable security theorem applies only to attacks of a specified sort and says nothing about clever

attacks that might not be included in the theorem. Moreover, the result is conditional in a strong sense. Unlike in mathematics, where conditional theorems usually mean something like "assuming that the Riemann Hypothesis is true" (which it almost certainly is), in cryptography the condition is of the sort "assuming that no one finds an improved algorithm for a certain math problem"—and that's anyone's guess. History has not been kind to the latter type of assumption. For example, in the late 1980s and early 1990s the development of the number field sieve for factoring an RSA modulus N resulted in a dramatic decrease of the running time of index-calculus factoring algorithms from $\exp\left((\log N)^{1/2+\epsilon}\right)$ to $\exp\left((\log N)^{1/3+\epsilon}\right)$.

Provable security results are often used to impress outsiders who have little understanding of their true meaning. Suppose that some people are using public key cryptography to protect credit card numbers in e-commerce, maintain confidentiality of medical records, or create digital signatures. How can they be certain that the system is secure? To nonspecialists "provable security" means that there's a guarantee that's every bit as ironclad as a proof of the Pythagorean Theorem. In our view this is very misleading.

There's also a difficulty that comes from the disciplinary culture of cryptography that I commented on before. People usually write papers under deadline pressure—more the way a journalist writes than the way a mathematician does. And they rarely read other authors' papers carefully. As a result even the best researchers sometimes publish papers with serious errors that go undetected for years.

In 1994 two of the leading specialists in the new area of provable security, Mihir Bellare and Philip Rogaway, proposed an RSA-based encryption method that they called OAEP (the O stands for "optimal," a much overused word in the overhyped high-tech world). They held the view that security proofs should be sufficiently detailed so that one can get concrete guarantees for specified key sizes and choices of parameters. Partly because of the security proof that accompanied OAEP, it was adopted for use in a new standard of Visa and MasterCard. It turned out, however, that the proof was fallacious, as Victor Shoup discovered seven years later. This was a bit of a scandal and caused many people to wonder about quality control in provable security papers.

If a careful and astute reader is watching closely—and Alfred Menezes is such a reader—then errors in proofs are discovered much more quickly. A case that in many ways is even more striking than that of OAEP is the recent flap over an "improved" set of key agreement protocols designed by Hugo Krawczyk. In February 2005 Krawczyk, who works for IBM and is a top researcher in provable security, submitted a paper to Crypto 2005 in which he claimed to have found flaws in the Menezes-Qu-Vanstone (MQV) key agreement system. He replaced it with a modified version (HMQV) that he claimed was both more efficient and *provably secure*. If his claims had been valid, this would have been a major embarrassment not only to Menezes and his coauthors, but also to NSA, which had licensed MQV from Certicom and whose experts had studied it carefully.

Krawczyk did not send his paper to Menezes or the other designers of MQV before submitting it, although to do so would be considered a standard courtesy in the scientific world. But what to me seemed more scandalous was that neither did anyone on the Crypto 2005 program committee. They apparently rushed to accept the paper after only a superficial reading. When Menezes finally got a copy of the paper—after it had been accepted by the program committee—he immediately saw that the so-called flaws in MQV that Krawczyk listed either were based on misunderstandings or else were picayune theoretical points that had no practical significance.

More importantly, Menezes found that the paper's main argument was fallacious. Krawczyk claimed that in his modified key agreement system he could increase efficiency by discarding a certain security check (called a "public key validation") that had been put into MQV so as to prevent known attacks. It was his security "proof" that gave him the confidence to do this. But Menezes quickly found that certain of the HMQV protocols succumb to the same attacks that MQV would have if those security checks had not been put in. After seeing that some of the conclusions of Krawczyk's theorems were false, Menezes started reading the "proof" carefully until he came upon a blatant gap in the argument.

Both Krawczyk and the referees on the program committee had been so mesmerized by the "proof" that they failed to use common sense. Anyone working in cryptography should think very carefully before dropping a validation step that had been put in to prevent security problems. Certainly someone with Krawczyk's experience and expertise would never have made such a blunder if he hadn't been over-confident because of his "proof" of security. As with many other over-hyped ideas—fallout shelters in the 1950s, missile shields in the 1980s—"proofs" of the security of a cryptographic protocol often give a false confidence that blinds people to the true dangers.

In our first paper on provable security, Menezes and I objected to the terminology:

> There are two unfortunate connotations of "proof" that come from mathematics and make the word inappropriate in discussions of the security of cryptographic systems. The first is the notion of 100% certainty. Most people

> not working in a given specialty regard a "theorem" that is "proved" as something that they should accept without question. The second connotation is of an intricate, highly technical sequence of steps. From a psychological and sociological point of view, a "proof of a theorem" is an intimidating notion: it is something that no one outside an elite of narrow specialists is likely to understand in detail or raise doubts about. That is, a "proof" is something that a non-specialist does not expect to really have to read and think about.
>
> The word "argument", which we prefer here, has very different connotations. An "argument" is something that should be broadly accessible. And even a reasonably convincing argument is not assumed to be 100% definitive. In contrast to a "proof of a theorem", an "argument supporting a claim" suggests something that any well-educated person can try to understand and perhaps question.

Menezes and I also investigated some subtler problems of interpretation of provable security results. Even when the proofs are correct, they often mask a big "tightness" gap. This means that in the reduction argument the attack on the protocol must be repeated millions of times in order to solve the hard computational problem. In this case the practical guarantee that one gets is very weak. Menezes found some extreme examples of this "nontightness" problem in a few well-known papers on random number generators. In one paper it turned out that, if you carefully follow the authors' argument with recommended parameter values, all they've really proven is that an attacker would need time at least 10^{-40} nanoseconds to break the system. That's much less time than it takes light to travel a micron.

What had happened was that people had made recommendations for parameter values that were based on an asymptotic theorem. That theorem said that in the limit as N approaches infinity, you can securely generate $O(\log \log N)$ pseudorandom bits each time you perform a squaring modulo the composite number N. (Here "securely" means, roughly speaking, that no one can distinguish between the sequence and a truly random one by an algorithm that runs in reasonable time.) However, as I mentioned when discussing Joe Silverman's xedni calculus, it is fallacious to use an asymptotic result as a practical guarantee of security. Rather, one needs to perform a detailed analysis using realistic ranges for the parameters. It is often a lot harder (as it was for xedni) to carry out this concrete analysis than to prove the asymptotic theorem, and sometimes the conclusions are not what one would hope for. In the case of the pseudorandom bit generator the analysis (if one assumes that $\log_2(\log_2 N)$ bits are taken in each iteration, as recommended) leads to an absurd lower bound on the amount of time that an adversary would need in order to successfully attack the generator.

The story of our first paper on "provable security" has an amusing postscript. Just before it was due to appear in *J. Cryptology*—and almost two years after it was accepted for publication—a member of the editorial board objected strongly to its acceptance by the journal. Although it was too late for him to block publication, the editor-in-chief was sufficiently worried that he wrote an unprecedented Editor's Note at the beginning of the January 2007 issue in which he justified his decision to go ahead with publication.

The editorial board member who objected to our article was Oded Goldreich of the Weizmann Institute, who is one of Israel's leading computer scientists and a top name (some would say *the* top name) in theoretical cryptography. When he was unable to prevent our article from appearing in *J. Cryptology*, he posted on the cryptography eprint server a 12-page essay titled "On Post-Modern Cryptography" that lashed out at us on philosophical grounds. (See `http://eprint.iacr.org/2006/461`.) He accused Menezes and me of being "post-modern" and "reactionary" because our criticisms of provable security "play to the hands of the opponents of progress."

The part of our paper that seems to have incensed Goldreich the most was our explanation of why we were not persuaded by certain arguments that he and others had made in order to undermine the so-called "random oracle" assumption. The random oracle assumption relates to what are called "hash functions" (short strings of symbols that act as a sort of "fingerprint" of a message). This assumption essentially says that the fingerprint that a well-constructed hash function gives is in practice indistinguishable from a random string of symbols. This is an intuitively reasonable assumption, and in our paper we argued that all attempts to undermine it—even ones that the authors claimed to be of practical relevance—in fact use constructions that violate basic cryptographic principles and so have no relation to real-world cryptography. We concluded our discussion by saying that "our confidence in the random oracle assumption is unshaken."

Goldreich responded to this by bringing down the wrath of the Old Testament upon us. Accusing us of turning the random oracle into a "fetish", he recounted a story from the Bible that our paper reminded him of (in what follows I've preserved

the emphasis, capitalization, and spelling of the original):

> Indeed, what happened with the Random Oracle Model reminds us of the biblical story of the Bronze Serpent, reproduced next. (See *Numbers* (21:4-8) and *2 Kings* (18:4).) During the journey of the People of Israel in the dessert, the prophet-leader Moses was instructed by the Lord to make a "fiery serpent" as a symbolic mean for curing people that have been bitten by snakes (which were previously sent by the Lord as a punishment for some prior sin). Several hundred years later, the bronze serpent made by Moses has become an object of idol worship. This led the righteous King Hezekiah (son of Ahaz) to issue an order for breaking this bronze serpent to pieces. Let us stress that the king's order was to *destroy an object that was constructed by direct instruction of the Lord*, because this object has become a fetish. Furthermore, this object no longer served the purpose for which it was constructed. This story illustrates the process by which a good thing may become a fetish, and what to do in such a case…. [G]iven the sour state of affairs, it seems good to us to abolish the Random Oracle Model.

Goldreich sees himself as a twenty-first-century righteous King Hezekiah defending the provable security researchers against infidels and post-modern fetishists such as Menezes and me. It is clear from his essay that he had not read our paper carefully before writing his response; nor does he seem to have been aware of our other two posted papers criticizing provable security. But of course it was not necessary to actually read the technical details in our three articles in order to denounce us on religious and philosophical grounds.

The angry reactions of a few researchers who seem to perceive our work as a threat to their interests are not the type of thing one normally encounters in theoretical mathematics, where usually the only issues that could cause someone to object to a paper would be an error or omitted acknowledgment of earlier work (neither of which has been found in any of our three papers on "provable security"). But far from being bothered by the accusations made by Goldreich and others, I am encouraged by them, because they at least show that people are paying attention.

Cryptography has the excitement of being more than just an academic field. Once I heard a speaker from NSA complain about university researchers who are cavalier about proposing untested cryptosystems. He pointed out that in the real world if your cryptography fails, you lose a million dollars or your secret agent gets killed. In academia, if you write about a cryptosystem and then a few months later find a way to break it, you've got two new papers to add to your résumé!

Drama and conflict are inherent in cryptography, which, in fact, can be defined as the science of transmitting and managing information in the presence of an adversary. The "spy vs. spy" mentality of constant competition and rivalry extends to the disciplinary culture of the field. This can get to be excessive—and even childish at times—but it also explains in part why it can be so much fun to do research in cryptography.