ANOTHER LOOK AT SECURITY DEFINITIONS

NEAL KOBLITZ AND ALFRED MENEZES

Abstract. We take a critical look at security models that are often used to give "provable security" guarantees. We pay particular attention to digital signatures, symmetric-key encryption, and leakage resilience. We find that there has been a surprising amount of uncertainty about what the "right" definitions might be. Even when definitions have an appealing logical elegance and nicely reflect certain notions of security, they fail to take into account many types of attacks and do not provide a comprehensive model of adversarial behavior.

1. Introduction

Until the 1970s, cryptography was generally viewed as a "dark art" practiced by secret government agencies and a few eccentric amateurs. Soon after the invention of public-key cryptography, this image changed rapidly. As major sectors of the economy started to become computerized — and large numbers of mathematicians and computer scientists entered the field of information security — cryptography became professionalized. By the 1990s, cryptographers could plausibly claim to occupy a subbranch of the natural sciences. In 1996 the *Handbook of Applied Cryptography* [69] proclaimed:

> The last twenty years have been a period of transition as the discipline moved from an art to a science. There are now several international scientific conferences devoted exclusively to cryptography and also an international scientific organization, the International Association for Cryptologic Research (IACR), aimed at furthering research in the area. (p. 6)

In addition to the professionalization of cryptography, there were two other reasons to view the field as becoming more "scientific" — the increasing use of sophisticated mathematical techniques and the development of rigorous theoretical foundations. The latter work could be seen in some sense as a continuation of the program initiated by the founders of modern computer science and information theory, Alan Turing and Claude Shannon, who set out to put those fields on a solid mathematical footing. As we explained in our first article on "provable security" [54],

> Starting in the 1980s it became clear that there is a lot more to security of a public-key cryptographic system than just having a one-way function, and researchers with a background in theoretical computer science set out to systematically develop precise definitions and appropriate "models" of security for various types of cryptographic protocols.

The exuberant mood during this early period has been vividly described by Phil Rogaway [84]:

Date: May 23, 2011; revised on March 27, 2012.

> It [the mid-1980s] was an extraordinarily exciting time to be at MIT studying theoretical computer science and cryptography. To begin with, it was a strangely young and vibrant place.... The theory group had three cryptographers, Shafi Goldwasser, Silvio Micali, and Ron Rivest.... Many of the papers that flowed from these people were absolutely visionary, paradigmatic creations. One has only to re-read some of their fantastical titles to re-live a bit of the other-worldliness of the time. How to play mental poker (1982). Paradoxical signatures (1984). Knowledge complexity and interactive proofs (1985). Proofs that yield nothing but their validity (1986). How to play *any* mental game (1987). The MIT cryptographers seemed to live in a world of unbridled imagination.
>
> As I saw it... cryptography at MIT did not much countenance pragmatic concerns... practical considerations would have to wait for some distant, less ecstatic, day. My wonderful advisor, Silvio Micali, would wax philosophically about how some definition or proof should go, and it is no exaggeration to claim that, as I saw it, philosophy and beauty had unquestioned primacy over utility in determining what was something good to do.

In developing a strong foundation for modern cryptography, a central task was to decide on the "right" definitions for security of various types of protocols. For example, the 1984 paper on "paradoxical signatures" by Goldwasser, Micali, and Rivest that Rogaway refers to contained a definition of security of a signature scheme that ever since has been central to any discussion of secure digital signatures. Informally speaking, they defined "existentially unforgeable against an adaptive chosen-message attack" to mean that even an adversary who is permitted to request valid signatures for any messages of his choice cannot feasibly produce a valid signature for any other message.

There are several reasons why definitions are important. In the first place, good definitions are needed in order to discuss security issues with clarity and precision. In the second place, such definitions allow analysts to know exactly what they're dealing with. When a protocol designer makes a security claim and a cryptanalyst then devises a break, the designer cannot simply say "that's not what I meant by security." If it weren't for precise definitions, security claims would be a moving target for analysts. In the third place, if one wants to be able to prove mathematical theorems, one needs precise definitions of what one is talking about. Thus, in their definitive textbook [48] on the foundations of cryptography, Katz and Lindell list "formulation of exact definitions" as Principle 1 in their list of "basic principles of modern cryptography." As they put it,

> One of the key intellectual contributions of modern cryptography has been the realization that formal definitions of security are *essential* prerequisites for the design, usage, or study of any cryptographic primitive or protocol.

Another benefit that is sometimes mentioned is that precise definitions enable cryptographers to avoid including features that are not needed for the security proofs. In [12] (p. 5 of the extended version) Bellare and Rogaway state:

> Our protocols are *simpler* than previous ones. Our ability to attain provable security while simultaneously simplifying the solutions illustrates an advantage of having a clear definition of what one is trying to achieve; lacking such a

> definition, previous solutions encumbered their protocols with unnecessary features.

In §5 we'll raise some concerns about over-reliance on formal security definitions as a guide for protocol construction.

There is one final "advantage" of precise definitions that is rarely mentioned in discussions of the subject. They are helpful for the attackers. Like everyone else, the adversaries have limited resources and a desire to concentrate them where they'll do the most good (or rather, the most bad). When publicity about a protocol specifies the security model that served as a guide for its design and a basis for the security guarantee, adversaries can also use that model as a guide, focusing their efforts on devising attacks that are outside the model. Experienced cryptanalysts are well aware of this use of precise descriptions of security models.[1]

* * *

This article is written for a general cryptographic readership, and hence does not assume any specific area of expertise on the reader's part. We have endeavored to make the exposition as self-contained and broadly accessible as possible.

The paper is organized as follows. We first discuss signature schemes and show that, despite being provably secure under the standard Goldwasser–Micali–Rivest (GMR) definition, several schemes are vulnerable to Duplicate Signature Key Selection (DSKS) attacks. We describe some real-world settings where DSKS attacks could undermine the usability of digital signatures. In §3 we examine some security issues in symmetric-key encryption. Focusing on Secure Shell (SSH) protocols, we find that provable security under the standard definitions does not give any assurance against certain classes of practical attacks.

We start §4 with a brief history of side-channel attacks and then comment on the long delay before theoreticians attempted to incorporate such attacks into their models of adversarial behavior. When they finally tried to do that – most notably by developing the concepts of leakage resilience and bounded retrieval – the practical relevance of their results has been questionable at best. In §5 we comment on the indispensable role of safety margins and dispute the claim that such features can be dropped in order to achieve greater efficiency whenever the security proof goes through without them. Finally, in §6 we make an analogy between the difficulty of modeling adversarial behavior in cryptography and the difficulty of applying mathematical models in economics. In §7 we list some conclusions.

2. Signatures

According to the Goldwasser–Micali–Rivest (GMR) definition [41], a signature scheme is (t, ϵ, q)-secure in the sense of unforgeability under adaptive chosen-message attack if no adversary that is allowed signature queries for $\leq q$ messages of its choice can produce a valid signature of an unqueried message in time $\leq t$ with success probability $\geq \epsilon$.

[1] Telephone conversation between Brian Snow (retired Technical Director of Research at NSA) and the first author, 7 May 2009.

In this section we discuss a type of attack on signature schemes that does not violate the GMR definition of a secure signature scheme but which, in our view, should nevertheless be a matter of concern for designers and users of digital signatures.

2.1. **The Duplicate Signature Key Selection (DSKS) attack.**

In a DSKS attack we suppose that Bob, whose public key is accompanied by a certificate, has sent Alice his signature s on a message m. A successful DSKS attacker Chris is able to produce a certified public key of his own under which the same signature s verifies as his signature on the same message m. We are not interested in the trivial attack where Chris simply claims Bob's public key as his own, and so we shall suppose that the certification authority (CA) demands a proof of knowledge of the corresponding private key before granting a certificate for the public key (this is required by several widely-used standards for public-key infrastructure, such as [73, 88]). Following [16, 68], we give some examples of DSKS attacks on well-known signature schemes. After that we discuss the potential damage that DSKS attacks can cause.

It should be noted that in all of these attacks the adversary needs to carry out only a modest amount of computation. In fact, in the first example in §2.2.1 he expends no more computational effort than a legitimate user.

2.2. **Examples of DSKS attacks on standard schemes.**

2.2.1. *GHR.* The Gennaro–Halevi–Rabin signature scheme [39] is a clever twist on RSA signatures that allowed the authors to prove existential unforgeability against chosen-message attack without using random oracles. It works as follows. Suppose that Bob wants to sign a message m. His public key consists of an RSA modulus N and a random integer t; here $N = pq$ is chosen so that p and q are "safe" primes (that is, $(p-1)/2$ and $(q-1)/2$ are prime). Let $h = H(m)$ be the hash value, where we assume that the hash function H takes odd values (so that there is negligible probability that h has a common factor with $p-1$ or $q-1$). Bob now computes $\widetilde{h}$ such that $\widetilde{h}h \equiv 1 \pmod{p-1}$ and $\pmod{q-1}$. His signature s is $t^{\widetilde{h}} \bmod N$. Alice verifies Bob's signature by computing h and then $s^h \bmod N$, which should equal t.

Suppose that an adversary Chris wants to mount a Duplicate Signature Key Selection attack. That is, he wants to find N' and t' such that $s^h \equiv t' \pmod{N'}$. But this is simple. He can take an arbitrary RSA modulus N' and then just set $t' = s^h \bmod N'$.

Remark 1. In [54] we argued that there is no evidence that the use of random oracles in security proofs indicates a weakness in the true security of the protocol. Thus, it does not make much sense to sacrifice either efficiency or true security for the sole purpose of getting security proofs without random oracles. Note that in [39] the main motive for introducing GHR signatures was to avoid the use of random oracles. The ease of carrying out a DSKS attack on GHR illustrates a danger in redesigning protocols so as not to need random oracles in a proof — doing so might open up new vulnerabilities to attacks that are outside the security model used in the proof.

2.2.2. *RSA.* Suppose that Bob has an RSA public key (N, e) and private key d. His signature s on a message m with hash value $h = H(m)$ is then $h^d \bmod N$. To verify the signature, Alice checks that $s^e \equiv h \pmod{N}$. The DSKS attacker Chris wants to construct a new public-private key pair (N', e'), d' such that $s^{e'} \bmod N'$ is h. First, Chris chooses two primes p' and q' such that (1) the product $N' = p'q'$ has the right bitlength to be an RSA modulus for the signature protocol; (2) both $p'-1$ and $q'-1$ are smooth, that is, they're products of small primes; (3) both s and h are generators of $\mathbb{F}_{p'}^*$ and $\mathbb{F}_{q'}^*$; and (4) $(p'-1)/2$ and $(q'-1)/2$ are relatively prime. Because of (2) and (3), Chris can easily solve the discrete log problems $s^x \equiv h \pmod{p'}$ and $s^y \equiv h \pmod{q'}$, using Pohlig-Hellman [80]. By the Chinese Remainder Theorem (which applies because of conditions (3) and (4)), he can find e' such that $e' \equiv x \pmod{p'-1}$ and $e' \equiv y$ (mod $q'-1$). Since Chris knows the factorization of $N' = p'q'$, he can easily compute the private key d' and get a valid public key certification (which presumably requires him to prove possession of the private key). It is easy to see that with the public key (N', e') the verifier accepts s as Chris's signature on the message m.

Remark 2. One could argue that Chris's RSA modulus N' was not properly formed, because it is well known that Pollard's $p-1$ algorithm [81, 72] can factor N' efficiently if either $p'-1$ or $q'-1$ is smooth. However, in the first place, Chris can easily choose $p'-1$ and $q'-1$ to be smooth enough for his purposes, but not smooth enough for N' to be factored feasibly. This is because, if $p'-1$ and $q'-1$ each have their two largest prime factors randomly chosen and of the same bitlength as r, then Chris's discrete logarithm problems can be solved in time roughly $\sqrt{r}$, whereas factorization of N' takes time roughly r. If, for example, r has in the range of 60 to 70 bits, then the Pollard $p-1$ algorithm involves raising to a power modulo N' that is the product of all primes $< 2^{60}$ or $< 2^{70}$, and this is not practical.

In the second place, some of the widely-used industry standards for RSA encryption and signature do not make any stipulations on the selection of the prime factors of the modulus. For example, RSA PKCS #1 v2.1 [87] even permits N to be the product of more than two primes, and it also allows each user to select a different e of arbitrary bitlength. Thus, there is nothing in those standards to preclude Chris's parameter selection.

Remark 3. The conventional wisdom about parameter selection is that for greatest security it is best to choose parameters as randomly as possible. If special choices are made in order to increase efficiency, the presumption is that the reduced randomness might at some point make an adversary's task easier. In [52] we described scenarios in which these assumptions might be wrong. One finds further counterexamples to the conventional wisdom if one is worried about Duplicate Signature Key Selection attacks, because the greater the number of random choices in the key generation algorithm, the easier the adversary's task.

For example, we just saw how to mount a DSKS attack on RSA signatures if each user chooses her own (supposedly random) encryption exponent e. However, if e is fixed for all users — say, $e = 3$ — then for most hash values h — that is, for most messages m — the adversary Chris cannot find an RSA modulus N' for the attack. This is because N' would have to be an n-bit divisor (where n is the bitlength of RSA moduli in the given implementation) of the $2n$-bit integer $(s^3 - (s^3 \bmod N))/N$, and Chris must be able to compute the factorization of N'. For most random s this cannot feasibly be done. Similarly, in §2.2.4 we shall see how a DSKS attack on ECDSA can be carried out if each user chooses her own generating point on the curve; however, if the generating point is fixed for all users, this attack will not work. Thus, for purposes of avoiding DSKS attacks the conventional wisdom about randomness in parameter selection is wrong.

2.2.3. *Rabin signatures.* Roughly speaking, Rabin signatures [82] are a variant of RSA with public exponent 2. Bob's public key is an RSA modulus N and his secret key is the factorization of N. If $h = H(m)$ is the hash value of a message, then Bob's signature s is the squareroot of h modulo N. (Some modifications are necessary because only 1/4 of the residues have squareroots, and such a residue has 4 squareroots; but we shall not dwell on the details.) The verifier checks that $s^2 \equiv h \pmod{N}$. All the DSKS attacker Chris has to do is choose N' to be the odd part of $(s^2 - h)/N$, which is likely to have the right bitlength to be an RSA-Rabin modulus. Then clearly s verifies as the signature of m under the public key N'.

Chris will succeed in this DSKS attack for a substantial fraction of all possible messages m. In order to get a certificate for his public key N', he will have to demonstrate to the CA that he knows its factorization. He can factor N' using the Elliptic Curve Method if N' is the product of a prime and a smooth number. From a remark in §4.2 of [16] it follows that there is a roughly 50% probability that a random 1024-bit integer N' is the product of a prime and a number that is sufficiently smooth so that N' is ECM-factorizable with current techniques.

2.2.4. *ECDSA.* The system-wide parameters consist of an elliptic curve E defined over a prime field $\mathbb{F}_p$ such that $\#E(\mathbb{F}_p)$ has a large prime factor n. Bob's private key is an integer d, $0 < d < n$, and his public key is a point $P \in E(\mathbb{F}_p)$ of order n and the point $Q = dP$. To sign a message m with hash value h, Bob chooses a random integer k, $0 < k < n$, and computes $r = x(kP) \bmod n$, where $x(kP)$ denotes the x-coordinate of the point kP. He also computes $s = k^{-1}(h + rd) \bmod n$; his signature is the pair (r, s). Signature verification consists of computing $R = s^{-1}hP + s^{-1}rQ$ and checking that $r = x(R) \bmod n$.

Given Bob's public key and his signature (r, s) on a message m with hash value h, the DSKS attacker Chris chooses an arbitrary integer d', $1 < d' < n$, such that the integer t defined as $t = s^{-1}h + s^{-1}rd' \bmod n$ is nonzero. Chris then computes $R = s^{-1}hP + s^{-1}rQ$, and then $P' = (t^{-1} \bmod n)R$ and $Q' = d'P'$. His public key is (P', Q'). Then (r, s) verifies under this public key, because $s^{-1}hP' + s^{-1}rQ' = (s^{-1}h + s^{-1}rd')P' = tP' = R$.

As mentioned before, this attack works only because each user gets to choose her or his own generating point P of the order-n subgroup. If this point is fixed as part of the domain parameters, then this DSKS attack does not work. In §2.6 we shall see that a somewhat weaker type of attack can still be launched in that case.

Remark 4. In a generalization of the DSKS attack that is of interest in Unknown Key Share attacks on key agreement protocols (see [16]), Chris has a message m' that may be different from m and wants to choose a key pair so that Bob's signature s on m also verifies as Chris's signature on m'. In the case of GHR, RSA, and ECDSA, the DSKS attacks described above immediately carry over to this generalized version — all one has to do is replace $h = H(m)$ by $h' = H(m')$ in the description of what Chris does. However, the DSKS attack on Rabin signatures does not carry over, because Bob's RSA modulus N does not divide $s^2 - h'$, and for almost all random h' Chris wouldn't be able to find a suitable factor N' of $s^2 - h'$.

Remark 5. A distinction should be made between DSKS attacks that require only knowledge of the message and signature and attacks that also require knowledge of the signer's public key. Among the DSKS attacks described above, the ones on GHR and RSA signatures are in the former category, while the attacks on Rabin and ECDSA signatures are in the latter category. In situations where the signer is known to come from a fairly small set of parties whose public keys are readily accessible, it is possible to determine the identity of the signer from the signature by simply attempting to verify the signature using all possible public keys. However, in other situations it might not be practical to do this, and so one should distinguish between signature-only DSKS, which can be used to attack anonymous contests, auctions, etc. (see §§2.3.2 and 2.3.3), and DSKS attacks that require knowledge of the signer's identity.

2.3. **Real-world impact of DSKS attacks.**

2.3.1. *Double-blind refereeing and plagiarism.* A DSKS attack can be most effective when the message that was signed did not contain identifying information about the sender. For example, at Crypto and some other conferences that use double-blind refereeing, it is forbidden for authors to include identifying information in the submitted version of a paper. Suppose that Bob emails his submission to the program chair Alice. He also sends her his signature so that his archrival Chris, who he believes lacks any sense of professional ethics, can't steal his work. The paper is rejected from Crypto, but a few months later appears somewhere else under Chris's name. Bob appeals to Alice, sending her his certificate for the public key that she uses to verify his signature, and she then confirms that Bob was the one who submitted the paper to Crypto. However, Chris points out that the signature actually verifies under Chris's certified public key. In anticipation of Bob's plagiarism complaint, Chris created his own keys so that the signature would verify as his.

Perhaps this scenario is not very realistic. Chris might have difficulty explaining why Alice's email records show Bob's return address on the submission. (This is assuming that Bob does not use an anonymizer such as Tor for his submission.) In addition, public key certificates normally include a date of issue. If Chris received his certificate after Bob, that means that Chris must have carried out the DSKS attack on Bob, rather than vice-versa. But what if Bob, who was frantically rushing to meet the submission deadline, signed the submission before getting his public key certified? In that case Chris might have been able to get his public key certified before Bob does. Moreover, note that neither the GMR security model nor the commonly-used digital signature standards require that messages or certificates include dates or timestamps, or that senders use valid return addresses.

Remark 6. Even in situations where there is no need for anonymity, it is becoming more common not to bother with identifying information in a message. The old-fashioned letter format — starting with a salutation and ending with a signature *over the printed name* — is not a universally accepted practice in the digital age. Often senders simply assume that the identity of the sender and recipient will be clear enough from context and can be determined in case of doubt from the message header. Similarly, with larger categories of communication being automated, efficiency seems to call for eliminating any unnecessary ingredients in a message. If Alice's computer is expecting a message from Bob, and when the message arrives her computer verifies his signature, then it seems pointless also to include Bob's name and address in the message.

2.3.2. *Anonymous contests.* The idea of anonymous contest entries — presumably to prevent jury bias — goes back a long way. For example, the Prix Bordin of the French Academy of Sciences, which Sofia Kovalevskaia won in 1888 for her work on rotation of a solid object about a fixed point, worked as follows. The contestant submitted her manuscript anonymously and wrote an epigram at the top that served as an identification number. She wrote the same epigram on a sealed envelope that contained her name and address. The judges would open the envelope of the winning entry only after the selection was complete.[2] An interesting feature of the Prix Bordin was that the judges promised to open the envelope only of the winner; the unsuccessful contestants were guaranteed anonymity [24]. Perhaps prominent mathematicians felt that it would have been embarrassing for the public to know that they had entered the competition and not been selected.

Returning to the present, let's suppose that an anonymous contest with a monetary award uses a digital signature scheme that is vulnerable to a signature-only DSKS attack. Alice, as one of the judges, receives the anonymous submissions with signatures. (In [102] it is proposed that to better preserve anonymity only part of the signature should be sent in; the winner sends the rest of the signature later when proving ownership.) When the finalists are chosen, Alice shares this data with her boyfriend, who has social ties to the cybercrime underworld of which she is unaware, and he sends it to his friend Chris, who immediately gets a certified key pair for each finalist's signature. (If only part of the signature is given, he can fill in the rest randomly and get a key pair for the resulting signature.) As soon as the winning entry is announced, Chris claims the prize and disappears with it. When the legitimate winner appears, Chris is nowhere to be found.

2.3.3. *Coupons.* Two other scenarios for Duplicate Signature Key Selection attacks are suggested in [95]. (1) Suppose that an electronic lottery system works as follows. Bob buys a lottery ticket consisting of a serial number that he signs. The lottery company keeps a record of the number and his signature. When the winning lottery numbers are announced, if Bob's is one of them he can prove ownership because the signature verifies under his certified public key.

(2) Let us say that a shop issues electronic coupons worth \$10 on future purchases to shoppers who have spent \$500 there. Once again, the coupon consists of a number that the shopper Bob signs. The shop saves a copy of the signed coupon. When Bob wants to use it, the shop verifies the signature and redeems the coupon.

As in the case of anonymous contests, a DSKS attacker can claim the lottery winnings or redeem the coupon before the legitimate buyer does. In addition, if an electronic coupon or lottery system does not have a mechanism for removal or expiration as soon as payment is made, then it can be undermined by a weaker type of attack than DSKS, where one supposes that the original signer is dishonest and arranges for both himself and a friend to claim the award or redeem the coupon; we discuss this briefly in §2.6.

[2] At least in Kovalevskaia's case, and most likely in other cases as well, the judges knew perfectly well who the author was, and the anonymity was merely a polite fiction [51]. But the actual historical nature of the Prix Bordin does not concern us here.

Remark 7. A recent paper of Bellare and Duan [7] defines a type of anonymous signature that can be used in any of the scenarios discussed above. In one of their constructions, in order to sign a message m Bob first uses a conventional signature scheme to produce a signature s. He then selects a random string r and computes $w = H(s, r, pk)$, where pk denotes his public key for the signature s and H is a hash function. His anonymous signature on m is w. When Bob is ready to claim his signature, he reveals (s, r) and produces his public-key certificate.

Note that this scheme prevents DSKS attacks for the same reason that including the public key in the message does [68]: Chris has no way to replace Bob's public key with his own and have the signature still verify.

2.3.4. *Digital vs. handwritten signatures.* The usual way that digital signatures are explained to the general public is that they have all the advantages of traditional handwritten signatures, but are much better. In the first place, they are much harder (in popular accounts the word "impossible" is sometimes used) to forge. In the second place, digital signatures authenticate not only the source, but also message integrity. A hand-signed document is much easier to tamper with undetected; in practice, an important document is required to be signed on each page, and it is assumed that a hard copy page with signature would be hard to alter without leaving a trace.[3]

Popular explanations of digital signatures do not usually delve into the familiar features of handwritten signatures that do not carry over. In the first place, an old-fashioned signature is an intrinsic feature of the signer, whereas an electronic one (except for biometrics) is to all appearances nothing but a random sequence of bits. In the second place, handwritten signatures provide a direct link from the signer to the document. For example, a user's bank branch kept a signature card, which in the case of a large check would be compared with the handwritten name on the document. In contrast, digital signatures are more indirect: they link the document to a key-pair, which, in turn, is linked to the signer's identity by means of a certificate. In the third place, old-fashioned signatures were (at least in theory) unique. No two people would have identical signatures. In the fourth place, a given signer would have only one or two signatures (perhaps a quick one for informal uses and a long one for important documents). In the digital world a user can create an arbitrary number of authenticated key-pairs for signatures, provided that he is willing to pay the CA's fee for certifying them.

It is because of these differences between digital and handwritten signatures that we have to worry about another damaging potential use of DSKS attacks: to undermine non-repudiation.

2.4. **The weakening of non-repudiation.** Since ancient times the function of signature systems has been twofold: source-authentication and non-repudiation. You needed to be sure who sent the message or document, and you wanted the sender not to be able to later deny having signed it. In many settings non-repudiation rather than authentication has been the most important need.

[3] The first author came to appreciate the validity of this assumption many years ago when he had to pay a $25 fine for reusing his university's daily parking permit; the change of date with the help of whiteout turned out not to be as undetectable as he had hoped.

For example, in America, where illiterates have traditionally been permitted to sign with an X, such signatures are required to be witnessed. The reason is to prevent Bubba[4] from later claiming

> Ain't my fuckin' X. Must be somebody else's fuckin' X.
> Roll Tide!
> We the people![5]

In the digital age, the authentication requirement can be met by symmetric-key cryptography — namely, by message authentication codes (MAC schemes) or the more general MA schemes. It is the second property — non-repudiation — that required the invention of public-key cryptography. Indeed, the ability to achieve non-repudiation in electronic communication was a central achievement — arguably even more important than public-key encryption — of the historic RSA paper [83].

The first U.S. government standard for digital signatures [74] makes it clear that non-repudiation is a central feature of a signature scheme:

> This Standard specifies a suite of algorithms that can be used to generate a digital signature. Digital signatures are used to detect unauthorized modifications to data and to authenticate the identity of the signatory. In addition, the recipient of signed data can use a digital signature as evidence in demonstrating to a third party that the signature was, in fact, generated by the claimed signatory. This is known as non-repudiation, since the signatory cannot easily repudiate the signature at a later time.

A serious practical danger of Duplicate Signature Key Selection attacks is that they'll undermine non-repudiation. Suppose that Bob signed a contract promising something to Alice. Later he refuses to honor the contract, and she takes him to court. Bob denies having signed the document, and hires a lawyer. What happened was that Bob never intended to carry out his promise, and before sending the contract to Alice he created a half-dozen entities with certified public keys such that his signature on the contract verifies under any of those keys.[6] His lawyer has an expert witness testify that there are many people other than Bob who could have sent the document and signature to Alice. In essence, Bob is able to say what Bubba said before him — "Ain't my fuckin' X!"

One can argue, of course, that the lawyer's strategy might not work. Most likely the contract included identifying information about Bob, who might not have a plausible explanation of why someone else would have signed a contract with Bob's name on it. If the lawyer says that perhaps someone was trying to harm Bob by falsely transmitting a promise that appeared to come from him, then Alice's lawyer might get the expert witness under cross-examination to admit that a duplicate signature is possible only if

[4] We chose the name Bubba rather than Bob because in all protocol descriptions of which we're aware Bob is assumed to be literate.

[5] The first slogan refers to the football team of the University of Alabama, which is nicknamed the Crimson Tide. Bubba is a proud alumnus of that institution, and played football for them when he was a "student." The second slogan is the motto of the Tea Party movement, which Bubba strongly supports. He doesn't believe in evolution or global warming, and is very proud to be an American because only in America do people like Bubba wield great political influence.

[6] The Rabin signature scheme allows Bob at most one DSKS attack on a given signature (see §2.2.3); however, the GHR, RSA, and ECDSA signature schemes (§§2.2.1, 2.2.2, 2.2.4) allow Bob to create an unlimited number of entities whose public key will verify a given signature.

Bob's valid signature was first available to the attacker. And no one but Bob could have created that valid signature (since the signature scheme is secure in the GMR sense). So if Alice has a good lawyer, Bob might lose in court. However, the signature scheme's failure to resist DSKS attacks at the very least muddies the water. The cryptographic weakness turns a simple court case into a more difficult one.

Moreover, Alice might lose in court for the simple reason that judges and jurors would be inclined to regard a signature scheme as faulty if it allows for different parties to have identical signatures on a document. That is, Bob's lawyer might not have to reconstruct a plausible scenario in which Bob did not really sign the document; it might be enough for him to impugn the credibility of the entire signature scheme. It's not clear how the courts would rule on the issue of whether a signature system that seems to have a serious flaw can still be used to establish the validity of a contract.

Even if a convincing mathematical argument can be made to the effect that Bob truly must have signed the contract, and that the other parties' identical signatures are either later fabrications or else irrelevant, that line of argument might not completely restore the judge's or jury's confidence in the signature scheme. After all, in many parts of the world, including the United States, the legal status of digital signatures remains unresolved, and there is still some resistance and mistrust. In such a social context, trying to get a court to accept the use of an apparently flawed digital signature scheme might be an uphill battle.

Remark 8. Recently A. K. Lenstra *et al.* [61] discovered that about 0.2% of users of RSA share a public or private key with another user (that is, either their modulus N is the same or else one of the prime factors is the same). As a result there has been great public concern about the problem of duplicate keys — the paper [61] was, for example, the subject of an article [65] in the business section of *The New York Times*. In this climate, the task of Alice's lawyer in convincing a jury of the legitimacy of Bob's signature would be particularly difficult. The jury would think that the situation created by the DSKS attack was very similar to different users sharing the same RSA modulus (or being able to determine another user's RSA secret because of a shared prime factor). It would be very hard to convince them that, unlike in the duplicate key scenario, Bob's signature was valid and the DSKS obfuscation by Bob's lawyer should be disregarded. Alice's lawyer would have to try to educate the jury about the difference between duplicate-signature key selection and duplicate keys. Good luck explaining that to a jury!

2.5. **The history of DSKS.** Duplicate Signature Key Selection attacks were first developed in 1998 not for the purpose of critiquing signature schemes, but rather in order to demonstrate the possibility of Unknown Key Share (UKS) attacks on station-to-station (STS) key agreement [30]. The occasion was a July 1998 ANSI standards meeting that discussed and compared the Menezes–Qu–Vanstone (MQV) and STS key agreement protocols. In response to Kaliski's UKS attack on MQV [46], Menezes demonstrated that a UKS attack was also possible on STS.

Nor was the published version of this work in 1999 [16] concerned with signatures except as an ingredient in key exchange. The authors explicitly discounted the importance of their work for signatures:

> It must be emphasized that possession of the duplicate-signature key selection property does not constitute a weakness of the signature scheme — the goal of a signature scheme is to be existentially unforgeable against an adaptive chosen-message attack [41].

Thus, their confidence in the Goldwasser–Micali–Rivest [41] definition of a secure signature scheme caused them to underestimate the significance of their attack for signature schemes.

There are two questions one can ask after one devises a type of attack on a signature scheme: (1) "Are there practical scenarios in which susceptibility to our attack would be a very bad property for a signature scheme to have?" (2) "Is vulnerability to our attack a violation of the GMR definition of a secure signature scheme?" When the answer to the second question is "no," one is less likely to ask the first question. Thus, over-reliance on the GMR definition can cause one to overlook a security issue that might be important in certain settings.

Five years later in [68] the question of DSKS attacks was finally treated as an issue for signature schemes.

> By considering the so-called "duplicate-signature key selection" attacks [16] on some commonly-used signature schemes, we argue that the well-accepted security definition for signature schemes (existential unforgeability against adaptive chosen-message attacks) by Goldwasser, Micali and Rivest [41] is not adequate for the multi-user setting...
>
> The question then is whether a [DS]KS attack really constitutes an attack on the signature scheme, or whether it is the result of an improper use of the signature scheme. Certainly, a successful [DS]KS attack does not violate the GMR security definition since there is only one public key in the single-user setting... On the other hand, it was shown in [16] that the station-to-station (STS) key agreement protocol [30] when used with a MAC algorithm for key confirmation succumbs to an unknown key-share attack when the signature scheme employed is susceptible to [DS]KS attacks. Since unknown key-share attacks can be damaging in practice, a [DS]KS attack in this application can have damaging consequences. Moreover, in the physical world of hand-written signatures, one would certainly judge an adversary to be successful if she can pass off a message signed by A as her own without having to modify the signature.
>
> Our thesis then is that a [DS]KS attack should be considered an attack on a signature scheme in the multi-user setting.

Remark 9. Here the authors take pains to distinguish between single-user and multi-user settings and to question the adequacy of the GMR definition only in the multi-user setting. However, the dividing line between single-user and multi-user settings is not always clear. In the scenario described in §2.4, Bob is the only one who communicated with Alice. He created several dummy users for the purpose of undermining the credibility of the signature scheme, but this should not really be considered a multi-user setting. More generally, a typical DSKS attack involves a single honest signer and an attacker who is given the power to register as a second legitimate user of the signature scheme. (In the GMR security definition the adversary is not given this power, and is not judged to be successful unless he produces a new signature that verifies under the honest signer's public key.) Again it is questionable whether such a scenario should be classified as a multi-user setting.

The main result in [68] was to show that DSKS attacks on the standard signature schemes can be prevented if the signer's public key is always included in the message. However, signature standards do not normally include such a requirement.

The reaction to DSKS attacks by some of the leading researchers in provable security has been curious. These researchers have written extensively about the formulation of comprehensive definitions of security of signature schemes in settings where they're being used in conjunction with public-key certificates, key agreement schemes, and other protocols. However, they have not dealt with DSKS attacks. On rare occasions (see [19, 20, 31]) they have briefly acknowledged the possibility of DSKS attacks. In [31] in reference to [68] Dodis *et al.* commented:

> However, their duplicate-signature key selection attack is not a flaw from the view of standard security notions and can be thwarted with ease.

But even if the DSKS attacks "can be thwarted with ease" — which might be true in some settings, but is probably false in others — it should still be a matter of concern that the need for such modifications in protocols (such as requiring the sender to include his public key in the message and requiring the recipient to check that it's there) was not anticipated when standards were drawn up, in part because the GMR "standard security notion" didn't require them. After all, why burden a protocol with unnecessary validation steps if it's "provably secure" under the GMR definition without them?

The viewpoint on DSKS attacks of other prominent researchers in provable security seems to be similar to that of Dodis *et al.* In his treatment of the theory of signature schemes [20], Canetti dismissed DSKS attacks in a footnote, in which he cited [68], explained what a DSKS attack is, and then said that while resistance to such attacks

> may be convenient in some specific uses, it is arguably not a basic requirement [for] signature schemes in general protocol settings.

It is peculiar that, despite their potential use to undermine non-repudiation and to break anonymous contest and coupon systems, Duplicate Signature Key Selection attacks are not perceived as important by leading researchers, such as Dodis and Canetti.

Remark 10. A possible defense of the GMR definition is that it was never intended to cover some of the types of applications described above. In other words, a GMR-secure signature scheme should not necessarily be used in applications such as double-blind refereeing, anonymous contests, and coupons. However, in general no such warning or qualification is given when leading researchers in provable security write about the GMR definition. For example, Cramer and Shoup [26] write:

> This is the strongest type of security for a digital signature scheme that one can expect, and a signature scheme that is secure in this sense can be safely deployed in the widest possible range of applications.

Moreover, it seems that signature schemes are rarely if ever deployed in the exact setting of the GMR definition — in particular, signature schemes are generally used in the multi-user setting — and so a user or protocol designer should *always* give additional scrutiny to the security of a protocol that employs a signature scheme, no matter how simple or natural the protocol may appear to be. Since the GMR definition is widely considered to be the "right" one, we think this is an unreasonable demand to make of the users of a "provably GMR-secure signature scheme". The need for additional scrutiny of protocols illustrates a limitation of the GMR definition.

Remark 11. As mentioned above, a proof of security of a signature scheme under the GMR definition gives no assurance of how it will behave in the multi-user setting. Another class of protocols where the multi-user setting has been neglected in the literature is MAC schemes, where the conventional security definition is concerned with only one legitimate pair of users. However, in practice MAC schemes are typically deployed in the multi-user setting where new attacks may be possible. An example of such an attack can be found in [22]. It is worth pointing out that this attack is applicable when the MAC scheme is used precisely for its intended purpose — for authenticating the origin and content of messages — and not in some complicated protocol that employs a MAC scheme along with other components.

2.6. **Another twist.**

In [18] the authors proposed a type of attack in which a dishonest signer colludes with a friend, giving him keys under which the signature will verify for either of them. The victim, then, is some third party. Although there are perhaps fewer practical scenarios than in the case of DSKS where this type of attack does real damage, one would want a signature scheme to resist this kind of compromise in the settings from [95] that we described in §2.3.3.

Suppose that Bob has signed the message m with hash value h using ECDSA, as described in §2.2.4, except that now the generating point P is fixed for all users, and a public key consists only of the point $Q = dP$, where d is the private key. Bob is dishonest, and is in cahoots with his friend Chris, whom he wants to have a public-private key pair under which Bob's signature (r, s) will also verify for Chris. He uses the fact that a verifier looks only at the x-coordinate of the point R, where $R = s^{-1}hP + s^{-1}rQ$, and on an elliptic curve in Weierstrass form the point $-R$ has the same x-coordinate as R. Thus, Chris takes $d' = (-2r^{-1}h - d) \bmod n$, $Q' = d'P$. Since $s^{-1}hP + s^{-1}rQ' = (s^{-1}h + s^{-1}r(-2r^{-1}h - d))P = -(s^{-1}h + s^{-1}rd)P = -R$, the signature also verifies as Chris's. Notice that, unlike DSKS (see §2.2.4), this weaker kind of attack works even if the generating point P is fixed for all users.

3. Symmetric-Key Encryption

Finding the "right" definition of security is often elusive. Concepts such as symmetric- and public-key encryption, hash functions, and message authentication have been around for decades, but the choice of security models is still fraught with unsettled issues.

3.1. **Public-key encryption.**

A central concept in encryption is resistance to chosen-ciphertext attack (CCA). In general terms, this means that the adversary Chris is permitted to ask Alice for the decryption of any ciphertexts of his choice except for the target ciphertext that he wants to cryptanalyze. In the indistinguishability model (IND-CCA) Chris gives Alice two plaintext messages of the same length, and she randomly chooses one to encrypt. That is, the target ciphertext is an encryption of one of the two messages. Chris must decide which it is with a substantially better than 50% probability of success.

However, there are four variants of this definition, depending on whether or not Chris is forbidden from querying the target ciphertext before as well as after he knows that it is the target ciphertext (in [9] the variant is denoted B if the ban applies to both stages and S if it applies only to the later stage after Chris is given the target ciphertext), and whether Alice's response to a request to decipher the target ciphertext is to refuse to do it or just to disregard the given run of the attack in the count of Chris' success rate (in [9] the former is denoted E and the latter is denoted P). In [9] the authors showed, somewhat surprisingly, that variant SP is a strictly stronger notion of security than BP, and that variant BP is strictly stronger than BE. This result does not seem to have any practical significance, because the examples used to separate SP, BP, and BE are contrived. However, in view of the importance that leading researchers attach to definitions as "*essential* prerequisites for the design, usage, or study of any cryptographic primitive or protocol" ([48], emphasis in original), it is curious that there is still no agreement on which variant should be used to define secure encryption.

3.2. **Hash functions.** It seems to be very difficult to come up with a definition of a secure hash function that, first of all, is adequate for all likely uses to which it will be put and, second of all, is not so strong as to make it impossible to prove security results. During the NIST competition to design a new Secure Hash Algorithm (SHA-3), there was considerable debate and no consensus about what security model should be used in "proofs" of security that would accompany proposed protocols. When NIST finally announced the five finalists and explained the reasons for the selection [96], surprisingly little mention was made of security proofs; the decisions were explained almost entirely in terms of concrete cryptanalysis, qualitative assessments of security margins, and efficiency considerations.

3.3. **Message authentication.** Message authentication protocols can be of two types. In a message authentication code (MAC) any message M is given a tag t_M using a key-dependent deterministic function. In a more general message authentication (MA) scheme, the tag t_M is computed using a randomized function. Generally speaking, security of either a MAC or MA scheme is defined in the same way as for a signature scheme (see §1). Namely, the adversary Chris, who is allowed to query the tags of any messages of his choice — that is, he is given a message-tagging oracle — then must produce a valid tag of any message other than the ones he queried. However, there are two variants of this definition for a general MA scheme, depending on whether or not Chris is also given a tag-checking oracle that, given M and t, tells him whether or not t is a tag of M. In [8] the authors showed that these two variants are not equivalent. Even though the notion of an MA scheme is fundamental, it is not yet clear whether or not the "right" definition of security needs to allow the adversary the use of a tag-checking oracle.

3.4. **Attacks on symmetric-key encryption schemes.** Similar difficulties have arisen in the search for definitions that could be used in proofs that symmetric-key encryption schemes will not succumb to specific types of attacks:

(1) In [98], Vaudenay proposed a new theory for block cipher construction that would supposedly ensure resistance to differential cryptanalysis. However, the following year Wagner [99] devised a new attack on block ciphers that exploited

differentials not considered in Vaudenay's security model; this attack was demonstrated to be effective on a block cipher constructed using Vaudenay's methodology.

(2) There has been a lot of controversy (see, for example, [1]) over whether or not related-key attacks (RKA) on the Advanced Encryption Standard (AES) and KASUMI (as reported in [15, 34]) are a practical threat. The main justification for considering RKA to be a legitimate attack is the possibility of side-channel attacks (see §4.1) that induce faults in the encrypting device. Attempts to construct theoretical models of RKA-resistance have fallen far short of what would be needed to provide realistic assurances. A recent version of RKA-resistance for pseudorandom permutations developed by Bellare and Cash [6], which they describe as an improvement over [10], assumes that the keys are elements of some group and the key perturbations arise from the group operation; as the authors of [6] acknowledge, this is almost never the case in practice.

(3) In [22] plausible attacks are described on several "provably secure" authenticated encryption schemes, including Offset Codebook Mode (OCB) [85] and Synthetic Initialization Vector (SIV) [86]. The attacks do not contradict the security proofs for OCB and SIV since the security definitions are in the 'single-user setting', which considers only one legitimate pair of communicating parties, whereas there are many pairs of communicating parties in real-world deployments of encryption schemes.

3.5. **Provable security of SSH.** The Secure Shell (SSH) protocol is a widely-used mechanism for securing remote login and other secure network services over an unsecured network [103]. Attempts to rigorously establish its security have an interesting history. In this section we shall describe an episode that illustrates how difficult it is to find an adequate formal model and suggests that there is some justification for Kevin McCurley's remark in his invited talk [66] at Eurocrypt 2006 that

> *...the structure of the Internet as we know it may actually preclude the existence of any reasonable model for completely secure encryption.* (emphasis in original)

We shall neglect some details and variants of SSH and consider only a few versions in a somewhat simplified form. We shall consider SSH in cipher-block-chaining (CBC) mode using AES as the underlying block cipher. Let us first review how data is encrypted and authenticated in SSH using CBC mode with AES. We suppose that Bob and Alice have already agreed upon a shared secret key for AES (which was presumably exchanged using a public-key protocol), and Bob wants to send Alice a long message. The message is divided into pieces that are each sent in a single "packet." The data "payload" making up each piece has a certain bytelength ℓ_d, and some padding (usually random) of bytelength ℓ_p is appended. A packet-length field and a padding-length field are placed in front of the payload, so that the packet has the following appearance:

Packet−length	Padding−length	Payload	Padding

Here the padding-length satisfies $4 \le \ell_p \le 255$, the padding-length field (containing the number ℓ_p) has only 1 byte allotted to it, and the packet-length field (containing the number $1 + \ell_d + \ell_p$) has 4 bytes allotted to it (and there might be further restrictions on this number).

In SSH using CBC mode with AES, the packet is split into blocks, each of length 16 bytes. An initialization vector (IV) is chosen; this IV is prepended to the ciphertext as the 0-th ciphertext block C_0. As depicted in Figure 1, the first block M_1 of the packet is XORed with the IV, and the result is put through AES to get the first ciphertext block C_1. Next, the second plaintext block M_2 is XORed with C_1 (this is called "chaining"), and the result is put through AES to get C_2. This process continues until the entire packet has been converted into ciphertext.

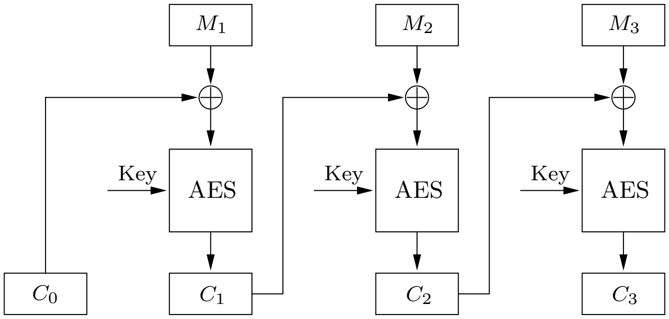

FIGURE 1. CBC encryption.

In addition, the packet (with a packet number prepended) is put through a Message Authentication Code (MAC) to get a tag. Bob sends Alice the encrypted packet and its tag. Alice then uses the secret AES key and the initialization vector C_0 to decrypt the message — starting with C_1, at which point she knows the packet-length and the padding-length — and verifies the MAC tag.

The most commonly used variant of SSH in CBC mode with AES has a feature called inter-packet chaining (IPC). This means that only the IV of the first packet is chosen randomly; each subsequent packet sets its IV equal to the last block of ciphertext from the previous packet. In addition, Bob typically chooses new pseudorandom padding for each packet (although the standards allow a fixed padding that's the same for all packets).

In [11] Bellare, Kohno, and Namprempre found a chosen-plaintext attack on SSH-IPC. This type of attack is of practical value in situations where the plaintext is known to come from a small set of possibilities ("buy" or "sell"; "attack now" or "delay attack"). The attack in [11] proceeds as follows. Chris has a target ciphertext block C^*, which is assumed to be the first block of an encrypted packet that Bob sent to Alice, and he has a guess P about what the corresponding plaintext block is. He knows the initialization vector (IV) for that block (which was the last ciphertext block of the previous packet). He also knows the last ciphertext block of the last packet that Bob encrypted, which is the IV for the first block of the next packet that Bob will send. Chris XORs the two IV's, and then XORs the result with P to get P', and somehow he persuades Bob to encrypt P' and send the ciphertext to Alice. If the ciphertext that Bob sends is C^*, then Chris knows that his guess was correct, and otherwise he knows that his guess was incorrect.

This attack is not very practical because Chris has to compute P' and get Bob to encrypt it during the time between two packets. In addition, as in any chosen-plaintext attack, Bob has to be willing to encrypt a plaintext chosen by the adversary. However, whether or not this vulnerability has practical significance, resistance to chosen-plaintext attack is widely viewed as important, and so the authors of [11] argue that SSH-IPC needs to be modified.

The first modification they consider, which they call "no packet chaining" (NPC), changes IPC in two ways. First, a fixed padding is used for all packets. Secondly, the IV for a packet is always a newly-chosen pseudorandom bitstring. In other words, in going from IPC to NPC the randomization is shifted from the padding to the IV. The authors of [11] say that it is possible to prove that SSH-NPC preserves confidentiality against chosen-plaintext attacks (and also integrity under a suitable definition). However, they do not give the proof because they found an attack on SSH-NPC that caused them to modify the protocol once again.

The attack on SSH-NPC in [11] is what's called a reaction-response attack. The adversary intercepts two ciphertexts C_1 and C_2 that Bob sent to Alice, and from them composes a new ciphertext C that he sends to Alice. He is able to deduce some information about the plaintexts corresponding to C_1 and C_2 based on whether Alice accepts or rejects C. (We omit the details.) This type of attack would be ruled out by a proof of security against chosen-ciphertext attacks, but not by the proof of security of SSH-NPC against chosen-plaintext attacks.

Then Bellare, Kohno, and Namprempre modify SSH-NPC in such a way as to preclude their reaction-response attack. The new protocol, which they denote SSH-$NPC, differs from SSH-NPC in that randomness is restored to the padding, which is newly generated for each packet. The big advantage of SSH-$NPC is that it prevents not only their attack on SSH-NPC, but in fact *any* chosen-ciphertext attack. That is, [11] contains a proof of security of SSH-$NPC against chosen-ciphertext attacks. The authors acknowledge that their attacks against SSH-IPC and SSH-NPC are theoretical and not very practical. However, they advise replacing SSH-IPC by SSH-$NPC on the grounds that at little additional cost one gets a protocol that's *provably* secure against a wide range of attacks, including the ones in their paper.[7]

3.6. The Albrecht–Paterson–Watson attack on SSH-$NPC.

The attack in [2] applies to any of the versions of SSH discussed above — SSH-IPC, SSH-NPC, and SSH-$NPC — even though the last of these was "proved secure" in [11]. The Albrecht–Paterson–Watson attack enables the adversary to learn the first 32 bits of a target ciphertext block (at least a non-negligible proportion of the time). This is not as bad as recovering the key or the complete plaintext, but it is nevertheless a very serious weakness to have in a system for authenticated encryption. The attack is practical; it is of much greater concern in the real world than the theoretical attack on SSH-NPC in [11] that SSH-$NPC was designed to avoid.

Again suppose that Bob sends Alice some packets that are encrypted and authenticated using one of the above variants of SSH. Let C^* denote the target ciphertext block that Chris wants to know about; perhaps the first 4 bytes of the corresponding plaintext

[7] The authors of [11] offer several other replacements for SSH-IPC, including one they call SSH-CTR, where the encryption is performed using the "counter" mode of operation.

contain some important personal data. During a pause in the transmission Chris sends Alice:

(1) a random block C_0 for the initialization vector (except in the case of SSH-IPC, when the IV is the last block of ciphertext in the previous encrypted packet);
(2) the block $C_1 = C^*$;
(3) a continuous stream of random bytes that make up the ciphertext blocks $C_2, C_3, C_4, \ldots$.

Chris simply waits until Alice rejects the message, and this tells him what he wants to know. This is because the first 4 bytes of Alice's decryption P of C^* are in the packet-length field, and so Alice uses that information to determine after what C_j to stop decrypting and verify the supposed tag C_{j+1}. Of course, the verification fails, so she rejects the message. The actual decryption of the ciphertext block C^* of the earlier packet is not P, because the IV in the earlier transmission of C^* was not C_0, but rather the previous ciphertext block C'. Of course, Chris knows both C_0 and C', and so can compute the first 32 bits of the true decryption of C^*, which is $P \oplus C_0 \oplus C'$.

Remark 12. The packet-length is generally required to be less than some bound and a multiple of the block-length. In the most common implementation of SSH, these two conditions restrict the number in the 4-byte packet-length field to 2^{14} out of 2^{32} possible integers. Thus, each Albrecht–Paterson–Watson attack has only a 2^{-18} chance of succeeding. Moreover, since Alice will terminate her connection with Bob if the decrypted text does not have the correct format, Chris will not be able to iterate the attack with the same target ciphertext.

Does this mean that the Albrecht–Paterson–Watson attack is not practical? Hardly. First, Chris might be able to iterate the attack over many sessions if some fixed plaintext (such as a password) is being encrypted in multiple sessions [2]. Second, Chris might be simultaneously attacking a large number of users, and he might consider himself successful if he captures 32 bits of personal data from any of them. (See §5 of [55] for an analogous discussion of the distinction between universal and existential key recovery.) If Chris is attacking a million users, he has a success probability of $1-\left(1-2^{-18}\right)^{10^6} \approx 98\%$.

A noteworthy feature of the Albrecht–Paterson–Watson attack is that it is of the same general type as the Bellare–Kohno–Namprempre one. Both are reaction-response attacks, that is, they use information obtained from Alice's response to the adversary. In other words, the failure of the security model in [11] was due not to a new type of attack, but rather to a clever variant on the same type of attack that the security proof was supposed to guarantee could not happen. The authors of [2] explain this paradoxical situation as follows:

> Some readers might wonder at this point how we would be able to attack a variant of SSH that was already proven secure in [11]. The basic reason is that, while the authors of [11] recognize that the decryption operation in SSH is not "atomic" and might fail in various ways, their security model does not distinguish between the various modes of failure when reporting errors to the adversary. Moreover, their model does not explicitly take into account the fact that the amount of data needed to complete the decryption operation is itself determined by data that has to be decrypted (the length field).

Remark 13. Paterson and Watson [79] developed a security model for authenticated encryption that more accurately captures the manner in which encryption is described in the SSH standards. They prove that SSH-CTR is secure in this model, thereby providing some assurances that implementations of SSH-CTR resist the attacks described in [2].

Remark 14. In [28], the authors describe successful attacks on the authenticated encryption schemes in two widely-used communications protocols — SSL/TLS and IPsec. More recently, distinguishing attacks on the TLS authenticated encryption scheme have been presented in [78]. The gap between theory and practice that is demonstrated by these attacks is strikingly similar to the gap exposed by the Albrecht–Paterson–Watson attack on SSH-$NPC. In a section of [28] titled "Cultures in Cryptography" the authors make the following comments:

> It might seem strange that there can be such an obvious discrepancy between the theory and practice of cryptography.... Provable security has become an important research area in modern cryptography but is still met with skepticism by many in the practical community. This is understandable considering the types of attack that we outline here. Practitioners might think provable security results provide an absolute statement of security, especially if they're presented in such a manner. When they later discover that a scheme is insecure because of an attack outside the security model, this might damage their confidence in the whole enterprise of provable security.

4. Leakage Resilience

4.1. **Brief history of side-channel attacks.** So-called "side-channel" attacks on cryptographic systems — based on information leaked as a consequence of physical processes or implementation features, including power consumption, electromagnetic radiation, timing of operations, induced faults, and error messages — go back a long way. It's a rich and colorful history that we shall just touch upon briefly (more details can be found in [5, 59]; see also [75, 89, 101]). In the West it was at Bell Labs during World War II where engineers first realized that the electromagnetic emanations from cryptographic hardware could be observed at a distance and used to uncover the plaintext. During the Cold War the U.S. and Soviet sides devoted considerable resources both to carrying out and to protecting themselves from this type of attack. And side-channel eavesdropping was also carried out among allies:

> In 1960, after the [British] Prime Minister ordered surveillance on the French embassy during negotiations about joining the European Economic Community, his security service's scientists noticed that the enciphered traffic from the embassy carried a faint secondary signal, and constructed equipment to recover it. It turned out to be the plaintext... ([5], p. 525; see also [101], pp. 109-112)

A recently declassified NSA report [75] makes it clear that by the 1950s side-channel attacks were perceived as a major problem. The article describes how extensive the leakage of radiation was:

> At the same time that we were trying to cope with the 131-B2 mixer, we began to examine every other cipher machine. *Everything* tested radiated, and radiated rather prolifically. ([75], p. 28, emphasis in original)

In addition, these tests revealed the possibility of power analysis attacks (which were not developed in the open cryptographic literature until almost a half century later in [57]):

> With rotor machines, the voltage on their power lines tended to fluctuate as a function of the number of rotors moving, and so a fourth phenomenon, called *power line modulation*, was discovered. ([75], p. 28)

Another disturbing discovery in those early years was that when one goes to great lengths to introduce countermeasures against one type of side-channel attack, that might just increase vulnerability to another type [75]. This situation, where "no good deed goes unpunished," persists to the present day.

According to [59], perhaps the first published mention of electromagnetic radiation as a computer security issue was in an article by R. L. Dennis [29] in 1966. The general public became aware of the problem in 1985 when the Dutch researcher Wim van Eck published an article [97] describing how one can use a device that's easy to build (essentially a modified TV set) to reconstruct the image that's on a TV monitor in the next room. This made quite a sensation, and the BBC ran a demonstration by Van Eck of what later came to be known as *Van Eck phreaking.* In the 1980s and 1990s Van Eck phreaking was the most widely publicized example of a side-channel attack on a supposedly secure set-up. It became so well known that it was even a major plot element in *Cryptonomicon* [94], which the literary critic Jay Clayton [23] called the "ultimate geek novel" and which has been quite popular among math, computer science, and crypto people.

The first presentation of a side-channel attack at a major cryptography conference was Paul Kocher's Crypto 1996 paper [56] about timing attacks; it was followed two years later by his power analysis attacks [57]. It was Kocher's papers that first brought these attacks to the center of attention of academic researchers on cryptography.

But even after Kocher's work, several years went by before any attempt was made to take side-channel attacks into account in the formal models of security. Theoreticians continued to concentrate their efforts on preventing attacks that were probably much less serious threats. This was a source of frustration for many of the practitioners who were trying to better understand side-channel attacks:

> Compared to the large number of minor and highly theoretical vulnerabilities of cryptographic primitives and protocols discussed in much of the current computer science literature, compromising emanations are a risk of practical interest that has so far [as of 2003] remained mostly undocumented. ([59], p. 133)

Despite the complaints of applied cryptographers, theoreticians continued to have confidence in the classical theoretical models. During this period the self-confidence of researchers and their faith in the provable security paradigm were forcefully articulated by Victor Shoup [91]:

> This is the preferred approach of modern, mathematical cryptography. Here, one shows with mathematical rigor that any attacker that can break the cryptosystem can be transformed into an efficient program to solve the underlying well-studied problem (e.g., factoring large numbers) that is widely believed to

> be very hard. Turning this logic around: if the "hardness assumption" is correct as presumed, the cryptosystem is secure. This approach is about the best we can do. If we can prove security in this way, then we essentially rule out all possible shortcuts, even ones *we have not yet even imagined*. The only way to attack the cryptosystem is a full-frontal attack on the underlying hard problem. Period. (p. 15; emphasis in original)

Ironically, this bold statement came in 1998, just as the cryptographic research community was becoming increasingly aware of the side-channel threat. It turned out that provably secure cryptosystems could be successfully attacked by "shortcuts" that had been around for many years. And new side-channel attacks were continually being developed. For example, in 2001 Manger [64] mounted a successful chosen-ciphertext attack on a version of RSA encryption in which he used information learned when ciphertexts are rejected — such as the time it took the decryption to fail or the nature of the error messages that were returned.

It is no wonder that some practitioners reacted with dismay to statements such as Shoup's (quoted above). They concluded that theoreticians were burying their heads in the sand by insisting on the rock-solid reliability of their models.

Finally, seven years after Kocher's timing attacks paper and two decades after the Van Eck phreaking paper, the first work appeared [71] that attempted to incorporate side-channel attacks into a theoretical model of security. The authors of [71] described what they called a "crisis" in complexity-theoretic cryptography:

> Complexity-theoretic cryptography considers only abstract notions of computation, and hence cannot protect against attacks that exploit the information leakage (via electromagnetic fields, power consumption, etc.) inherent in the physical execution of any cryptographic algorithm. Such physical observation attacks bypass the impressive barrier of mathematical security erected so far, and successfully break mathematically impregnable systems. The great practicality and the inherent availability of physical attacks threaten the very relevance of complexity-theoretic security.

One of the most dramatic examples of a successful side-channel attack is also one of the most recent. A group at the Norwegian University of Science and Technology were able to demonstrate a complete break of commercial implementations of quantum key exchange (see [70, 63]). By shining a small laser at Bob's detector, they could disable the detector, intercept the quantum key bit, and convert it to a classical bit for Bob that agrees with Alice's bit. They showed how to construct "an eavesdropping apparatus built from off-the-shelf components [that] makes it possible to tracelessly acquire the full secret key" [63]. Although quantum key agreement supposedly is "provably secure" — in fact, in the same strong, information-theoretic sense as one-time pads — the laser attack in [63], which revealed the protocol to be completely insecure, was simply outside the security model in the proof.

4.2. **The response – leakage resilience.** Micali and Reyzin [71] wrote the first article in the theoretical literature that attempted to deal with side-channel attacks in a comprehensive way. Their article, which was posted in 2003 and published in 2004, announced the following ambitious agenda:

> To respond to the present crisis, we put forward physically observable cryptography: a powerful, comprehensive, and precise model for defining and delivering cryptographic security against an adversary that has access to information leaked from the physical execution of cryptographic algorithms. Our general model allows for a variety of adversaries. In this paper, however, we focus on the strongest possible adversary, so as to capture what is cryptographically possible in the worst possible, physically observable setting. In particular, we
>
> • consider an adversary that has full (and indeed adaptive) access to any leaked information;
>
> • show that some of the basic theorems and intuitions of traditional cryptography no longer hold in a physically observable setting; and
>
> • construct pseudorandom generators that are provably secure against all physical-observation attacks.
>
> Our model makes it easy to meaningfully restrict the power of our general physically observing adversary. Such restrictions may enable schemes that are more efficient or rely on weaker assumptions, while retaining security against meaningful physical observations attacks.

Reading these paragraphs could give the practical cryptographer false hopes. Despite the extravagant promises, the paper [71] contains no concrete construction of any pseudorandom generator, let alone one that resists side-channel attacks. Nor does [71] describe any techniques that "make it easy to meaningfully restrict the power" of the side-channel attacker.

Rather, what [71] does contain is "philosophy" in the sense that Rogaway uses that term in the passage quoted in the Introduction — that is, a high-level discussion of some of the general issues involved in security against a "physically observing adversary." For example, the authors note that the classical theorem about pseudorandom generators that says that unpredictability and indistinguishability are equivalent (in the earlier theoretical models) no longer holds if one allows for adversaries that observe traces of the physical act of computation. It is possible that such an adversary may be unable to make any prediction about the output, but nevertheless can distinguish the output from random output.

Despite readers' disappointment in finding that everything in [71] is on a broad philosophical plane and that they are no better able to develop countermeasures or evaluate protocols for resistance to side-channel attacks after reading the paper than they were before reading it, the authors deserve credit for opening up a new area of research and for having the courage to criticize the earlier theoretical models that ignored side-channel adversaries.

A similar remark applies to another widely-cited article that also attempted to present a broad theoretical framework for analyzing resistance to side-channel attacks. The paper [93] by Standaert, Malkin, and Yung, which was posted in 2006 but not published until 2009, promises to present

> ...a framework for the analysis of cryptographic implementations that includes a theoretical model and an application methodology.... From a practical point of view, the model implies a unified methodology for the analysis of side-channel key recovery attacks. The proposed solution allows getting rid of most of the

> subjective parameters that were limiting previous specialized and often ad hoc approaches in the evaluation of physically observable devices.

But again the main contribution of [93] is a series of observations and theorems of a very general nature; by no means does it "get rid of [the need for] ad hoc approaches." For example, the authors emphasize the rather obvious point that there are two aspects of information leakage that must be measured separately: the gross amount of leakage in the information-theoretic sense, and the value of this information to an attacker who's trying to uncover the secret key. The paper includes some 2- and 3-dimensional pictures of Gaussian leakage distributions (consisting of two or more bell curves centered at different points, the relevance of which to cryptography is far from clear), and one section explains how the validity of one of the theorems can be seen in a simplified side-channel attack on a reduced block cipher. This shows that the authors are attempting to establish a connection between their theoretical remarks and the real world; nevertheless, in reality this paper is of no greater help than [71] for the practitioner.

Although an implementer would search in vain for anything of practical use in [71] or [93], among theoreticians [71] and [93] are considered landmark papers. In [47, 49] we read:

> In the past few years, cryptographers have made tremendous progress toward modeling security in the face of such information leakage [71, 93], and in constructing *leakage-resilient* cryptosystems secure even in case such leakage occurs.

This use of the words "tremendous progress" to refer to the philosophical essays [71, 93] seems to us to be a bit of hyperbole.

4.3. **The bounded retrieval model.** Fortunately, some of the more recent papers have presented not only general definitions and theoretical discussions, but also some clever and potentially useful cryptographic constructions. We turn our attention to these constructions, and start by giving an informal explanation of the so-called "bounded retrieval model" that was developed in [21, 27, 35] (see also [4, 3]).

The idea of the bounded retrieval model is simple and elegant. Suppose that an adversary is able somehow to acquire a certain amount of secret key data. One countermeasure is to increase the sizes of all of the parameters of the cryptosystem so that the secret key becomes larger than what the adversary is likely to be able to capture. This would almost certainly result in extremely inefficient cryptography.

But what if Bob could generate and store a very large number M of secret keys, all corresponding to the same public key, and in carrying out a protocol he would use only one of those keys or a small randomly chosen subset of them (his correspondent Alice would send him a clue that would tell him which of them he needs to use)? Suppose also that an adversary cannot compromise the execution of the protocol using knowledge of other secret keys.

Bob's stable of secret keys could be huge; M is limited only by available storage and the time required to generate all of the M keys during the initial set-up stage. Meanwhile, the size of the public key and everything else (ciphertexts, signatures, shared keys, etc.) — and the amount of time to carry out protocols — would be essentially fixed and independent of the number of secret keys. This is what the bounded retrieval model requires. Now the adversary's task becomes much more difficult, because any secret key material he acquired would be unlikely to be sufficient for breaking a given execution of the protocol. However, the cost to Bob is quite reasonable.

In [4, 3] the authors describe how a basic ingredient that they call a "hash proof system" (HPS) can be used to construct encryption schemes that are "provably secure" unless the adversary can acquire a tremendous amount of secret key material. The HPS is identity-based so as to avoid the need to have a large amount of public key material.

4.4. **A construction used to implement the bounded retrieval model.**

In an identity-based hash proof system, Alice uses one of Bob's identities to send him a type of ciphertext from which he can extract a shared secret, that is, an essentially random number or group element k that Alice and Bob can later use for encryption, authentication, or other purposes. We now describe one of the constructions of an identity-based HPS that is given in [3] (originally due to Gentry [40]).

Let $\mathbb{G}$ be a group of prime order n in which the discrete logarithm problem is hard (and, for the security proof, one has to assume that a related problem called the "truncated augmented bilinear Diffie-Hellman exponent problem" is also hard). The group $\mathbb{G}$ is assumed to have an efficiently computable non-degenerate bilinear pairing $e(x, y)$ that maps to some group $\mathbb{G}_T$. Let $g \in \mathbb{G}$ be a fixed non-identity element. All of these system-wide parameters are publicly known. In addition, Bob chooses a random group element $h \in \mathbb{G}$ and a random mod-n integer α, and sets $g_1 = g^\alpha \in \mathbb{G}$. The group elements h, g_1 form Bob's public key, and α is his master secret key. Bob's publicly known identity is a mod-n integer ID. There are actually M different publicly known identities ID_i, $1 \leq i \leq M$. In practice, the simplest thing is to set ID_i equal to $H(\mathrm{ID}, i)$ mod n, where H is a publicly known hash function. One supposes that none of the mod-n integers ID_i is equal to α.

Here is how Bob generates a large number of secret keys using α. Each key sk_i, $1 \leq i \leq M$, corresponds to a different randomly chosen mod-n integer r_i and is computed as follows: Bob sets $h_i = (hg^{-r_i})^{1/(\alpha - \mathrm{ID}_i)}$, and sets $sk_i = (r_i, h_i)$.

Now suppose that Alice wants to agree with Bob on a shared secret k. She chooses a random mod-n integer s and sets $k = e(g, h)^s \in \mathbb{G}_T$. She also computes $v = e(g, g)^s$, and for some i she computes $u = g_1^s g^{-s\,\mathrm{ID}_i}$; she sends (i, u, v) to Bob. Bob can recover k as $e(u, h_i)v^{r_i}$, using his i-th secret key $sk_i = (r_i, h_i)$. Namely, it is an elementary exercise using the properties of a bilinear pairing to show that $e(u, h_i)e(g, g)^{sr_i} = e(g, h)^s$.

This "hash proof system" is a basic ingredient, but is not itself a complete protocol. The main use of this tool in [3] ("Improvement II" in the Introduction) is for identity-based encryption. In that case Bob uses a randomly selected subset of the sk_i in order to decipher Alice's message. The details of converting an HPS to an encryption scheme do not concern us here.

Unfortunately, despite their elegance and their promise, the bounded retrieval model implementations in [3] have a serious limitation that must be considered by anyone who is interested in practical deployment. Namely, as the authors acknowledge, they are assuming that leakage cannot occur in the key generation stage. That is, they are supposing either that Bob has already generated and stored his sk_i before any adversary was observing him, or else that when generating keys he uses a system that is hardened against all known side-channel attacks.

However, suppose that this assumption fails. For example, Bob, realizing that he's being observed by an adversary, decides to increase security by doubling his number of secret keys. He generates sk_i, $i = M+1, \ldots, 2M$, while the adversary is watching. In that case a relatively small amount of leakage could be fatal. That is because in every computation of an sk_i Bob performs an exponentiation, and exponentiations are particularly vulnerable to timing, power-consumption, and other side-channel attacks. If the adversary learns the value of a single exponent $1/(\alpha - \mathrm{ID}_i)$ (and knows i),[8] he can easily find the master secret key α. Thus, the system, while it might be provably resistant to side-channel attacks that occur during execution of protocols, is extraordinarily vulnerable to side-channel attacks during key generation. Note that the danger of leakage is magnified because the number of times a secret key is computed — using, of course, the master secret α in each of these computations — is much larger than it would be in a conventional public-key encryption scheme. This is an example of how countermeasures against one type of side-channel attack (in this case leakage during protocol execution) sometimes actually increase vulnerability to another type (leakage during key generation).

A similar remark applies to the other two constructions in [3] used to implement hash proof systems. For example, in the construction based on quadratic residuosity the master secret key is the factorization of an RSA modulus $N = pq$. Bob generates his large supply of secret keys by doing a lot of squareroot extractions modulo N, and this involves arithmetic modulo p and q. So a side-channel attacker who observes the key generation stage will have plenty of opportunity to try to get enough leakage to determine p and q.

4.5. **Schnorr-ℓ.** Not all leakage-resilience results assume that no leakage occurs during key generation. We next discuss a construction of signatures that was developed for leakage-resilience purposes independently by Katz [47] and by Alwen, Dodis, and Wichs [4]. They generalized a version due to Okamoto [77] of Schnorr's discrete-log-based signatures. The generalization depends on a parameter ℓ, and we shall refer to the scheme as "Schnorr-ℓ." It turns out that the system is "provably secure" if up to roughly $\frac{1}{2}\ell\lceil\log_2 n\rceil$ bits are leaked (more precisely, $\frac{1}{2}$ should be replaced by $\frac{1}{2} - \frac{1}{2\ell} - \varepsilon$), and leakage is permitted during key generation.

In Schnorr-ℓ a group $\mathbb{G}$ of prime order n in which the discrete log problem is hard is a system-wide parameter.

Key Generation: Bob chooses ℓ random elements $g_1, \ldots, g_\ell \in \mathbb{G}$ and ℓ random mod-n integers $x_1, \ldots, x_\ell$, and sets $X = \prod_{i=1}^{\ell} g_i^{x_i}$. His public key consists of $g_1, \ldots, g_\ell$, and X; his secret key is the ℓ-tuple of x_i.

Signing a Message: To sign a message m for Alice, Bob chooses ℓ random mod-n integers $r_1, \ldots, r_\ell$ (a different ℓ-tuple for each message he signs) and computes the group element $R = \prod_{i=1}^{\ell} g_i^{r_i}$. He then computes $c = H(R, m)$, where H is a hash function. Finally, he computes the mod-n integers $y_i = cx_i + r_i$, $i = 1, \ldots, \ell$. His signature is $(R, y_1, \ldots, y_\ell)$.

[8] It is not even essential to know i. If the adversary finds two different exponents $e_i = 1/(\alpha - \mathrm{ID}_i)$ and $e_j = 1/(\alpha - \mathrm{ID}_j)$ but doesn't know i or j, he can set $z = 1/e_i - 1/e_j$ and then compute $\mathrm{ID}_i = H(\mathrm{ID}, i)$, $i = 1, \ldots, M$, and compare ID_j with $z + \mathrm{ID}_i$ until he gets a match, at which point he'll know the master secret $\alpha = \mathrm{ID}_j + 1/e_j$.

Verifying a Signature: When Alice receives the signature $(R, y_1, \ldots, y_\ell)$ on the message m from Bob, she first computes $c = H(R, m)$ and then uses Bob's public key to compute $Y = \prod_{i=1}^{\ell} g_i^{y_i}$ and also $X^c R$. If these two elements are equal, she accepts the signature.

From a practical standpoint a crucial drawback of this scheme is similar to the problem with the HPS that was discussed in the previous subsection. Namely, suppose that the assumed bound on leakage fails by a factor of two or three. Suppose that the adversary is able to determine all the $\ell \lceil \log_2 n \rceil$ bits of a single random ℓ-tuple $(r_1, \ldots, r_\ell)$ used for any one of Bob's messages. Then the static secret key $(x_1, \ldots, x_\ell)$ can be immediately computed, because $x_i = c^{-1}(y_i - r_i)$, and $c = H(R, m)$ and y_i are obtained from the message and its signature. Thus, this scheme has the property that revealing the random secret used to generate a single signature is equivalent to revealing the secret key and thereby losing everything. In the context of information leakage, this is not a desirable property for a signature scheme to have. It would be much better to have some control on the damage that is done if one's bound on the adversary fails by a small amount, such as a factor of two or three.

In certain cases — such as the attack on Schnorr signatures by Nguyen and Shparlinski [76] (see Remark 13) — the mode of operation of a side-channel attack is to gather data that give conditions or constraints satisfied by the secret bits. For a while the data are insufficient for the adversary to get anything useful at all. But then a threshold is reached, and suddenly a large number of secret bits — perhaps all of them — are revealed.

The situation is similar to the spy vs. spy stories by authors such as John le Carré where an agent is transmitting from a secret location. The enemy is frantically trying to triangulate his position, but if he changes frequencies within a certain time period, he's completely safe. If he lingers beyond that time, he's dead meat; that's what happens, for example, in [60].

In a similar way, the number of secret bits a side-channel attacker can compute might increase as a sharply discontinuous function of the amount of data he's able to gather. For example, in Coron's differential power analysis attack [25] on the computation of dP where d is a secret integer and P is a publicly-known elliptic curve point, a number of power traces are collected, each one for a different P. The traces are used to verify a guess for a single bit d_1 of d. Provided that there are sufficiently many traces, this key bit can be determined. Subsequently, the *same* traces are used to verify a guess for a second bit d_2 of d; provided that d_1 was correctly found, the probability of success in determining d_2 has essentially the same dependence on the number of traces collected as did the success rate for determining d_1. The process is repeated until all the bits of d have been determined. Since success in determining a particular bit is contingent on correctly finding the previous bits, the attack is likely to succeed in determining all of d provided that enough power traces were gathered for the initial stage to be successful. Although the secret might be secure — even provably secure — when the amount of data gathered remains below a certain bound, it is possible that even a small violation of that bound will lead to a total loss of security.

On the other hand, there are certain side-channel attacks — such as cold-boot memory leakage — that do not proceed in the manner just described. In such an attack the adversary can examine the memory of Bob's computer for traces of secret bits that were involved in his recent signature computations. The data available to the attacker are frozen once and for all. In contrast to timing, power consumption, and electromagnetic radiation, it is arguably realistic to model cold-boot attacks by bounds such as in Katz's security proof for Schnorr-ℓ.

Let us look at a specific case of the Schnorr-ℓ signature scheme, see what guarantee Katz gives us, and also ask what price in efficiency must be paid to get this added security. Theorem 4 of [47] says that if the discrete log problem is hard in $\mathbb{G}$, then for any $\varepsilon > 0$ Schnorr-ℓ remains secure in the event of leakage of up to $(\frac{1}{2} - \frac{1}{2\ell} - \varepsilon)$ times the bitlength of a secret key. Katz calls the scheme "fully leakage-resilient" because there are no restrictions on which bits can be leaked — they can be from past and present secret random $r = (r_1, \ldots, r_\ell)$ and from the static secret key $x = (x_1, \ldots, x_\ell)$.

Suppose, for example, that Schnorr is implemented with elliptic curves over a 160-bit finite field. Let us take $\ell = 5$. Then the 5-tuples x and r each have about 800 bits. Katz's Theorem 4 tells us that the scheme is secure even if roughly 300 bits are leaked. If more than 320 bits are leaked, then the theorem says nothing, and if the 800 bits of a single r are leaked, then the scheme is broken.

Suppose that Bob has signed 100 messages in quick succession. Then his computer's memory might contain traces of the 80,000 secret bits of the different r's as well as the 800 secret bits of the static key x. Katz's theorem provides assurances only if at most 300 of these 80,800 bits are leaked during a memory attack. This is not very comforting.

Comparing Schnorr-5 with traditional Schnorr (i.e., Schnorr-1), we see that this marginal protection comes with a significant performance penalty: (1) 800-bit private key for Schnorr-5 versus 160 bits for Schnorr-1, (2) 960-bit versus 320-bit signature size, (3) 800 versus 160 random bits needed per signature, and (4) five rather than one exponentiation needed per signature.

In summary, Schnorr-5 probably has a slight security advantage in a few types of side-channel attacks and no advantage in most attacks.

4.6. **No good deed goes unpunished.** As mentioned in §4.1, implementers who go to considerable effort to include countermeasures against some of the known side-channel attacks often find that they have inadvertently increased vulnerability to other types of attacks. A striking example of this phenomenon can be found in the work of Standaert [92], who demonstrated that his implementation of a provably leakage-resilient stream cipher designed by Dziembowski and Pietrzak [36] is in fact more susceptible to differential power analysis attack than a similar implementation of an unprotected AES-based stream cipher. A second example is the induced fault attack described by Joye et al. [43] on a variant of the RSA public-key encryption scheme that was designed by Shamir to guard against a different type of induced fault attack; the former attack was possible precisely because of the changes proposed by Shamir to the original scheme.

We discuss another example in the setting of elliptic curve cryptography, relying heavily on the excellent survey [37]. Suppose that in order to make timing, power consumption, and electromagnetic radiation attacks much more difficult, Bob follows the suggestion of Coron [25] and inserts dummy operations in ECC scalar multiplication: namely, he always doubles and adds, independently of the values of the bits. In addition, in order to thwart fault attacks [14], Bob institutes a point validity check (as suggested in [14]) and also a check on the coherence and consistency of the operations, as described in [32].

However, all three of these countermeasures potentially make Bob much more vulnerable to the safe-error attacks introduced in [44, 45]. For example, the adversary might attempt to introduce a slight error during a double-and-add step. If the error has no effect, then he knows that the step was a dummy one, and from this he can deduce that the true value of the corresponding bit was zero; if the error affects the outcome, then he knows that the double-and-add was a needed step and the secret key had a 1-bit. In addition, during the delicate process of introducing faults, the adversary can be guided by the output of the point validity check and coherence check, which give him valuable feedback about where his induced error landed in the algorithm. For a similar reason, the point validity check, although it can be helpful in protecting not only against fault attacks, but also against the use of a weak twisted elliptic curve (see [38]), makes Bob more vulnerable to the sign change attack in [17].

Thus, as emphasized in [37], the choice of countermeasures is a difficult balancing act and depends upon constantly updated information about diverse types of side-channel attacks. The relevance to this process of the theorems in the theoretical leakage-resilience literature is unclear. Despite the claims in [93], no one has come close to finding a way of "getting rid of... specialized and often ad hoc approaches in the evaluation of physically observable devices."

4.7. **Limitations of the formal definitions.** At first glance the leakage-resilience security models appear strong. In [3], for example, the attacker is allowed to learn not only the bits of secret key material (up to a certain bound), but also the bits of "any efficiently computable function of the secret key." However, upon closer examination one finds a mismatch between this notion of security and what one encounters in real-world side-channel attacks.

The model imposes a fixed bound on the attacker and does not distinguish between different forms that the leaked information might take. For this reason some realistic attacks are not covered. For example, in the Schnorr-ℓ signature scheme (see §4.5) suppose that the adversary can learn the least significant bit of every ephemeral secret key r_i. This might be possible because of the way in which exponentiation is implemented or because of a flaw in the pseudorandom bit generator. In this case presumably the attacker can learn much more than $\frac{1}{2}\ell\lceil\log_2 n\rceil$ bits of secret data. However, we would still like some assurance that Schnorr-ℓ is secure.

Remark 15. By a result of Nguyen and Shparlinski [76], the Schnorr scheme is known to be insecure if the *three* least significant bits of many ephemeral secrets are revealed. However, it is unknown whether or not it is insecure if the attacker learns just one least significant bit of each secret.

In leakage-resilient public-key encryption schemes the adversary is not permitted to perform side-channel attacks after seeing the target ciphertext. As explained in [3],

> ...we only allow the adversary to perform leakage attacks before seeing the challenge ciphertext.... [T]his limitation is inherent to (non-interactive) encryption schemes since otherwise the leakage function can simply decrypt the challenge ciphertext...

Thus, the security definition does not allow for partial information that the adversary might be able to glean while observing the device during the process of decrypting the target ciphertext. From a practical point of view this is a highly artificial restriction on the attacker.

More broadly, a fundamental shortcoming of the formal definitions of leakage resilience is that in certain types of side-channel attacks — such as power analysis and electromagnetic radiation — it is difficult (if not impossible) to measure how many useful bits of secret keying material are leaked. As a result, it is not clear whether the guarantees provided by a proof translate into concrete assurances of resistance to the many known side-channel attacks.

It is not surprising that side-channel attacks are difficult to encompass within a general model of the adversary's power. After all, the attacker's capabilities depend upon many things: proximity to the device, the physical security measures in place, and the architecture of the processor. The attack might exploit variations in time, power consumption, or electromagnetic radiation; it might measure responses to errors and induced faults; it might exploit information leaked about access patterns of a CPU's memory cache; it might use bugs or peculiarities in the programming; or it might use some combination of these.

In practice there is no avoiding the need for ad hoc countermeasures. One cannot expect to achieve a high level of resistance to side-channel attacks if one simply uses the formal definitions as a guide in designing protocols. On the contrary, the provably-secure leakage-resilient systems tend to be more complicated than their traditional counterparts, and so it is likely to be a more complicated task to develop the appropriate countermeasures for them. In this sense the use of a provably-secure leakage-resilient cryptosystem may be a step backwards from a security viewpoint.

5. Safety Margins

In the introduction to [48] the first reason Katz and Lindell give for the importance of definitions is "to better direct our design efforts" when developing a cryptographic protocol. Echoing the comment by Bellare and Rogaway [12] quoted in §1, they say that

> ...it is much better to define what is needed *first* and then begin the design phase, rather than to come up with a *post facto* definition of what has been achieved once the design is complete. The latter approach risks having the design phase end when the designers' patience is tried (rather than when the goal has been met), or may result in a construction that achieves more than is needed and is thus less efficient than a better solution.

This explanation seems reasonable and unexceptionable. However, implicit in the last sentence is some advice about protocol construction that in our view is very ill-conceived. Katz and Lindell are saying that to achieve the best solution one should exclude all features that are not needed to satisfy the definition. In many settings this is a bad idea.

It is regrettable that this viewpoint, which is based upon unwavering faith in the adequacy of one's security model and the validity of one's proof, seems to be widespread. It has sometimes led to over-confidence about removing safety features from protocols.

One of the basic concepts in applied science is that of a "safety margin." For example, in cryptography it has long been customary to regard 80 bits of security as the proper standard for short term security. In reality, the number of steps most attackers are prepared to carry out is much less than 2^{80}. However, it's prudent to allow for the possibility that incremental improvements will be found in the algorithm, that adversaries might "get lucky" and succeed somewhat faster than the expected running time, or that they might be blessed with the use of an unusually large botnet.[9]

Similarly, when cryptographers formulate security definitions, they usually allow the adversary considerably more power than might be feasible in real-world situations. For example, the adversary may be allowed to mount chosen-ciphertext attacks on an encryption scheme. In practice, the attacker might be able to learn only whether or not a decryption is valid. However, if resistance to chosen-ciphertext attacks is guaranteed, then one also has the assurance that the weaker attacks are not possible.

In protocol design it's also wise to have a safety margin. It's a good idea to include in one's protocol a certain number of "validation" steps that assure the user that the parameters and keys have been properly formed, that the protocol is being carried out in the proper time sequence, and so on. Even if no one has yet devised an attack that takes advantage of a violation, and even if the commonly used security models do not mandate such validations, history has shown that it's better to be safe than sorry.

One reason for including "extra" validation steps in a protocol is that they might help in the event of attacks that are not anticipated by the security model. For example, suppose that the digital signature standards had mandated that the sender include his public key in the message and the recipient verify that it's there. It was shown in [68] that this would have thwarted the standard Duplicate Signature Key Selection attacks. Or, alternatively, suppose that all signatures had to be timestamped, all certificates issued by a CA also had to be timestamped, and verification of a signature included checking that the certificate's timestamp precedes the one on the signature. Such a requirement — which might be an unreasonable imposition in some settings because a reliable clock is not always available — also would have prevented DSKS attacks. But neither inclusion of the public key in the message nor timestamping is required by the GMR security model; nor is either required by any of the commonly-used digital signature standards.

A second reason not to eliminate features that do not seem to be required by the security definition is that one has to allow for the possibility of flaws in the security proofs. An illustration of the pitfalls of prematurely dropping a validation step can be found in the story of HMQV, which was Hugo Krawczyk's attempt to design a provably secure and more efficient modification of the MQV key agreement protocol. In his Crypto 2005 paper [58] he claimed not only that HMQV had a proof of security (unlike the original MQV), but that it had greater efficiency because of the removal of a no longer necessary public-key validation step that had been put into MQV to prevent known attacks. According to Krawczyk, the security proof went through without that step, and so the validation could be dispensed with.

[9] One could argue that 80 bits of security no longer give much of a safety margin even in the short term. The factorization of a 768-bit RSA modulus by Kleinjung *et al.* [50] required about 2^{67} instructions. Bernstein *et al.* [13] will soon complete a cryptanalysis of 130-bit ECC that will have required approximately 2^{77} bit operations.

However, Menezes [67] showed that some of the HMQV protocols succumb to the same attacks that MQV would have if the public-key validation had not been put in. In addition, he found a fallacy in the security proof. There's a lesson to be learned from this episode. As commented in [53],

> Both Krawczyk and the referees on the [Crypto 2005] 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.

A third reason for safety margins in protocol design is that the assumptions underlying the security models are often not realistic. For example, Luby and Rackoff [62] proved that Feistel ciphers with three rounds are secure (in the sense of being indistinguishable from a random permutation) under the assumption that the underlying round function is a pseudorandom function; and a stronger pseudorandom property was proved if four rounds are used. However, the round functions used in practice are difficult to analyze, and one does not know how close they come to the ideal. For this reason Feistel constructions that are deployed have many more than four rounds. The number of rounds used is decided based not on the theoretical results in [62], but rather on intuition, experience, and the current state of cryptanalytic knowledge, as well as efficiency considerations.

6. Mathematical Models in Cryptography

In the introduction to [48], Katz and Lindell argue that the discrepancies between cryptographic models of security and the realities of practical cryptography are no different from the discrepancies that exist in any science.

> This possibility of a disconnect between a mathematical model and the reality it is supposed to be modeling is not unique to cryptography but is something that occurs throughout science.

They then make an extended analogy between the difficulties faced by theoreticians in cryptography and the challenges encountered by Alan Turing when he was developing a rigorous foundation for the theory of computability.

However, such analogies have limited validity. In the physical sciences, the mathematical models used in the early days of modern science have not so much failed as been improved upon to account for new discoveries. For example, the equations of Newton and Kepler are still valid on the scale of everyday life. If, however, we are working in astrophysics or nanotechnology, we must switch to relativistic and quantum models. New models are needed to accommodate the advances of science and technology.

Katz and Lindell claim that this is what happened in the case of side-channel attacks:

> It is quite common, in fact, for a widely-accepted definition to be ill-suited for some new application. As one notable example, there are encryption schemes that were proven secure (relative to some definition like the ones we have discussed above) and then implemented on smart cards. Due to physical properties of smart cards, it was possible for an adversary to monitor the *power usage*..., and it turned out that this information could be used to determine the key.

This misleadingly suggests that side-channel attacks came after the introduction of a new technology (smart cards) that didn't exist when the security models were devised. On the contrary, devices that are much older than smart cards have been susceptible to side-channel attacks, and such attacks have been known to the NSA since the 1950s and to the general public since 1985. As we saw, there was a gap of about twenty years between public awareness of the problem and the first attempt to incorporate side-channel attacks into security definitions.

Moreover, what we study in cryptography has a feature that is missing in the physical sciences — the human element. By definition, cryptography deals with the handling of information in the presence of adversaries — adversaries who should be assumed to be as powerful, clever, and devious as they are malicious.

Perhaps a better analogy with the models of theoretical cryptography would be some of the mathematical models one finds in the social sciences, such as economics. And instead of finding similarities between the work of theoretical cryptographers and that of Alan Turing, as Katz and Lindell do, we might find closer parallels in the mathematical models of David Li.

David Li is perhaps the most famous designer of mathematical models in the financial world. In 2000 in an article in the *Journal of Fixed Income*, Li devised a mathematical model that would predict the probability that a given set of corporations would default on their bond debt in rapid succession. According to an article [100] in *The Wall Street Journal* written three years before the financial meltdown of 2008,

> The model fueled explosive growth in a market for what are known as credit derivatives: investment vehicles that are based on corporate bonds and give their owners protection against a default. This is a market that barely existed in the mid-1990s. Now it is both so gigantic — measured in trillions of dollars — and so murky that it has drawn expressions of concern from several market watchers....
>
> The model Mr. Li devised helped estimate what return investors in certain credit derivatives should demand, how much they have at risk and what strategies they should employ to minimize that risk. Big investors started using the model to make trades that entailed giant bets with little or none of their money tied up. Now, hundreds of billions of dollars ride on variations of the model every day.
>
> "David Li deserves recognition," says Darrell Duffie, a Stanford University professor who consults for banks. He "brought that innovation into the markets [and] it has facilitated dramatic growth of the credit-derivatives markets."
>
> The problem: The scale's calibration isn't foolproof. "The most dangerous part," Mr. Li himself says of the model, "is when people believe everything coming out of it."

Out of fairness David Li should be commended for warning investors that they should not "believe everything coming out of" his model; see also [33]. (Similarly, cryptographers who support their protocols using "provable security" theorems deserve praise when they take care to warn practitioners that the theorems are only as good as the intractability assumptions and models of adversarial behavior that they are based upon.)

The incident that gave rise to this article in *The Wall Street Journal* and the warning about not "believing everything" was the loss of hundreds of millions of dollars by hedge funds because of the collateral effects of a downgrading of General Motors' debt in May 2005 — effects that had not been predicted by Li's model. Not surprisingly, the gently-worded warnings of David Li and *The Wall Street Journal* were not heeded, and the frenzied growth of mortgage-based collateralized debt obligations (CDO) continued. When the financial collapse finally came in September 2008, a flurry of articles such as [42, 90] described (occasionally with exaggeration) the contribution of David Li's models to bringing on the crisis.

The analogy between mathematical models in cryptography and David Li's models should not be taken too far. It is hard to conceive of reliance on faulty security models in cryptography ever leading to financial losses anything like what occurred when the mortgage-based CDO market collapsed. Most failures in cryptography just result in patches, not in major losses. Thus, even if one lacks faith in the common security models used in cryptography, one should not overreact. Just as there are many cases when badly flawed protocols have functioned for years without any significant problems, so also it is likely that little damage, at least in the short run, will result from reliance on security guarantees that are based on flawed models of adversarial behavior.

7. Conclusions

(1) Despite its elegance and intuitive appeal, even in the single-user setting the Goldwasser–Micali–Rivest definition of a secure signature scheme does not account for all potentially damaging attacks.

(2) There is considerable disagreement and uncertainty about what the "right" definition of security is for such fundamental concepts as general-purpose hash functions, encryption, and message authentication.

(3) It is debatable whether or not related-key attacks are significant in practice and should be incorporated into security models. But in any event attempts to formalize what it means for a symmetric-key encryption scheme to be secure against such threats as differential cryptanalysis and related-key attacks have not been successful. And in the case of reaction-response attacks on SSH, a security model under which SSH could be proved to be safe from these attacks turned out to have a fatal gap.

(4) The security models in the leakage resilience literature might be able to handle a few of the practical types of side-channel attacks, but they are woefully inadequate for many of them.

(5) As we saw in §4.4 and §4.6, a provably leakage-resilient scheme might be even more vulnerable to certain types of side-channel attacks than a more efficient scheme that is not provably leakage-resilient.

(6) Resistance to side-channel attacks can be achieved only by extensive ad hoc testing, and must be continually reevaluated in light of new developments. The provable security results on leakage resilience in the theoretical literature should never be relied upon as a guarantee.

(7) Whenever possible, standards should call for protocols to include verification that all keys and parameters are properly formed. Despite the advice given in the

introduction to [48], such validation steps should not be omitted simply because they are unnecessary in order to satisfy a certain formal security definition.

(8) One of the criteria for evaluating cryptographic primitives and protocols should be whether or not efficient methods are available for verifying proper formation of parameters and keys.

(9) Mathematical models in cryptography can play an important role in analyzing the security of a protocol. However, it is notoriously difficult to model adversarial behavior, and no security model can provide a guarantee against all practical threats.

Finally, we wish to reiterate what we said in the Introduction: good definitions are important if one wishes to discuss security issues with clarity and precision. Any claims about the security of a protocol must be stated using well-defined criteria, and this is true whether or not one uses formal reductionist security arguments to support those claims. Our object in this paper is not to deny the need for definitions — and it is certainly not to justify vague or sloppy arguments for security. Rather, our work highlights the crucial role that concrete cryptanalysis and sound engineering practices continue to play in establishing and maintaining confidence in the security of a cryptosystem.

Acknowledgments

We wish to thank Greg Zaverucha for bringing reference [7] to our attention and for giving us feedback on the first draft of the paper, Dan Bernstein for correcting a flaw in a previous version of Remark 2, and Serge Vaudenay for informing us about reference [99]. We also thank Dale Brydon, Sanjit Chatterjee, Carlos Moreno, Kenny Paterson, Francisco Rodríguez-Henríquez, Palash Sarkar, and Gaven Watson for commenting on the earlier draft, and David Wagner for his comments on Section 2. In addition, we thank Ann Hibner Koblitz for helpful editorial and stylistic suggestions, and Gustav Claus for providing a model of Bubba's dialect of English.

References

[1] M. Albrecht, P. Farshim, K. Paterson, and G. Watson, On cipher-dependent related-key attacks in the ideal-cipher model, http://eprint.iacr.org/2011/213.

[2] M. Albrecht, K. Paterson, and G. Watson, Plaintext recovery attacks against SSH, *IEEE Symposium on Security and Privacy*, IEEE Computer Society, 2009, pp. 16-26.

[3] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs, Public-key encryption in the bounded-retrieval model, *Advances in Cryptology — Eurocrypt 2010*, LNCS 6110, Springer-Verlag, pp. 113-134.

[4] J. Alwen, Y. Dodis, and D. Wichs, Leakage-resilient public-key cryptography in the bounded-retrieval model, *Advances in Cryptology — Crypto 2009*, LNCS 5677, Springer-Verlag, pp. 36-54.

[5] R. Anderson, *Security Engineering*, 2nd ed., Wiley, 2008.

[6] M. Bellare and D. Cash, Pseudorandom functions and permutations provably secure against related-key attacks, *Advances in Cryptology — Crypto 2010*, LNCS 6223, Springer-Verlag, 2010, pp. 666-684.

[7] M. Bellare and S. Duan, Partial signatures and their applications, http://eprint.iacr.org/2009/336.pdf.

[8] M. Bellare, O. Goldreich, and A. Mityagin, The power of verification queries in message authentication and authenticated encryption, http://eprint.iacr.org/2004/309.pdf.

[9] M. Bellare, D. Hofheinz, and E. Kiltz, Subtleties in the definition of IND-CCA: When and how should challenge-decryption be disallowed?, http://eprint.iacr.org/2009/418.pdf.

[10] M. Bellare and T. Kohno, A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PFRs, and applications, *Advances in Cryptology — Eurocrypt 2003*, LNCS 2656, Springer-Verlag, 2003, pp. 491-506.

[11] M. Bellare, T. Kohno, and C. Namprempre, Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the encode-then-encrypt-and-MAC paradigm, *ACM Transactions on Information and System Security*, **1** (2004), pp. 206-241.

[12] M. Bellare and P. Rogaway, Entity authentication and key distribution, *Advances in Cryptology — Crypto '93*, LNCS 773, Springer-Verlag, 1994, pp. 232-249; full version available at http://cseweb.ucsd.edu/~mihir/papers/eakd.pdf.

[13] D. J. Bernstein, H.-C. Chen, C.-M. Cheng, T. Lange, R. Niederhagen, P. Schwabe, and B.-Y. Yang, ECC2K-130 on NVIDIA GPUs, *Progress in Cryptology — Indocrypt 2010*, LNCS 6498, Springer-Verlag, 2010, pp. 328-346.

[14] I. Biehl, B. Meyer, and V. Müller, Differential fault attacks on elliptic curve cryptosystems, *Advances in Cryptology — Crypto 2000*, LNCS 1880, Springer-Verlag, 2000, pp. 131-146.

[15] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, and A. Shamir, Key recovery attacks of practical complexity on AES variants with up to 10 rounds, *Advances in Cryptology — Eurocrypt 2010*, LNCS 6110, Springer-Verlag, 2010, pp. 299-319.

[16] S. Blake-Wilson and A. Menezes, Unknown key-share attacks on the station-to-station (STS) protocol, *Public Key Cryptography — PKC 1999*, LNCS 1560, Springer-Verlag, 1999, pp. 156-170.

[17] J. Blömer, M. Otto, and J.-P. Seifert, Sign change fault attacks on elliptic curve cryptosystems, *Fault Diagnosis and Tolerance in Cryptography (FDTC)*, LNCS 4236, Springer-Verlag, 2006, pp. 36-52.

[18] J.-M. Bohli, S. Röhrich, and R. Steinwandt, Key substitution attacks revisited: Taking into account malicious signers, *Intern. J. Information Security*, **5** (2006), pp. 30-36.

[19] A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi, A closer look at PKI: Security and efficiency, *Public Key Cryptography — PKC 2007*, LNCS 4450, Springer-Verlag, 2007, pp. 458-475.

[20] R. Canetti, Universally composable signature, certification, and authentication, http://eprint.iacr.org/2003/239; a shorter version appeared in *Computer Security Foundations Workshop (CSFW-17 2004)*, IEEE Computer Society, 2004, pp. 219-235.

[21] D. Cash, Y. Z. Ding, Y. Dodis, W. Lee, R. J. Lipton, and S. Walfish, Intrusion-resilient key exchange in the bounded retrieval model, *Fourth Theory of Cryptography Conference — TCC 2007*, LNCS 4392, Springer-Verlag, 2007, pp. 479-498.

[22] S. Chatterjee, A. Menezes and P. Sarkar, Another look at tightness, *Selected Areas in Cryptography — SAC 2011*, to appear.

[23] J. Clayton, *Charles Dickens in Cyberspace: The Afterlife of the Nineteenth Century in Postmodern Culture*, Oxford Univ. Press, 2003.

[24] R. Cooke, *The Mathematics of Sonya Kovalevskaya*, Springer-Verlag, 1984.

[25] J. Coron, Resistance against differential power analysis for elliptic curve cryptosystems, *Cryptographic Hardware and Embedded Systems (CHES)*, LNCS 1717, Springer-Verlag, 1999, pp. 292-302.

[26] R. Cramer and V. Shoup, Signature schemes based on the strong RSA assumption, *ACM Transactions on Information and System Security*, **3**, No. 3 (August 2000), pp. 161-185.

[27] G. Di Crescenzo, R. J. Lipton, and S. Walfish, Perfectly secure password protocols in the bounded retrieval model, *Third Theory of Cryptography Conference — TCC 2006*, LNCS 3876, Springer-Verlag, 2006, pp. 225-244.

[28] J. P. Degabriele, K. G. Paterson, and G. J. Watson, Provable security in the real world, *IEEE Security & Privacy*, **9**, No. 3 (May/June 2011), pp. 18-26.

[29] R. L. Dennis, Security in the computer environment, SDC-SP 2440/00/01, AD 640648 (August 18, 1966).

[30] W. Diffie, P. van Oorschot, and M. Wiener, Authentication and authenticated key exchanges, *Designs, Codes and Cryptography*, **2** (1992), pp. 107-125.

[31] Y. Dodis, P. Lee, and D. Yum, Optimistic fair exchange in a multi-user setting, *J. Universal Computer Science*, **14** (2008), pp. 318-346.

[32] A. Domínguez-Oviedo, On fault-based attacks and countermeasures for elliptic curve cryptosystems, Ph.D. Dissertation, University of Waterloo, Canada, 2008.

[33] C. Donnelly and P. Embrechts, The devil is in the tails, *ASTIN Bulletin*, **40** (2010), pp. 1-33.

[34] O. Dunkelman, N. Keller, and A. Shamir, A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony, *Advances in Cryptology — Crypto 2010*, LNCS 6223, Springer-Verlag, pp. 393-410.

[35] S. Dziembowski, Intrusion-resilience via the bounded-storage model, *Third Theory of Cryptography Conference — TCC 2006*, LNCS 3876, Springer-Verlag, 2006, pp. 207-224.

[36] S. Dziembowski and K. Pietrzak, Leakage-resilient cryptography, *Proc. 49th Annual IEEE Symposium on the Foundations of Computer Science*, 2008, pp. 293-302.

[37] J. Fan, X. Guo, E. De Mulder, P. Schaumont, B. Preneel, and I. Verbauwhede, State-of-the-art of secure ECC implementations: A survey of known side-channel attacks and countermeasures, *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, IEEE, 2010, pp. 76-87.

[38] P. Fouque, R. Lercier, D. Réal, and F. Valette, Fault attack on elliptic curve Montgomery ladder implementation, *Fault Diagnosis and Tolerance in Cryptography (FDTC)*, IEEE, 2008, pp. 92-98.

[39] R. Gennaro, S. Halevi, and T. Rabin, Secure hash-and-sign signatures without the random oracle, *Advances in Cryptology — Eurocrypt '99*, LNCS 1592, Springer-Verlag, 1999, pp. 123-139.

[40] C. Gentry, Practical identity-based encryption without random oracles, *Advances in Cryptology — Eurocrypt 2006*, LNCS 4004, Springer-Verlag, 2006, pp. 445-464.

[41] S. Goldwasser, S. Micali, and R. Rivest, A "paradoxical" solution to the signature problem, *Proc. 25th Annual IEEE Symposium on the Foundations of Computer Science*, 1984, pp. 441-448.

[42] S. Jones, The formula that felled Wall St., *Financial Times*, 24 April 2009, http://www.ft.com/cms/s/2/912d85e8-2d75-11de-9eba-00144feabdc0.html.

[43] M. Joye, J. J. Quisquater, S. M. Yen, and M. Yung, Observability analysis — Detecting when improved cryptosystems fail, *Topics in Cryptology — CT-RSA 2002*, LNCS 2271, Springer-Verlag, 2002, pp. 17-29.

[44] M. Joye and S. M. Yen, Checking before output may not be enough against fault-based cryptanalysis, *IEEE Transactions on Computers*, **49** (2000), pp. 967-970.

[45] M. Joye and S. M. Yen, The Montgomery powering ladder, *Cryptographic Hardware and Embedded Systems (CHES)*, LNCS 2523, Springer-Verlag, 2002, pp. 291-302.

[46] B. Kaliski, Contribution to ANSI X9F1 and IEEE P1363 working groups, 17 June 1998.

[47] J. Katz, Signature schemes with bounded leakage resilience, http://eprint.iacr.org/2009/220.pdf.

[48] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, Chapman and Hall/ CRC, 2007.

[49] J. Katz and V. Vaikuntanathan, Signature schemes with bounded leakage resilience, *Advances in Cryptology — Asiacrypt 2009*, LNCS 5912, Springer-Verlag, pp. 703-720.

[50] T. Kleinjung *et al.*, Factorization of a 768-bit RSA modulus, *Advances in Cryptology — Crypto 2010*, LNCS 6223, Springer-Verlag, 2010, pp. 333-350.

[51] A. H. Koblitz, *A Convergence of Lives: Sofia Kovalevskaia — Scientist, Writer, Revolutionary*, Birkhaüser, 1983.

[52] A. H. Koblitz, N. Koblitz, and A. Menezes, Elliptic curve cryptography: The serpentine course of a paradigm shift, *J. Number Theory*, **131** (2011), pp. 781-814.

[53] N. Koblitz, The uneasy relationship between mathematics and cryptography, *Notices of the Amer. Math. Soc.*, **54** (2007), pp. 972-979.

[54] N. Koblitz and A. Menezes, Another look at "provable security," *J. Cryptology*, **20** (2007), pp. 3-37.

[55] N. Koblitz and A. Menezes, Another look at "provable security." II, *Progress in Cryptology — Indocrypt 2006*, LNCS 4329, Springer-Verlag, 2006, pp. 148-175.

[56] P. Kocher, Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems, *Advances in Cryptology — Crypto '96*, LNCS 1109, Springer-Verlag, 1996, pp. 104-113.

[57] P. Kocher, Differential power analysis, *Advances in Cryptology — Crypto '99*, LNCS 1666, Springer-Verlag, 1999, pp. 388-397; a brief version was presented at the Rump Session of Crypto '98.

[58] H. Krawczyk, HMQV: A high-performance secure Diffie-Hellman protocol, *Advances in Cryptology — Crypto 2005*, LNCS 3621, Springer-Verlag, 2005, pp. 546-566.

[59] M. G. Kuhn, Compromising emanations: Eavesdropping risks of computer displays, *Technical Report* 577, University of Cambridge Computer Laboratory, 2003, available at http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-577.pdf.

[60] J. le Carré, *The Looking Glass War*, New York: Coward-McCann, 1965.

[61] A. K. Lenstra, J. Hughes, M. Augier, J. Bos, T. Kleinjung, and C. Wachter, Ron was wrong, Whit is right, available at: http://eprint.iacr.org/2012/064.pdf.

[62] M. Luby and C. Rackoff, How to construct pseudorandom permutations from pseudorandom functions, *SIAM J. Computing*, **17** (2) (1988), pp. 373-386.

[63] L. Lydersen, C. Wiechers, C. Wittmann, D. Elser, J. Skaar, and V. Makarov, Hacking commercial quantum cryptography systems by tailored bright illumination, *Nature Photonics*, **4** (2010), pp. 686-689.

[64] J. Manger, A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as standardized in PKCS #1 v2.0, *Advances in Cryptology — Crypto 2001*, LNCS 2139, Springer-Verlag, 2001, pp. 230-238.

[65] J. Markoff, Researchers find a flaw in a widely used online encryption method, *The New York Times*, 15 February 2012, p. B4.

[66] K. McCurley, Language modeling and encryption on packet switched networks, *Advances in Cryptology — Eurocrypt 2006*, LNCS 4004, Springer-Verlag, 2006, pp. 359-372.

[67] A. Menezes, Another look at HMQV, *J. Mathematical Cryptology*, **1** (2007), pp. 47-64.

[68] A. Menezes and N. Smart, Security of signature schemes in a multi-user setting, *Designs, Codes and Cryptography*, **33** (2004), pp. 261-274.

[69] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[70] Z. Merali, Hackers blind quantum cryptographers, *Nature News*, http://www.nature.com/news/2010/100829/full/news.2010.436.html.

[71] S. Micali and L. Reyzin, Physically observable cryptography, *First Theory of Cryptography Conference — TCC 2004*, LNCS 2951, Springer-Verlag, 2004, pp. 278-296.

[72] P. Montgomery and R. Silverman, An FFT extension to the $P-1$ factoring algorithm, *Mathematics of Computation*, **54** (1990), pp. 839-854.

[73] M. Myers, C. Adams, D. Solo, and D. Kapa, Internet X.509 Certificate Request Message Format, IETF RFC 2511, 1999, http://www.rfc-editor.org/rfc/rfc2511.txt.

[74] National Institute of Standards and Technology, Digital Signature Standard, FIPS Publication 186, 1994.

[75] National Security Agency, Tempest: A signal problem, approved for release 27 September 2007, available at http://www.nsa.gov/public_info/_files/cryptologic_spectrum/tempest.pdf.

[76] P. Nguyen and I. Shparlinski, The insecurity of the digital signature algorithm with partially known nonces, *Designs, Codes and Cryptography*, **30** (2003), pp. 201-217.

[77] T. Okamoto, Provably secure and practical identification schemes and corresponding signature schemes, *Advances in Cryptology — Crypto '92*, LNCS 740, Springer-Verlag, 1993, pp. 31-53.

[78] K. Paterson, T. Ristenpart, and T. Shrimpton, Tag size *does* matter: Attacks and proofs for the TLS record protocol, *Advances in Cryptology — Asiacrypt 2011*, to appear.

[79] K. Paterson and G. Watson, Plaintext-dependent decryption: A formal security treatment of SSH-CTR, *Advances in Cryptology — Eurocrypt 2010*, LNCS 6110, Springer-Verlag, pp. 345-369.

[80] S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance, *IEEE Transactions on Information Theory*, **24** (1978), pp. 106-110.

[81] J. M. Pollard, Theorems on factorization and primality testing, *Proc. Cambridge Philos. Soc.*, **76** (1974), pp. 521-528.

[82] M. O. Rabin, Digitalized signatures and public-key functions as intractable as factorization, MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[83] R. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public key cryptosystems, *Communications of the ACM*, **21** (1978), pp. 120-126.

[84] P. Rogaway, Practice-oriented provable security and the social construction of cryptography, Unpublished essay based on an invited talk at Eurocrypt 2009, May 6, 2009, available at http://www.cs.ucdavis.edu/~rogaway/papers/cc.pdf.

[85] P. Rogaway, M. Bellare, and J. Black, OCB: A block-cipher mode of operation for efficient authenticated encryption, *ACM Transactions on Information and System Security*, **6** (2003), pp. 365-403.

[86] P. Rogaway and T. Shrimpton, A provable-security treatment of the key-wrap problem, *Advances in Cryptology — Eurocrypt 2006*, LNCS 4004, Springer-Verlag, 2006, pp. 373-390.

[87] RSA Laboratories, PKCS #1 v2.1: RSA Cryptography Standard, ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf.

[88] RSA Laboratories, PKCS #10 v1.7: Certification Request Syntax Standard, ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-10/pkcs-10v1_7.pdf.

[89] D. Russell and G. T. Gangemi Sr., *Computer Security Basics*, Chapter 10: TEMPEST, O'Reilly & Associates, 1991.

[90] F. Salmon, Recipe for disaster: the formula that killed Wall Street, *Wired Magazine*, 23 Feb. 2009.

[91] V. Shoup, Why chosen ciphertext security matters, IBM Research Report RZ 3076 (#93122), 23 November 1998.

[92] F.-X. Standaert, How leaky is an extractor?, *Progress in Cryptology — Latincrypt 2010*, LNCS 6212, Springer-Verlag, 2010, pp. 294-304.

[93] F.-X. Standaert, T. Malkin, and M. Yung, A unified framework for the analysis of side-channel key recovery attacks, *Advances in Cryptology — Eurocrypt 2009*, LNCS 5479, Springer-Verlag, 2009, pp. 443-461.

[94] N. Stephenson, *Cryptonomicon*, New York: Perennial, 1999.

[95] C.-H. Tan, Key substitution attacks on some provably secure signature schemes, *IEICE Trans. Fundamentals*, **E87-A** (2004), pp. 226-227.

[96] M. Turan et al., Status report on the second round of the SHA-3 cryptographic hash algorithm competition, NIST Interagency Report 7764, 2011.

[97] W. van Eck, Electromagnetic radiation from video display units: An eavesdropping risk?, *Computers and Society*, **4** (1985), pp. 269-286.

[98] S. Vaudenay, Provable security in block ciphers by decorrelation, *15th Annual Symposium on Theoretical Aspects of Computer Science — STACS '98*, LNCS 1373, Springer-Verlag, 1998, pp. 249-275.

[99] D. Wagner, The boomerang attack, *Fast Software Encryption — FSE '99*, LNCS 1636, Springer-Verlag, 1999, pp. 156-170.

[100] M. Whitehouse, Slices of risk, *The Wall Street Journal*, 12 Sept. 2005.

[101] P. Wright, *Spycatcher — The Candid Autobiography of a Senior Intelligence Officer*, William Heinemann, Australia, 1987.

[102] G. Yang, D. S. Wong, X. Deng, and H. Wang, Anonymous signature schemes, *Public Key Cryptography — PKC 2006*, LNCS 3958, Springer-Verlag, 2006, pp. 347-363.

[103] T. Ylonen and C. Lonvick, The Secure Shell (SSH) Authentication Protocol, IETF RFC 4252, 2006, http://www.rfc-editor.org/rfc/rfc4252.txt.

Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195 U.S.A.

E-mail address: koblitz@math.washington.edu

Department of Combinatorics & Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1 Canada

E-mail address: ajmeneze@uwaterloo.ca