ANOTHER LOOK AT AUTOMATED THEOREM–PROVING. II

NEAL KOBLITZ

Abstract. I continue the discussion initiated in [22] of whether or not computer-assisted proofs are a promising approach to preventing errors in reductionist security arguments. I examine some recent papers that describe automated security proofs for hashed ElGamal encryption, Boneh-Franklin identity-based encryption, and OAEP.

Keywords: Automated theorem–proving, computer-assisted proof, proof checking, public key cryptography, encryption.

AMS classification: 03B35, 68T15, 94A60, 11T71

1. Introduction

The notion of a carefully constructed logical proof of a theorem, which goes back to the ancient Greeks, has been a central paradigm in theoretical mathematics, and more recently in certain branches of applied mathematics as well. In general terms there is broad agreement about what a proof is — an airtight, convincing argument that a certain assertion must be true. However, there has sometimes been controversy about what constitutes a proof. For example, the proof of the Four Color Conjecture in 1976, which used a computer to run through a large number of cases, left many mathematicians deeply troubled. Another example can be found in algebraic geometry, where for many years proofs often cited results whose own proofs could not be found in the published literature but had only been presented in outline form in lectures; this also left some mathematicians uneasy. There is a certain social component in the notion of proof: what constitutes a satisfactory proof might vary from one research community to another. As Yu. I. Manin observed, "a proof only becomes a proof after the social act of 'accepting it as a proof'" ([25], p. 48).

The two most important characteristics of a satisfactory proof of a theorem are correctness and clarity. That is, it must (1) be free of gaps or errors, and (2) be understandable to anyone with the necessary technical prerequisites. The main tool for ensuring the first property is peer review — first by the journal's referees and later by the readers of the paper. The process is not perfect. In particular, how much peer review a proof gets depends on how important the result is, and errors are much more likely to go undetected in proofs of minor results than in proofs of major ones.

The second property is more subjective, but no less important than the first one — in part because how much careful peer review a proof is likely to get depends on how much time and effort is required to get through it. A well-written proof is highly valued in mathematics; in his well-regarded guidelines for mathematical writing [23], Steven Krantz devotes several pages to advice for organizing a proof so that it is as readable as possible. A proof of an important result that is particularly elegant and well-organized — where the central ideas and logical progression are crystal clear — is sometimes said to be a proof from "The Book," which was Paul Erdös' term for an imaginary repository of all of the most beautiful proofs of mathematics.

Several leaders of the cryptographic research community have acknowledged that proofs in cryptography often fall far short of the above objectives. According to Bellare and Rogaway [8],

> In our opinion, many proofs in cryptography have become essentially unverifiable. Our field may be approaching a crisis of rigor.

They say that this crisis "has at its core a significant social element," and Halevi [18] also says that

> Some of the reasons for this problem are social (e.g., we mostly publish in conferences rather than journals).

It is Halevi in [18] who is credited with articulating a response to this problem that is based on computer-aided construction and checking of proofs.

The purpose of the present article is to continue the discussion initiated in [22] of whether or not this program has contributed to resolving the crisis of rigor in cryptography. I examine some recent state-of-the-art papers on computer-assisted proofs in cryptography, starting with [2], which won the Best Paper Award at Crypto 2011.

I first examine the security proof for hashed ElGamal encryption in [2], and then discuss more briefly the treatment of Boneh-Franklin identity-based encryption in [4]. Next I comment on the automated theorem–proving approach to OAEP in [3] and consider whether computer-assisted proof techniques are likely to help prevent fallacies in proofs. In the conclusion I suggest a methodology for determining whether or not computer-aided reductionist security arguments are likely to help prevent errors in proofs.

2. Hashed ElGamal Encryption

Let $\mathcal{G}$ be a cyclic group (written multiplicatively) of order equal to a k-bit prime q, and let g be a generator. Suppose that plaintexts are bitstrings of length n, and let $H\colon \mathcal{G} \to \{0,1\}^n$ be a hash function. Hashed ElGamal encryption is defined as follows. In key generation Alice selects a random integer $x \bmod q$ as her secret key and computes $\alpha = g^x$ as her public key. Given the public key α and a message m, Bob encrypts by choosing a random integer $y \bmod q$ (which is chosen fresh for each m) and computing $\beta = g^y$,

$h = H(\alpha^y)$ and $\zeta = h \oplus m$; he sends (β, ζ) to Alice. She decrypts by finding $h = H(\beta^x)$ and then $m = \zeta \oplus h$.

The security of this scheme is assured provided that the Computational Diffie–Hellman problem (CDH) is intractable in the underlying family of groups $\mathcal{G}$. In other words, one assumes that for uniformly random elements x and y mod q it is hard to compute g^{xy} given g^x and g^y.

We now give the proof that this encryption scheme is semantically secure in the random oracle model if the CDH is intractable. This means that from the ciphertext it is impossible to determine any information about the plaintext, unless one can solve the CDH. A version of semantic security that is sometimes convenient for proofs is IND-CPA ("indistinguishability under chosen-plaintext attack"). This means that if the attacker chooses two plaintexts m_0 and m_1 and gets to see the encryption of one of the two (chosen at random), then she cannot determine which of the two was encrypted with more than $\frac{1}{2} + \epsilon$ probability of being right.

The "random oracle model" [6] means that the hash function H is modeled as a random function whose output is obtained not from an algorithm but from an "oracle" that chooses a random string of n bits every time it is queried for $H(\gamma)$ for a group element γ. The oracle keeps a record of all previous queries, and if $H(\gamma)$ is queried a second time, it must give the same value.

Theorem 1. *Hashed ElGamal encryption is semantically secure in the IND-CPA sense in the random oracle model if the CDH is intractable.*

2.1. **Mathematical proof.** The random oracle assumption on H means that nothing can be determined about its output unless the exact value of its input is known. The attacker knows g^x (which is Alice's public key) and g^y (which is part of the ciphertext), but by assumption she cannot compute g^{xy}. Since $h = H(g^{xy})$, this means that the attacker does not know the input to the hash function. Thus, to the attacker h is a random bitstring about which she has no information, and so the encryption $\zeta = h \oplus m$ is a one-time pad, which, as Claude Shannon proved in the 1940s, has perfect security. QED

Remark 1. In my view, this proof has an appealing elegance and simplicity that make it worthy of respect. There are, however, some mathematicians who would disagree and would disparage it as nothing but a "trivial tautology." This type of stigmatization of a nice result is unfair, and I agree with Jonathan Katz that we should protest vehemently, vigorously, and vociferously against the tendency of some mathematicians to look upon those of us who work in cryptography with elitist and snobbish condescension (see [21]).

2.2. **Crypto proof (early style).** In cryptography the preferred style of proof is by a *reductionist* argument. This means that we assume that there is an adversary Cynthia who has an algorithm to distinguish (with probability

more than $\frac{1}{2} + \epsilon$) whether (β, ζ) is the encryption of m_0 or m_1. We must show how Sam the Solver could use Cynthia's algorithm to solve the CDH.

But in order for the reduction to work, we need to replace the CDH by a closely related problem called the List Computational Diffie-Hellman problem (LCDH). That is the problem, given g^x and g^y, of producing a polynomial-size list L of elements of $\mathcal{G}$ that includes g^{xy}.[1]

Thus, given g^x and g^y, Sam wants to interact with Cynthia (or rather, with her algorithm) in such a way as to produce a list $L \ni g^{xy}$. Sam gives Cynthia the key $\alpha = g^x$ and waits for her to give him two test messages m_0 and m_1. He chooses $b \in \{0, 1\}$ at random, sets $\beta = g^y$, and chooses a random n-bit string h as $H(g^{xy})$. He then sets $\gamma = h \oplus m_b$ and gives Cynthia's algorithm the ciphertext (β, γ). Whenever Cynthia's algorithm queries a hash value $H(\delta)$, Sam responds with a random n-bit number, except that he keeps a record, called the H-List, of all queried pairs $(\delta, H(\delta))$, and if the same δ is queried a second time, Sam gives the same value $H(\delta)$. Sam's list L is simply the set of δ.

If Cynthia is to have a better than 50-50 chance of guessing which message m_b was encrypted, at some point she must query the hash of $\delta = g^{xy}$. Since in general Sam has no way to recognize that δ is the answer to the CDH and so needs to be assigned the same hash value h as before, he'll pick a random h', almost certainly not equal to h, as $H(\delta)$. Since this is a violation of the rules, there is no telling how Cynthia's algorithm might act starting from that moment. But it makes no difference, since Sam already has a list that includes g^{xy}. QED

2.3. **Crypto proof (late style).** Over the past decade or two the style of proof-writing in cryptography has changed. Partly in response to the "practice-oriented" paradigm introduced in [5], it is now considered desirable to give detailed estimates of both running times and probabilities. Thus, a reductionist proof might establish a result of the form, "If the protocol A succumbs to an attack of type B in time $\le t(k)$ with probability $\ge \epsilon(k)$, then the mathematical problem C can be solved in time $\le t'(k)$ with probability $\ge \epsilon'(k)$, where k is the security parameter." Both $t'(k)$ and $\epsilon'(k)$ are typically functions not only of k, t and ϵ, but also of bounds on the number of hash queries and decryption or signature queries, and other parameters of the protocol. Along with the growing complexity of provable security statements, there has been a concomitant introduction of new notation, such as Pr (for probability), Adv (for advantage), Succ (for success probability), etc.; each of these may carry subscripts and superscripts that indicate parameters, problem names, security definitions, algorithms, or "games."

[1] Note that an adversary who solves LCDH has broken IND-CPA, since she can compute a list of possible messages $H(\gamma) \oplus \zeta$ for $\gamma \in L$, one of which is the test-message m_b, $b \in \{0, 1\}$.

$$\left[\begin{array}{c}\textbf{Game } G_2\colon\\ x \leftarrow_{\$} \mathbb{Z}_q;\ \alpha \leftarrow g^x;\\ y \leftarrow_{\$} \mathbb{Z}_q;\ \hat{y} \leftarrow \alpha^y;\\ (m_0, m_1) \leftarrow \mathcal{A}_1(\alpha);\\ b \leftarrow_{\$} \{0,1\};\\ h \leftarrow_{\$} \{0,1\}^k;\\ b' \leftarrow \mathcal{A}_2(g^y, h \oplus m_b);\\ \text{return } (b = b')\end{array}\right] \longrightarrow \left[\begin{array}{c}\textbf{Game } G_3\colon\\ x \leftarrow_{\$} \mathbb{Z}_q;\ \alpha \leftarrow g^x;\\ y \leftarrow_{\$} \mathbb{Z}_q;\ \hat{y} \leftarrow \alpha^y;\\ (m_0, m_1) \leftarrow \mathcal{A}_1(\alpha);\\ \gamma \leftarrow_{\$} \{0,1\}^k;\\ b' \leftarrow \mathcal{A}_2(g^y, \gamma);\\ b \leftarrow_{\$} \{0,1\};\\ \text{return } (b = b')\end{array}\right]$$

$$\models G_2 \sim G_3\colon \text{true} \Longrightarrow =_{(\text{res},\hat{y},L_{\mathcal{A}})}$$

$$\Pr[G_2\colon b = b'] = \Pr[G_3\colon b = b'] = 1/2 \qquad \Pr[G_2\colon \hat{y} \in L_{\mathcal{A}}] = \Pr[G_3\colon \hat{y} \in L_{\mathcal{A}}]$$

FIGURE 1. Part of the proof sketch for hashed ElGamal encryption.

Lately many reductionist security proofs are written in a style called "game-hopping" (see [8, 9, 28, 33]). Here the reductionist argument is broken down into a series of small transitions from one sequence of interactions with the adversary to a slightly different sequence. At each stage the error or discrepancy between the two "games" is measured. This provides a running total that can be used to express t' and ϵ' in terms of t, ϵ, and various parameters.

In §2 of [2] the authors give a "proof sketch" of hashed ElGamal security that consists of five games: Game IND-CPA (the game corresponding to indistinguishability under chosen-plaintext attack), Games G_1, G_2, and G_3, and the Game LCDH (the game corresponding to solving the LCDH problem). For example, Fig. 1 shows the transition from G_2 to G_3 (in Fig. 1 $\mathbb{Z}_q$ denotes the integers mod q, $\leftarrow_{\$}$ means "chosen uniformly at random from," and $\mathcal{A}_i$ denotes actions by the adversary).

The "proof sketch" consists of four such transitions, going from Game IND-CPA to G_1 to G_2 to G_3 to Game LCDH, occupying a full page of small type in [2]. Despite its formidable appearance, the authors point out that, compared to an earlier version [1] the same authors prepared for a different programming language called CertiCrypt,

> The resulting proof sketch is about 250 lines long, about 5 times shorter than the proof in CertiCrypt... and arguably much simpler and close to a pen-and-paper proof.

Presumably it is because of its brevity compared to CertiCrypt that they chose the name "EasyCrypt" for the programming language used in [2].

2.4. **Automated proof.** Notice that the 250 lines of reduction steps are only the proof "sketch"; it is merely the *input* to EasyCrypt, which produces the actual proof.

Remark 2. The process of transforming a proof sketch into a format that EasyCrypt understands has to be done manually and could be error-prone. For instance, a step that is actually invalid might be inadvertently replaced by a step that's trivially valid, in which case EasyCrypt would presumably generate an apparently valid proof, and it's not clear that the fallacy would ever be detected.

After a human has already laboriously constructed the 250 lines of input for EasyCrypt from the proof sketch for hashed ElGamal, one might wonder what is left for the computer to do. It turns out that the computer does a lot — generating several thousand lines of code for the purpose of

> *Generating Verifiable Evidence.* EasyCrypt implements a compiler that turns proof sketches into Coq files that are compatible with the CertiCrypt framework and can be verified using the type checker of Coq. The compiler... significantly increases confidence in proof sketches by producing independently verifiable proofs, and providing means of checking the consistency of the set of axioms used in a proof sketch. (§3 of [2])

The expansion of the original proof from a paragraph or two to several thousand lines of computer output is remarkable. Can one find anything similar in the history of mathematical proofs? There is one striking analogy that comes to mind. Early in the 20th century, Russell and Whitehead wrote *Principia Mathematica*, a three-volume compendium of formal logical arguments that spelled out in minute detail the foundations of arithmetic and elementary mathematics. A true *tour de force* — and somewhat influential in its time, because of the philosophical debates going on between different schools of thought on the foundations of mathematics — it is now of only antiquarian interest, gathering dust deep in the recesses of university math libraries. Figure 2 shows the proof of a proposition late in the first volume that establishes that $1 + 1 = 2$.

Extremely long computer-aided reductionist security arguments would not be meaningless exercises if there were evidence that they are likely to prevent the types of errors in proofs that have caused great embarrassment to the cryptographic research community. But this is doubtful. I'll return to this question in §5.

Remark 3. In §4.2 of [1] the authors refer to the List CDH assumption as a "slightly different formulation that is equivalent in an asymptotic setting" to CDH. Let us put aside the question of what an "asymptotic setting" means, and look at their explanation of the equivalence:

$*54{\cdot}43.\quad \vdash :. \alpha, \beta \in 1. \supset : \alpha \cap \beta = \wedge. \equiv . \alpha \cup \beta \in 2$

Dem.

$$\begin{aligned}
&\vdash . *54{\cdot}26. \supset \vdash :. \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2. \equiv . x \neq y . \\
&[*51{\cdot}231] \qquad\qquad \equiv . \iota`x \cap \iota`y = \wedge. \\
&[*13{\cdot}12] \qquad\qquad \equiv . \alpha \cap \beta = \wedge \quad (1) \\
&\vdash . (1) . *11{\cdot}11{\cdot}35 . \supset \\
&\qquad \vdash :. (\exists x, y) . \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2 . \quad \equiv . \alpha \cap \beta = \wedge \quad (2) \\
&\vdash . (2) . *11{\cdot}54 . *52{\cdot}1 . \supset \vdash . \mathrm{Prop}
\end{aligned}$$

From this proposition it will follow, when arithmetical addition has been defined, that $1 + 1 = 2$.

FIGURE 2. A proposition from page 362 of *Principia Mathematica*.

> To see this, note that an adversary against List CDH with a non-negligible advantage can be converted into an adversary against CDH by returning a random element in the result list L; since L is necessarily of polynomial size, the... CDH advantage of the resulting adversary is still non-negligible.

In other words, if $\Pr_{LCDH}(k)$ is the adversary's probability of success in solving the LCDH (as a function of the security parameter k), and if $q_H(k)$ is the size of L (here q_H, which is a bound on the number of hash function queries, is polynomially bounded in k), then the adversary's probability of success in solving CDH is

$$\Pr_{CDH}(k) \geq \frac{1}{q_H(k)} \Pr_{LCDH}(k).$$

However, q_H can be quite large — it can be of the order of the adversary's running time. This means that the connection established in [2] between the security of hashed ElGamal and the CDH is extremely non-tight.

Some theoreticians would say that the math proof given in §2.1 is too intuitive and informal, and its only value is as heuristics. However, because the automated proof in [2] is non-tight, one can argue that its value is also merely heuristic. Because of security problems that have arisen with non-tight reductions, many researchers are coming around to the view that non-tight reductionist proofs should be regarded as only heuristic, unless the security parameters are increased to account for the non-tightness (which, because of the resulting loss of efficiency, is almost never done). See, for example, the conclusion of [12]. Thus, one can question whether from the

standpoint of rigor the automated proof has any advantage over the very short math proof.

In the case of hashed ElGamal, fortunately it is possible to obtain a much tighter reduction than the one in [2] that links its security to CDH. Namely, in §5 of [31] Shoup proves that LCDH and CDH are tightly equivalent. His method is as follows. Run your LCDH-solver twice, once with the CDH input (g, g^x, g^y) and once with shifted input $(g, (g^x)^a g^b, g^y)$ for randomly chosen a and b. Let $L = \{l_i\}$ be the list produced the first time and $L = \{l'_i\}$ be the list produced with the altered input. Compare the two lists, looking for l_i and l'_j such that $l_i^a (g^y)^b = l'_j$. When you find a match, you can be almost certain that the corresponding l_i is the answer to CDH.

Shoup's technique is clever, and in fact is the most nontrivial step in the security reduction from CDH to breaking hashed ElGamal. Because the automated techniques in [2] are apparently limited to generating proofs for trivial steps, they give only a much weaker result.

Remark 4. The paper [2] gives a second example of EasyCrypt computer-aided theorem–proving — the security of Cramer-Shoup encryption [13] against chosen-ciphertext attack. This part of the paper closely follows Halevi's proof of the same result in [18], and the comments I made in [22] apply equally to the EasyCrypt version. In particular, when discussing the security argument for the Cramer-Shoup cryptosystem one should not lose sight of the central point in the original proof — the fact that a certain bad outcome is equivalent to a codimension-1 condition in the parameter space. This is the step that is likely to require the most thought on the part of a careful reader in order to become convinced of its validity. In the automated proofs in [18] and [2] this step is accomplished by fiat; there is no attempt to construct an automated check of the validity of the most important part of the proof.

* * *

At Crypto 2011 the Program Chair, Phil Rogaway, explained that the paper [2] was chosen for the Best Paper Award "overwhelmingly" by the Program Committee, which "praised the work for its broad appeal...and its potential impact." And, indeed, the treatment of hashed ElGamal encryption in [2] is in some sense a remarkable achievement. Usually one has to leave the sciences entirely if one wishes to find works of scholarship — for example, Michel Foucault's turgid 760-page three-volume philosophical treatise on sex [14] — that have been so successful in turning something that should be interesting and accessible to everyone into something lengthy, unreadable, and boring.

3. Identity-Based Encryption

The title and abstract of [4] promise an automatically-verifiable security proof for the Boneh-Franklin identity-based encryption scheme in [11]. The

protocol that is treated is not, however, the full scheme but rather a simplified version called BasicIdent, which works as follows. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be cyclic groups of prime order q (written additively) equipped with a bilinear pairing $\hat{e}\colon \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$, and let $H_1\colon \{0,1\}^* \to \mathcal{G}_1$ and $H_2\colon \mathcal{G}_2 \to \{0,1\}^n$ be two hash functions (where n is the message length). In the initial setup, a trusted third party generates a master key pair. The master secret key is a randomly chosen integer a mod q, and the master public key is the pair (P, P_{pub}), where P is a randomly chosen element of $\mathcal{G}_1$ and $P_{pub} = aP$.

Let id be Alice's public identity, and set $Q_{id} = H_1(id) \in \mathcal{G}_1$. Using a secure channel, the trusted third party sends Alice her secret key $S = aQ_{id}$. To encrypt a message m for Alice, Bob computes Q_{id} and chooses a random integer c mod q. He then computes cP and $h = H_2(\hat{e}(Q_{id}, P_{pub})^c)$; his ciphertext is the pair $(cP, h \oplus m)$. To decrypt a ciphertext (Q, u) Alice simply computes $u \oplus H_2(\hat{e}(S, Q))$. To see why this works, let $Q_{id} = bP$ for some (unknown) b mod q. Then by the bilinearity of $\hat{e}$, both $\hat{e}(Q_{id}, P_{pub})^c$ and $\hat{e}(S, Q)$ are equal to $\hat{e}(P, P)^{abc}$.

The IND-CPA security of BasicIdent reduces to intractability of the following Bilinear Diffie-Hellman problem (BDH): given a quadruple of uniformly chosen elements $(P, aP, bP, cP) \in \mathcal{G}_1^4$, find $\hat{e}(P, P)^{abc}$. The BDH, which was stated for the first time in [11], fits in very well with BasicIdent, in the sense that the security argument (under the random oracle assumption for H_1 and H_2) is immediate. Namely, the information available to the adversary Cynthia is the uniformly chosen quadruple (P, P_{pub}, Q_{id}, cP). If Cynthia is unable to find $\hat{e}(P, P)^{abc}$, then she has no information about the hash function's output h, which is then nothing but a random bitstring. This means that to Cynthia the encryption $h \oplus m$ is a one-time pad, as in §2.1.

The entire treatment of Boneh-Franklin encryption in [4] amounts to verifying the above argument by the automated system CertiCrypt. The game-hopping proof consists of transitions involving nine games $G_{\text{IND-CPA}}$, $G_1, \ldots, G_7$, G_{BDH} (actually, twelve games if you count $G_{3'}$, $G_{6'}$ and $G_{7'}$). The hops from game to game occupy roughly three densely-packed pages of [4] that can be read with the help of a magnifying glass.

As explained in the introduction to [4], in order to prove that their identity-based encryption scheme is secure against chosen-ciphertext attack in the random oracle model, Boneh and Franklin constructed BasicIdent:

> The proof proceeds in two stages: first, an identity-based scheme BasicIdent is introduced and proved secure against chosen-plaintext attacks; second the BasicIdent scheme is transformed into a scheme that is secure against chosen-ciphertext attack by applying a variant of the Fujisaki-Okamoto transformation [15]. A flaw in the second part of the proof was discovered and fixed by Galindo [17]. Although, fortunately, in this case the fix did not require to modify

> the scheme or the underlying assumption, this shows that some degree of wariness is needed when evaluating provable security arguments.

From this passage and from the title and abstract the reader might get the impression that proof-checking software has been used for the whole security proof. However, this is not the case. The only part that is verified in [4] is the easy, tautological first stage; the second part, which is where the error and fix occurred, are outside the scope of the paper.

Remark 5. As in the case of hashed ElGamal, the automated proof linking BDH to the IND-CPA security of BasicIdent is very non-tight, because of the need to reduce BDH to List BDH. Once again Shoup's methods for CDH in [31] will also work for BDH and give a tight reduction. However, because Shoup's argument seems to be outside the reach of the automated techniques in [4], that paper, like [2], has to settle for a weaker result — a result which is arguably of only heuristic value.

4. OAEP

In this section and the next I discuss some questions about the computer-aided security proof in [3] for the Optimal Asymmetric Encryption Padding (OAEP) of Bellare and Rogaway [7].

The OAEP encryption scheme works as follows. Suppose that f is a trapdoor permutation on $\{0,1\}^k$ corresponding to a public-key/secret-key pair (pk, sk). For example, in RSA-OAEP the functions f and f^{-1} corresponding to $pk = (N, e)$ and $sk = d$, where $N \approx 2^k$, are given by $f(x) = x^e \bmod N$ and $f^{-1}(y) = x^d \bmod N$. Using f, Bob sends a plaintext m of n bits with k_1 zero-bits and k_0 random bits appended, where $k = n + k_1 + k_0$. Before applying f he puts the k-bit string through two Feistel rounds as follows. Let

$$G\colon \{0,1\}^{k_0} \longrightarrow \{0,1\}^{n+k_1}, \qquad H\colon \{0,1\}^{n+k_1} \longrightarrow \{0,1\}^{k_0}$$

be two hash functions. Bob applies the hash-function G to his random $r \in \{0,1\}^{k_0}$ and sets $s = m^0 \oplus G(r)$, where m^0 is m with k_1 zeros appended, and then $t = r \oplus H(s)$. He sets $x = s \parallel t$, and, using the public key, he computes the ciphertext $c = f(x)$. The construction is shown in Fig. 3. To decrypt c, Alice uses her secret key to compute $f^{-1}(c) = s \parallel t$, then finds $r = t \oplus H(s)$ and $m \parallel 0^{k_1} = s \oplus G(r)$. (If the k_1 zero-bits aren't there, she rejects the message as invalid ciphertext.)

The introduction to [3] summarizes the history of work on OAEP as follows:

> OAEP is widely deployed... Yet, the history of OAEP security is fraught with difficulties. The original 1994 paper of Bellare and Rogaway [7] proves that, under the hypothesis that the underlying trapdoor permutation family is one-way, OAEP is semantically secure under chosen-ciphertext

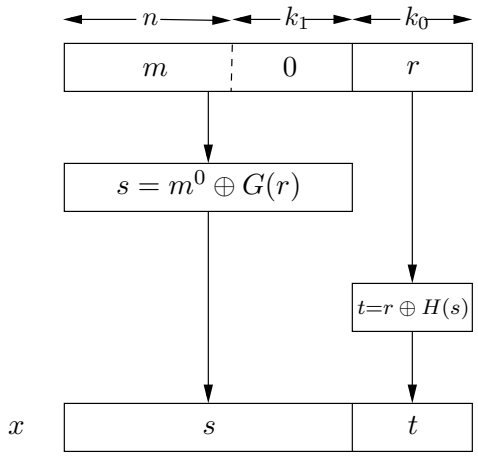

FIGURE 3. OAEP encryption.

> attacks. Shoup [32] subsequently discovered in 2000 that this proof only established the security of OAEP against non-adaptive chosen-ciphertext attacks, and not (as was believed at that time) against the stronger version of IND-CCA that allows the adversary to adaptively obtain the decryption of ciphertexts of its choice. Shoup suggested a modified scheme... Simultaneously, Fujisaki, Okamoto, Pointcheval and Stern [16] proved that OAEP in its original formulation is indeed secure against adaptive attacks, but under the assumption that the underlying permutation family is partial-domain one-way.... In 2004, Pointcheval [29] gave a different proof of the same result; this new proof fills several gaps in the reduction of [16], which results in a weaker bound than originally stated.

I won't go through a conceptual, mathematical proof (that could be stigmatized as too "informal" or "heuristic") or an early-style crypto proof of the IND-CCA security of OAEP under the partial-domain one-way assumption. Suffice it to say that such proofs would be quite short.[2] In dramatic contrast, the verifiable proof in [3] consists of "over 10,000 lines of Coq scripts... and contains about 30 games." Amusingly, ten pages earlier in the same article the authors write:

> As with any other mathematical activity, formal proofs strive for elegance and conciseness.

One can only wonder what they would consider to be an inelegant, unconcise proof!

[2]On p. 33 of [29] Pointcheval gives a half-page informal proof in a style similar to that of the proof given in §2.1 for hashed ElGamal.

Although the authors of [3] give the impression that they are carrying out Pointcheval's proof in [29], their techniques give a significantly weaker result: their non-tightness factor (roughly: the ratio of the time required to find partial preimages to the time required to successfully attack IND-CCA security of OAEP) is equal to the product of all three query bounds $q_D q_G q_H$, whereas Pointcheval's non-tightness factor does not include the bound q_D on the number of decryption queries. In [29] Pointcheval explains why his time estimate has only a $q_G q_H$ and not a $q_D q_G q_H$ term:

> Although the plaintext-extractor is called q_D times, there is no q_D multiplicative factor in the bound for t'. This comes from a simple bookkeeping argument. Instead of only storing the lists G-List and H-List, one stores an additional structure consisting of tuples $(\gamma, G(\gamma), \delta, H(\delta), y)$. A tuple is included only for $(\gamma, G(\gamma)) \in G$-List and $(\delta, H(\delta)) \in H$-List. For such a pair, one defines $\sigma = \delta$, $\theta = \gamma \oplus H(\delta)$, $\mu = G(\gamma) \oplus \delta$, and computes $y = f(\sigma \parallel \theta)$. If the last k_1 bits of μ are 0^{k_1}, one stores the tuple $(\gamma, G(\gamma), \delta, H(\delta), y)$. The cumulative cost of maintaining the additional structure is $q_G q_H (T_f + \mathcal{O}(1))$; [however, giving] it to the plaintext-extractor allows one to output the expected decryption of y, by table lookup, in constant time. Of course, a time-space tradeoff is possible, giving up the additional table, but raising the computing time to $q_D q_G q_H (T_f + \mathcal{O}(1))$.

(I've changed the notation in this passage to be consistent with the notation in this section and in [3].) Apparently a limitation of CertiCrypt in [3] was that it couldn't handle this argument giving the better bound.

Remark 6. It should be noted that, despite the comment at the end of the above quotation, actually no significant time-space tradeoff is involved in Pointcheval's "bookkeeping argument" for reducing the term in the time estimate from $q_D q_G q_H$ to $q_G q_H$. That is, the additional storage is not significantly greater than the storage required for the G-List and H-List. Namely, except with negligible probability, for a given δ there will be at most one γ such that $G(\gamma) \oplus \delta$ has the k_1 zero-bits that result in the tuple being stored. If there were a second such γ', then $G(\gamma)$ and $G(\gamma')$ would agree on those k_1 bits, of which the probability is only 2^{-k_1}. The probability that there are two different γ with G-values agreeing on those k_1 bits is equal to

$$1 - \prod_{i=0}^{q_G - 1} \frac{2^{k_1} - i}{2^{k_1}} \approx \frac{q_G^2}{2^{k_1+1}}$$

(this is just the birthday paradox). Thus, if one simply subtracts an additional $q_G^2/2^{k_1+1}$ in the expression for the probability ϵ', one need not worry about a time-space tradeoff.

5. Fallacies in Proofs

What is much more important than the weakness of the bound in [3] is the question of whether it meets the standard the authors set in §2 of [3]:

> Yet, what matters most about a formal proof is that it provides a nearly absolute degree of assurance, without requiring expensive human verification.

I'm skeptical that it's possible to find and eliminate the types of errors we tend to make in proofs without "expensive human verification." Except for obvious, tautological arguments — such as when there's no substantive difference between the two problems for which a reduction is being given — any nontrivial proof is going to involve some conceptual steps, and verification that there's no subtle error in such steps cannot be done by machine, at least not with any software I'm aware of.

Remark 7. It was because of statements such as the one quoted above that I wrote in [22] that "the hope of authors of recent papers on automated theorem–proving is to take the uncertainties, flaws, and human foibles out of the process of deriving reductionist security arguments." In [19] Halevi quoted this statement and commented: "I seriously doubt that there is anyone working on automated proofs that has this view... Koblitz completely missed the point of the work that is done in automated/computer-aided cryptographic proofs." Apparently the authors of [3] are missing the point, too, since they say pretty much the same thing that I did.

The authors of [3] have the benefit of hindsight. They know that the best security proof for the original OAEP is Pointcheval's in [29], so that is the one that they follow as they construct their elaborate proof sketch for CertiCrypt. But what if the game-hopping proof outline had been built up for either the proof in [7] (by someone who was unaware that Shoup [32] had found a subtle but fundamental error in it) or for the proof in [16] (by someone who was not aware that the proof, while generally sound, had some errors that affected the bounds)? Perhaps in the process of a painstaking construction of a very detailed proof outline for CertiCrypt or EasyCrypt, they would have found the errors. But if they hadn't, would the software have found the gaps in logic, or would it have obediently filled in the details of the proof as if it had been valid? I don't know the answer to this hypothetical question, but it seems doubtful that the software is capable of finding the types of errors that have occurred in the published literature on OAEP.

Similarly, one has to be skeptical about whether automated techniques would have caught some of the other high-profile fallacies in important security proofs, for example:

- Krawczyk's proof of security for HMQV [24] inadvertently assumed that an element of the supergroup $\mathcal{G}' \supset \mathcal{G}$ was necessarily in the subgroup $\mathcal{G}$, although in order to achieve greater efficiency a key

validation step had been removed from the protocol that would have ensured this (see [26, 27]).

- The security proof for the sequential aggregate signature scheme in [10] erroneously overlooked the fact that an attacker could work in the publicly known exponent space, and their "provably secure" protocol was completely broken a year later (see [20]).

It would be nice to have a way of testing whether automated software is really capable of detecting these sorts of errors. I'll propose a possible approach in the concluding section.

6. Conclusion

Much of the response to my critique [22] has been negative (no surprise). For example, the authors of [3] accuse me of having "fragmentary knowledge" and "profound misconceptions." I plead guilty to the first charge. I wrote [22] and the present paper from the viewpoint of an outsider, and I have no expertise in automated theorem–proving. In fact, not once in my professional life have I even attempted to construct a "game-hopping" proof of anything.

As far as "profound misconceptions" are concerned, it is hard for me to answer that charge, because the authors of [3] do not say what they believe my misconceptions to have been. Was it the notion that the true test of whether automated proofs are worthwhile is whether or not they are likely to prevent fallacies of the sort that have embarrassed the research community in the past? Was it my claim that most of the published computer-aided proofs for cryptographic protocols have been for reductionist arguments that are essentially trivial, and not for those that give the deeper results? Was it my assertion that on the rare occasion when automated proofs are given of nontrivial reductions (e.g., for Cramer-Shoup encryption) it is the trivial steps that are automated, while the most important steps — of the sort that in many security proofs have had errors — are inserted by fiat and not verified by the automated software?

In contrast to the authors of [3], in [19] Halevi wrote a reasoned and thoughtful response to [22]. First, he faulted me for quoting his comment about the social causes of the crisis of rigor in cryptography (also quoted in §1 above) without quoting the rest of his explanation, which is as follows:

> ...but the true cause of it [the problem of erroneous proofs] is that our proofs are truly complex. After all, the objects that we deal with are non-trivial algorithms, and what we try to prove are some properties of an intricate interaction between a few of these algorithms.

But then, after distinguishing between the creative, nontrivial parts of proofs and the "mundane" and "obvious" parts, he goes on to say that all that he claims about automated theorem–proving is that "computer programs can help with the task of checking the 'obvious details' of a proof, making sure

that there are no errors there." According to him, "Errors in published proofs are often contained in exactly those 'obvious details' (see the OAEP case for example), and computer software may help avoiding such errors." This contradicts what he said before about the "non-trivial" and "intricate" nature of proofs — not the trivial tautological equivalences — being the source of error. And indeed, in the OAEP case the error of Bellare and Rogaway was a subtle one that went undiscovered for seven years.

In some cases a protocol is tailor-made to fit a given problem $\mathcal{P}$ so that the reduction of $\mathcal{P}$ to the problem of breaking the protocol (in a specified sense) can be proved (perhaps under the random oracle assumption) in a short paragraph. We saw an example of this in §2. In other cases the problem $\mathcal{P}$ is tailor-made to fit the protocol, and again the reduction argument is immediate; we saw such a case in §3. In both situations computer-constructed proofs have worked fine, confirming the obvious in the same sense that *Principia Mathematica* confirmed some well-known identities (see Fig. 2).

The danger of error arises, as Halevi said, when "we deal with non-trivial algorithms, and...an intricate interaction between a few of these algorithms." The unresolved question, then, is whether automated theorem–proving software is likely to help in such circumstances.

In looking for a way to evaluate to what extent computer-aided proofs can help prevent errors, it is useful to recall what is often done with other security issues in cryptography. In many cases in making an evaluation the methodology that works best is interactive. To evaluate how good your firewall is, the best approach is to hire a team of hackers to try to break through it. The best way to stimulate work on the Integer Factorization and Elliptic Curve Discrete Logarithm problems — which lie at the heart of RSA and ECC, respectively — is to give a series of RSA challenges and ECC challenges, preferably with monetary awards attached. Before describing an analogous approach for computer-assisted proofs I'd like to digress for two paragraphs.

A pedagogical digression. Some mathematicians have argued that the removal of rigorous proofs from the teaching of most elementary math courses — such as first-year calculus — in American universities was a mistake, and we should return to an emphasis on proofs. According to them, even prospective engineering students should learn $\epsilon\delta$-proofs of the basic properties of limits. My response when I hear that viewpoint expressed is to conjecture that in practice all that would happen would be that the students would merely attempt to memorize the required proofs with no true understanding. Rather than seeing proofs the way mathematicians see them — as a way to develop critical thinking, logical discipline, and creativity — the students will think of them as a hated and meaningless ritual.

I have the following challenge to my colleagues who wish to teach a proof-based calculus course to engineering students. Half-way through the semester come to class and with a straight face give a plausible-sounding proof that makes no logical sense whatsoever. See if any of the students notice.

If not, then it seems to me that all your efforts to teach proofs have been a waste of time.

* * *

By analogy, the method I would propose to assess the value of computer-aided proofs in cryptography is to set up some challenges (preferably supported by monetary incentives). For a given protocol, challenge your colleagues to write a game-hopping proof that can be input to CertiCrypt or EasyCrypt in which they give a security reduction that they know has a fallacy. If the computer fills in the proof without objecting, they win and the computer loses; but if the software discovers their fallacy, then they get nothing, and you have some evidence that there's hope that this type of software can detect the kinds of errors in proofs that working cryptographers are likely to make.

7. Acknowledgments

I would like to thank Ann Hibner Koblitz and Alfred Menezes for reading and commenting on earlier drafts of this paper.

References

[1] G. Barthe, B. Grégoire, S. Heraud, and S. Zanella Béguelin, *Formal certification of ElGamal encryption: A gentle introduction to CertiCrypt*, 5th Intern. Workshop on Formal Aspects in Security and Trust – FAST 2008, LNCS 5491, Springer-Verlag, 2009, pp. 1-19.

[2] G. Barthe, B. Grégoire, S. Heraud, and S. Zanella Béguelin, *Computer-aided security proofs for the working cryptographer*, Advances in Cryptology – Crypto 2011, LNCS 6841, Springer-Verlag, 2011, pp. 71-90..

[3] G. Barthe, B. Grégoire, Y. Lakhnech, and S. Zanella Béguelin, *Beyond provable security verifiable IND-CCA security of OAEP*, CT-RSA 2011, LNCS 6558, Springer-Verlag, 2011, pp. 180-196.

[4] G. Barthe, F. Olmedo, and S. Zanella Béguelin, *Verifiable security of Boneh-Franklin identity-based encryption*, 5th Intern. Conference on Provable Security – ProvSec 2011, to appear.

[5] M. Bellare, *Practice-oriented provable security*, 1st Intern. Workshop on Information Security – ISW '97, LNCS 1396, Springer-Verlag, 1998, pp. 221-231.

[6] M. Bellare and P. Rogaway, *Random oracles are practical: a paradigm for designing efficient protocols*, 1st Annual Conference on Computer and Communications Security, ACM, 1993, pp. 62-73.

[7] M. Bellare and P. Rogaway, *Optimal asymmetric encryption*, Advances in Cryptology – Eurocrypt 1994, LNCS 950, Springer-Verlag, 1995, pp. 92-111.

[8] M. Bellare and P. Rogaway, *Code-based game-playing proofs and the security of triple encryption*, available at http://eprint.iacr.org/2004/331

[9] B. Blanchet and D. Pointcheval, *Automated security proofs with sequences of games*, Advances in Cryptology – Crypto 2006, LNCS 4117, Springer-Verlag, 2006, pp. 537-554.

[10] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum, *Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing*, 14th

ACM Conference on Computer and Communications Security – CCS 2007, ACM Press, 2007, pp. 276-285.
[11] D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing*, SIAM J. Computing 32(3) (2003), pp. 586-615.
[12] S. Chatterjee, A. Menezes, and P. Sarkar, *Another look at tightness*, to appear; available from http://anotherlook.ca and from http://eprint.iacr.org/2011/442.
[13] R. Cramer and V. Shoup, *A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack*, Advances in Cryptology – Crypto '98, LNCS 1462, Springer-Verlag, 1998, pp. 13-25.
[14] M. Foucault, *The History of Sexuality* (3 vols.), Pantheon Books, 1978.
[15] E. Fujisaki and T. Okamoto, *How to enhance the security of public-key encryption at minimum cost*, 2nd Intern. Workshop on Practice and Theory in Public Key Cryptography – PKC 1999, LNCS 1560, Springer-Verlag, 1999, pp. 53-68.
[16] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern, *RSA-OAEP is secure under the RSA assumption*, J. Cryptology 17 (2004), pp. 81-104.
[17] D. Galindo, *Boneh-Franklin identity based encryption revisited*, 32nd Intern. Colloquium on Automata, Languages and Programming – ICALP 2005, LNCS 3580, Springer-Verlag, pp. 791-802.
[18] S. Halevi, *A plausible approach to computer-aided cryptographic proofs*, available at http://eprint.iacr.org/2005/181.
[19] S. Halevi, posting to the IACR discussion forum for the eprint version of [22], 22 October 2007.
[20] J. Y. Hwang, D. H. Lee, and M. Yung, *Universal forgery of the Identity-Based Sequential Aggregate Signature Scheme*, ACM Symposium on Information, Computer & Communication Security – ASIACCS 2009, ACM Press, 2009, pp. 157-160.
[21] J. Katz, *Letter to the Editor*, Notices of the Amer. Math. Soc., December 2007, available at http://www.ams.org/notices/200711.
[22] N. Koblitz, *Another look at automated theorem–proving*, J. Math. Cryptology 1 (2007), pp. 385-403.
[23] S. Krantz, *A Primer of Mathematical Writing*, Amer. Math. Soc., 1996.
[24] H. Krawczyk, *HMQV: A high-performance secure Diffie-Hellman protocol*, Advances in Cryptology – Crypto 2005, LNCS 3621, 2005, pp. 546-566.
[25] Yu. I. Manin, *A Course in Mathematical Logic*, translated by N. Koblitz, Springer-Verlag, 1977.
[26] A. Menezes, *Another look at HMQV*, J. Mathematical Cryptology 1 (2007), pp. 47-64.
[27] A. Menezes and B. Ustaoglu, *On the importance of public-key validation in the MQV and HMQV key agreement protocols*, Progress in Cryptology – Indocrypt 2006, LNCS 4329, 2006, pp. 133-147.
[28] D. Nowak, *A framework for game-based security proofs*, available at http://eprint.iacr.org/2007/199.
[29] D. Pointcheval, *Provable security for public key schemes*, in *Contemporary Cryptology*, Advanced Courses in Mathematics — CRM Barcelona, Birkhäuser, 2005, pp. 133-189.
[30] B. Russell and A. N. Whitehead, *Principia Mathematica*, Vol. 1, 2nd ed., Cambridge University Press, 1950.
[31] V. Shoup, *Lower bounds for discrete logarithms and related problems*, Advances in Cryptology – Eurocrypt '97, LNCS 1233, Springer-Verlag, 1997, pp. 256-266.
[32] V. Shoup, *OAEP reconsidered*, Advances in Cryptology – Crypto 2001, LNCS 2139, Springer-Verlag, 2001, pp. 239-259.
[33] V. Shoup, *Sequences of games: a tool for taming complexity in security proofs*, available at http://eprint.iacr.org/2004/332.

Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195 U.S.A.
E-mail address: koblitz@uw.edu