Fundamental problems in provable security and cryptography

By Alexander W. Dent

Information Security Group, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK

This paper examines methods for formally proving the security of cryptographic schemes. We show that, despite many years of active research, there are fundamental problems which have yet to be solved. We also present a new approach to one of the more controversial aspects of provable security: the random oracle model.

Keywords: cryptography; provable security.

1. Introduction

We live in an age and society that surrounds us with information, and increasingly our day-to-day lives depend upon this information and our ability to manipulate it. For example, it is often taken for granted that we can control our bank accounts from almost anywhere in the world using a combination of satellite and cellular phone networks to talk to bank representatives, specialised wired ATM networks to withdraw money, and the Internet for online banking services. Sadly, whenever there are services for manipulating information that has value, there will be unscrupulous elements in society that will seek to subvert these services for their own benefit. This has led to the development of research into information security.

Information security is the field of research that aims to protect information from malicious attackers while still allowing legitimate users to manipulate data freely. Cryptography is the branch of information security which covers the study of algorithms and protocols that secure data. It is a diverse field, taking in elements of electronic engineering, computer science and mathematics, including computational and algebraic number theory, combinatorics, group theory, and complexity theory.

However, despite the subject's mathematical basis, cryptographers are only just beginning to develop the mathematical rigour that they need in order to be able to produce algorithms and protocols in which one can have true confidence. For many years the vast majority of cryptographic algorithms and protocols (collectively known as *cryptosystems*) proposed for practical use offered very little in the way of security guarantees. They were developed in an *ad-hoc* fashion, following a cycle in which cryptographic schemes were attacked, broken, repaired and attacked again. Some of these schemes have proven successful beyond the wildest dreams of their designers; most have fallen, irrevocably broken, by the wayside.

Increasingly, standardisation bodies and implementers are becoming dissatisfied with this *ad-hoc* approach and are demanding mathematically proven guarantees about the security of cryptosystems. The first significant results in this direction were obtained by Shannon (1949) who characterised the requirements for an encryption scheme to be 'perfect'. However, Shannon's theory proved difficult to extend

and little research was conducted in the area until the invention of asymmetric cryptography in 1976 (Diffie & Hellman 1976). This new direction in cryptography demonstrated that systems could be secure without ever being able to achieve perfect secrecy, and forced the community to revisit the formal notions of security. The first cryptosystem with a modern 'security proof' followed shortly (Rabin 1979). This 'proof' formally related the difficulty of breaking a particular scheme (in some security model) to the difficulty of factoring a number which is the product of two large primes. We believe this factoring problem to be hard to solve in any meaningful length of time.

This year has marked the 30th anniversary of the publication of the groundbreaking paper of Diffie & Hellman. Unfortunately, despite some significant successes, research has not advanced as far as one might like. There is heated debate about how we should formally model security; the relationship between provable security and complexity theory is still not well understood; and the theory underpinning the research area is inconsistently applied and full of unrealistic simplifying assumptions. These limitations are accepted simply because it is unclear how to proceed without them.

In this paper we will study these theoretical limitations and some of their practical implications. We will survey a wide-range of problems in the field of provable security, focussing most closely on the problems associated with the random oracle model, a powerful simplifying assumption which allows the analysis of a cryptosystem by modelling certain parts of its internal structure as random functions that act in a manner that is unknown to an attacker. We shall also present a novel application of the random oracle model that may shed some light on its future use within cryptography.

2. Fundamental problems in provable security

(*a*) *Symmetric cryptography and Shannon's theory of security*

Loosely speaking, cryptography may be separated into two overlapping branches: *symmetric* and *asymmetric*. In a symmetric cryptosystem, a group of privileged users all know a secret piece of data called a *key*, which we assume is not known by any user who may wish to attack the cryptosystem. This key is usually a short piece of data that is used by a complex algorithm to achieve some security functionality. We always assume that the attacker knows the complete description of this algorithm; this corresponds to the real-world assumption that the algorithm will be a piece of software or hardware that the attacker may be able to reverse-engineer or of which the attacker may be able to purchase a complete description. The classic example of a symmetric cryptosystem is a *block cipher*.

A block cipher cryptosystem consists of two algorithms: an encryption algorithm and a decryption algorithm. The encryption algorithm takes as input an n-bit message m and a k-bit key K and outputs an n-bit ciphertext C; the decryption algorithm takes as input an n-bit ciphertext C and a k-bit key K and outputs an n-bit message m. For any fixed key, we require that the decryption algorithm acts as the inverse of the encryption algorithm. Furthermore, we would hope that without knowledge of the secret key, an attacker can gain no useful information about a message from its ciphertext. Block ciphers are used as building blocks for several

other, more useful types of cryptosystems, including encryption schemes that can take messages of arbitrary length as input. These schemes are sometimes referred to as *modes of operation of a block cipher*.

Shannon (1949) proposed a theory to assess the secrecy of symmetric cryptosystems. This theory was based on his earlier theory of information and entropy (Shannon 1948) and involved examining the amount of information about a random message (drawn from some probability distribution) an attacker gains after being given an encryption of that message. An encryption scheme is said to be *perfect* if an attacker gains no information about the message from its encryption. However, it has been shown that for perfect secrecy to be achieved, every bit of information in the message must be encrypted using a bit of information in the key.

As an example, consider a perfect block cipher. Assuming that every n-bit message is equally likely to occur, Shannon's theory tells us that we will require n-bit keys. Of course, this result may be more easily seen by noting that, when given a ciphertext C encrypted using a k-bit key generated uniformly at random, there exist 2^k possible pre-images for C. Thus if $k < n$ then the attacker will be able to narrow down the number of possibilities for the message m and so gain information about the message. Furthermore, these keys are not reusable. If we wish to use the block cipher twice, then we will be encrypting a total of $2n$-bits of message and so require a key of length at least $2n$-bits. It is impossible to produce a system that is perfectly secure for arbitrary length messages.

Shannon's requirement that the key be as long the message is shockingly impractical for general use. In general, we are unable to generate, store or securely transport the large amount of keying material required for perfect secrecy; and Shannon's theory is very difficult to extend to cover the case where the key is shorter than the message. In such a case, we know that the encryption scheme cannot be perfect, but we have no effective method for estimating whether the scheme is secure enough to be used in practice. It took nearly thirty years for a new direction in cryptography to emerge that could consider the problem of provable security in a more flexible way.

(*b*) *Asymmetric cryptography and the reductionist theory of security*

In 1976, Diffie & Hellman published the first paper on asymmetric cryptography. In an asymmetric cryptosystem, rather than there being a single secret key, there exist two related keys: a *public key*, which is widely known, and a *private key*, which is only known by a single user. We assume that any attacker who wishes to break the cryptosystem is fully aware of the public key and any algorithms that may be used as part of the cryptosystem; the only piece of information that is denied to the attacker is the private key. Typically, asymmetric cryptosystems are based on the computation of large numerical values and are a lot slower than their symmetric counterparts (which tend to rely on faster, bit-oriented operations).

One classic example of an asymmetric cryptosystem is a *digital signature scheme*. A digital signature is a block of data appended to the end of a message that attests to the origin of the message and to the fact that the message has not been changed. A user produces a digital signature by executing a signing algorithm, which takes as input the message to be signed and the user's private key, and outputs a signature for that message. The veracity of the signature can be checked by executing a

verification algorithm with the message, the signature and the user's public key as input. We require that it is infeasible to produce a signature for a message without knowledge of the private key. Thus, any message with a valid signature attached to it must have been produced by the owner of the private key and cannot have been changed after it was produced.

The nature of the relationship between the public and private keys means that it is impossible for any asymmetric scheme to achieve a perfect notion of security. The public key, by definition, must contain enough information to compute its associated private key. Security is obtained by using large enough public and private key values so that, while it may be *theoretically* possible to recover the private key from the public key, it is not computationally feasible to do so. This notion of computational infeasibility led researchers to consider phrasing security requirements in terms of Turing's complexity theory (Turing 1936) rather than Shannon's information theoretic approach.

A common approach used in modern security proofs is to parameterise a cryptosystem's security in terms of a security parameter. This security parameter typically dictates the length of certain elements of the public and private keys. We determine which attacks are computationally feasible by defining a computationally feasible attacker as any probabilistic Turing machines whose running time is bounded by a polynomial in the security parameter. Note that if we follow this approach then we must also insist that any algorithm that forms part of the cryptosystem must also be computationally feasible, i.e. run in polynomial time.

However, this complexity-based approach has problems of its own. Let us consider the aforementioned digital signature scheme. Clearly, we wish the problem of producing a valid digital signature without knowledge of the private key to be computationally infeasible, even though the problem of checking whether a given signature is valid for a given message is easily solvable. This makes the problem of breaking the digital signature scheme an NP-type problem, and the security of the scheme depends upon the problem not being a P-type problem. Unfortunately, it is not known whether NP=P or not. Hence, without making a huge step forward in complexity theory, we cannot even derive definite mathematical statements about the security of the scheme: the best we can do is to prove that a reduction exists between the difficulty of breaking a cryptosystem and the difficulty of solving some well-studied mathematical problem (such as factoring large numbers or computing discrete logarithms in finite fields). Almost all modern security proofs take this reductionist approach and so rely on an assumption about the hardness of some mathematical problem in order to 'prove' their security.

The reductionist approach works well when proving the security of asymmetric cryptosystems and certain types of symmetric cryptosystems. The problem with using the reductionist approach with arbitrary symmetric cryptosystems is that there exist no natural candidate problems to which one may reduce the security of the scheme. Typically, if one can prove the security of a symmetric scheme, one does so by reducing the security of that scheme to the problem of distinguishing between an oracle which computes the permutation defined by a block cipher encryption function under a random key, and an oracle which computes a completely random permutation. This is a useful technique when proving the security of a mode of operation of a block cipher, but does not give us any way to prove the security of the block cipher itself. However, despite these drawbacks, this reductionist approach

now almost universally accepted within the cryptographic community as the most effective method to prove the security of a scheme.

(*c*) *Formal security models*

We may now begin to investigate the more controversial aspects of modern provable security. The first major area of controversy involves the descriptions of the formal security models used to assess security. A formal security model consists of two definitions: it must specify how an arbitrary, probabilistic, polynomial-time attacker can interact with legitimate users of a cryptosystem, and it must state what that attacker should achieve in order to 'break' the cryptosystem. There are two general approaches to formal security models.

The first is the game-based approach. In this style of security model the attacker interacts with a hypothetical probabilistic algorithm called a *challenger*. The challenger generates all the keys used in the system, and may respond to queries made by the attacker. The game terminates when the attacker terminates, and we assess whether the attacker has met the condition for breaking the cryptosystem. If a cryptosystem is to be proven secure, then we must show that the probability that an arbitrary attacker breaks the cryptosystem is small. Widely accepted game-based security models have been proposed for many types of cryptosystem, including digital signatures (Goldwasser *et al.* 1988), asymmetric encryption (Rackoff & Simon 1991) and symmetric encryption (Bellare *et al.* 1997).

As an example, we will consider the security model for a digital signature scheme. Consider an arbitrary, probabilistic, polynomial-time attacker. The challenger generates an asymmetric key pair of the appropriate security level (as determined by the security parameter). The attacker algorithm is then executed. It takes the public key and the security parameter as input. During its execution, the attacker may ask the challenger to produce signatures for messages of the attacker's choice. This the challenger does faithfully using the signing algorithm and the private key. The attacker terminates by outputting a signature σ and a message m. The attacker is deemed to have broken the system if the verification algorithm declares that σ is a valid signature for the message m and the attacker did not ask the challenger to sign the message m. This is a strong notion of security, but does capture many of the real-world capabilities of an attacker, particularly that they may be able to 'trick' a user or system into signing certain messages of their choice.

Game-based security models have the advantage of being simple to understand and easy to work with. However, a security proof in a game-based security model makes no claims about how secure a scheme is when it is placed in the context of a larger system. Most cryptographic schemes are not used as 'stand-alone' protocols, but are subroutines in larger computer systems, and one has to be careful to ensure that the security guarantees presented by the proof hold in the larger environment in which the cryptographic algorithm is used. It is also often more difficult to phrase the security requirements of complex protocols as game-based security models.

The other way to approach security modelling is to use simulation. In this scenario we envisage a system in which an arbitrary, probabilistic, polynomial-time attacker can interact with each algorithm of the cryptosystem, and also with an arbitrary, probabilistic, polynomial-time *environment*. The environment represents all other parties that may have access to the algorithms of the cryptosystem. We

also produce an idealised version of the cryptosystem that can never be broken. This is not a practical system: it will generally involve using an abstract third party who can always be trusted to transport and/or vouch for data and whose operation is outside of the view of both the environment and the attacker. To determine whether a scheme is secure, we examine the outputs of the attacker and the environment when they interact with the real cryptosystem, and when they interact with the idealised cryptosystem. Since the idealised system can never be broken, if the outputs of the environment and attacker are (roughly) the same when the idealised cryptosystem is used in place of the real cryptosystem, then the real cryptosystem must be secure. Hence, we declare the cryptosystem secure if the probability of being able to tell the difference between these two outputs is small.

It should be clear that simulation-based security models are stronger than game-based security models. In particular, simulation-based security models provide proofs that take into account the larger environment in which the cryptosystem will be used and so provide more reassuring security guarantees. Several simulation-based security models are currently in use (Pfitzmann & Waidner 2000; Canetti 2001). However, it has been shown that certain cryptographic functions can *never* be proven secure in simulation-based security models (Canetti & Fischlin 2001). Hence, neither approach to security modelling gives the full range of applicability and security guarantees that is desirable and there is fierce debate about which style of model should be regarded as correct.

(*d*) *Small inconsistencies: concrete vs. asymptotic security*

Another issue that has caused some controversy among cryptographers is the definition of 'small' in the statement 'the probability that an attacker can break the system should be small'. The original definition is that the attacker's probability should be negligible as a function of the security parameter.

Definition 2.1. *A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for every polynomial p there exists a positive integer $N(p)$ such that $|f(n)| \leq 1/|p(n)|$ for all $n \geq N(p)$.*

The problem with this definition is that it does not tell us anything about the security of a scheme for practical security parameters. It may be true that the probability of breaking a cryptosystem is asymptotically small, but that does not mean that the scheme is secure for security parameters that can actually be used. The alternative to the asymptotic definition is a concrete definition (Bellare 1997).

In a concrete security analysis, we still reduce the security of a cryptosystem to a well-studied mathematical problem; however, now we evaluate the security of the scheme based on the quality of the reduction. Typically, we prove the security of a cryptosystem by considering an arbitrary attacker that breaks the scheme and showing that we can use such an attacker to create an algorithm that will solve the underlying problem. A concrete security proof assumes that the attacker runs in time bounded by a known function $t(\lambda)$ and has an (unknown) success probability $\epsilon(\lambda)$, where λ is the security parameter. The reduction allows us to derive an algorithm for solving the underlying problem in time bounded by $t'(t(\lambda), \epsilon(\lambda))$ and with a success probability $\epsilon'(t(\lambda), \epsilon(\lambda))$. We may approximate an upper bound for the success probability ϵ' as less than that of the best known algorithm for solving the underlying problem in time t' (determined through experimental results). It is

then possible to derive an upper bound for ϵ; and so a lower bound for the security parameter λ above which the probability that an attacker breaks the scheme can be estimated to be below a given security value. Hence, we can estimate the values of the security parameter for which the scheme is secure.

Opponents of this interpretation, for example Koblitz & Menezes (2004), claim that the quality of the reduction (i.e. the relationship between ϵ and ϵ') provides a false metric for the security of the scheme. Specifically, they claim that efficient cryptosystems often make use of adapted versions of known mathematical problems, and so reducing to the 'standard' version of a problem results in a comparatively inefficient security reduction and unnecessarily high estimates for the security parameter. Koblitz and Menezes point out that our choice of standard underlying hard problems is entirely arbitrary and any slight reworking of this set could completely change our view on which schemes are the most secure.

A further controversial trend is the proliferation of the use of new, unstudied security assumptions. As cryptographers attempt to produce security proofs that do not rely on the random oracle model (see §3), they are increasingly extending the 'standard' set of underlying problems. The hardness of many of these new problems are justified with a simplified analysis in the generic group model (Shoup 1997), despite the fact that it has been shown that the generic group model suffers from the same theoretical weaknesses as the random oracle model (Dent 2002). While computational number theorists are attempting to evaluate the hardness of these new problems, their production is far outpacing the ability of the cryptographic community to evaluate them.

(*e*) *Deriving the reduction: can we detect the simulation?*

There are some fundamental difficulties in implementing reductionist security arguments. As previously mentioned, cryptographers typically prove the security of a cryptosystem by assuming the existence of an attacker who can break the cryptosystem and then using that attacker as a subroutine in a larger algorithm that solves the underlying problem. The assumption that there exist no efficient algorithms that solve the underlying problem implies that there are no attackers who can break the cryptosystem; this is a well-known technique in complexity theory. Unfortunately, there is a difference between a complexity theoretic reduction and the kinds of reduction used in proofs of security. In order to construct a complexity theoretic reduction, one simply has to find a way to phrase an instance of one problem as an instance of the other problem. This is not true when reducing the security of a cryptosystem to the difficulty of solving a mathematical problem.

Recall that in a security model, the attacker normally does more than just receive an instance of the cryptosystem to break. Often, in a security model, the attacker may also query other entities in the system (for example, the challenger in a game-based security model or the environment in a simulation-based security model). These entities compute values and return the results to the attacker, and are modelled as oracles to which the attacker has access. Thus, in order to prove the security of a cryptosystem, it is not only necessary to phrase the instance of the underlying problem as an instance of the problem of breaking the cryptosystem, it is also necessary to make sure that the responses to the attacker's oracle queries

are correct. It is the problem of responding to these oracle queries that typically makes producing security proofs so difficult.

It is frustrating that many security proofs cannot be completed, or require additional assumptions, owing to the problems associated with correctly responding to 'trivial' oracle queries. A trivial oracle query is one in which the attacker already knows the response that it should receive from an oracle before it make the query: thus, the query does not help them break the cryptosystem in any way, but it does allow them to detect whether the oracle is responding correctly or not.

Some progress has been made in overcoming this problem. For an asymmetric encryption scheme, cryptographers have developed the notion of *plaintext awareness* (Bellare et al. 1998; Bellare & Palacio 2004). An encryption scheme is plaintext aware if, for every algorithm that can produce ciphertexts, there exists an algorithm that can extract the underlying message from the ciphertext by observing the execution of the algorithm that produced it. Sadly, this approach has not yet yielded much in the way of practical results. Another interesting idea is that of Barak (2001) who demonstrates that we can achieve provable security without having to execute an attacker as a self-contained (black-box) subroutine. This idea has yet to be fully explored in mainstream provable security research.

3. The random oracle model

(*a*) *The use of the random oracle model in cryptography*

Possibly the most controversial issue in provable security research is the use of the random oracle model (Bellare & Rogaway 1993). The random oracle model involves modelling certain parts of cryptosystems, called *hash functions*, as totally random functions about whose internal workings the attacker has no information. This theoretical model vastly simplifies the analysis of cryptosystems and allows many schemes to be 'proven' secure that would otherwise be too complicated to be proven secure.

A hash function is a keyless algorithm that takes arbitrary-length inputs and outputs a fixed-length *hash value* or *hash*. There are several properties that one would expect a hash function to exhibit, including pre-image resistance (given a random element of the output set, it should be computationally infeasible to find a pre-image of that element) and collision resistance (it should be computationally infeasible to find two elements that have the same hash value). However, there are many more properties that we might require of a hash function depending on the circumstances. For example, it might be hoped that if the hash function is evaluated on two related inputs, then the outputs will appear unrelated.

From a provable security point of view, hash functions present a difficult problem. They are usually developed using symmetric techniques, either as stand-alone algorithms or based on the use of a block cipher. Thus it is difficult to apply the reductionist theory of provable security to them because there are no natural candidate problems to which we may reduce the security. There are constructions of hash functions from block ciphers for which it can be proven that the hash function has certain properties (such as pre-image and collision resistance) as long as the underlying block cipher is indistinguishable from a random permutation (see §2 *b*);

however, it is impossible for *any* publicly-known function to produce outputs that appear independent when evaluated on two known inputs.

The random oracle model attempts to overcome our inability to make strong statements about the security of hash functions by modelling them as completely random functions about which an attacker has no information. The attacker (and all other parties in the security model) may evaluate such a random hash function by querying an oracle. The original interpretation of this simplification was that it heuristically demonstrated that a cryptosystem was secure *up to attacks against the system that may be introduced via the use of a specific hash function.* Equivalently, it was thought that a proof of security in the random oracle model meant that, with overwhelming probability, the cryptosystem was secure when instantiated with a randomly chosen hash function.

This interpretation of the random oracle model is correct up to a point. It is possible to construct families of efficient hash functions for which it is computationally infeasible to differentiate between access to an oracle which computes a randomly selected hash function, and access to an oracle which computes a random function. If such a hash function is used in place of the random oracle, then we can be sure that the scheme is secure against attackers whose only interaction with the hash function is to directly compute the output of the hash function on certain inputs. This is subtly different from the interpretation of the random oracle model given above.

The one key difference between the random oracle model and the use of a hash function selected at random from a random-looking function family is that in the latter case the attacker is given access to a description of a Turing machine that can compute the hash function; in the former the attacker is not given such a description. This led to the cataclysmic result of Canetti *et al.* (2004) who demonstrated that it was possible to have a scheme that was provably secure in the random oracle model, and yet insecure when the random oracle was replaced *with any hash function.* The trick Canetti *et al.* employ is to use knowledge of the Turing machine that computes the hash function like a password that forces the cryptosystem to release sensitive information (such as its private key).

As an example, consider the formal game-based security model for an asymmetric encryption scheme. In this model, the cryptosystem is represented as three separate polynomial-time algorithms: a probabilistic key generation algorithm $\mathcal{G}$ that takes as input the security parameter in unary format 1^k, and outputs a public key pk and a private key sk; a probabilistic encryption algorithm $\mathcal{E}$ that takes as input the public key pk and a message m drawn from a message space $\mathcal{M}$ that is defined by the public key, and outputs a ciphertext C; and a deterministic decryption algorithm $\mathcal{D}$ that takes as input the secret key sk and a ciphertext C, and returns either a message $m \in \mathcal{M}$ or the error symbol $\perp$. For an arbitrary, probabilistic polynomial-time attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and a security parameter k, the security model is as follows:

1. The challenger generates an asymmetric key-pair $(pk, sk) = \mathcal{G}(1^k)$.
2. The attacker $\mathcal{A}_1$ is executed on the input $(1^k, pk)$. During its execution, $\mathcal{A}_1$ may query a decryption oracle with a ciphertext C. This decryption oracle returns $\mathcal{D}(sk, C)$ to the attacker. $\mathcal{A}_1$ terminates by outputting distinct equal-length messages (m_0, m_1) and some state information *state*.

3. The challenger randomly selects a bit $b \in \{0, 1\}$ and computes $C^* = \mathcal{E}(pk, m_b)$.
4. The attacker $\mathcal{A}_2$ is executed on the input $(C^*, state)$. As before, during its execution, $\mathcal{A}_2$ may query a decryption oracle; however now we forbid $\mathcal{A}_2$ to query the decryption oracle on C^*. $\mathcal{A}_2$ terminates by outputting a bit b'.

The attacker is deemed to have won the game if $b' = b$ and the cryptosystem is considered secure if the attacker's advantage

$$|Pr[b = b'] - 1/2| \tag{3.1}$$

is negligible as a function of the security parameter. An encryption scheme which is secure in this model is said to be IND-CCA2 secure. This has been shown to capture the notion that an attacker cannot gain any information about a message from its encryption. A cryptosystem that is secure against all attackers that do not make any decryption oracle queries is said to be IND-CPA secure.

The typical construction of an asymmetric encryption schemes that are IND-CCA2 secure in the random oracle model but insecure when the random oracle is instantiated with any hash function involves adapting an IND-CCA2 secure encryption scheme so that the decryption algorithm checks whether the ciphertext C contains a description of the hash function before continuing with its normal decryption routine. If the ciphertext does contain such a description, then the decryption algorithm returns sk. The Canetti *et al.* paper proves this using a game-based notion of security, but a similar separation result has been proven for simulation-based notions of security (Nielsen 2002).

Recently we have seen the emergence of practical and provably secure cryptosystems that do not require the random oracle model in their proofs of security (Cramer & Shoup 2003; Boneh & Boyen 2004). This is clearly a huge step forward for provable security research. However, these schemes are currently either less efficient than their counterparts that are proven secure in the random oracle model, or are based on less-studied mathematical problems (see §2 *d*).

(*b*) *Random oracles and zero-knowledge protocols*

For game-based security models, all known proofs for the separation between the random oracle model and the standard (real-world) model are based on the Canetti *et al.* trick of passing a (binary) description of the hash function to the challenger as part of an oracle query. It is therefore natural to ask whether this is the only way in which a cryptosystem might be provably secure in the random oracle model, yet insecure when that oracle is instantiated with any hash function. If so, then an examination of the algorithms of a cryptosystem might be enough to (heuristically) convince users that this situation does not occur and therefore that a proof of security in the random oracle model is sufficient.

One approach to this problem might be to consider an extended version of the random oracle model in which the attacker is given some form of identifier which uniquely identifies the hash function in use and allows the evaluation of that hash function on arbitrary inputs, but does not give any information about the internal structure of the hash function. For example, one may consider using code obfuscation to disguise the internal workings of the hash function, or encrypting the hash function and providing the attacker with an oracle that executes encrypted

code. Sadly, this does not appear to work. The former approach fails because it is impossible to provide sufficiently strong code obfuscation (Barak *et al.* 2001). The latter approach fails because we may construct schemes that are provably secure in this 'encrypted random oracle model', but insecure in the standard model. These examples use knowledge of the key used to decrypt the hash function as a 'password' in exactly the same way that Canetti *et al.* used the hash function code.

Thus we are forced to look for another way to view the separation between the random oracle model and the standard model. The remainder of this section will be used to describe an interesting quirk that can either be viewed as a positive property of the separation, or as a potential method of showing that the separation is not as serious as has been previously thought.

Let $\mathcal{F}$ be a family of hash functions that are indistinguishable from random (i.e. no polynomial-time algorithm can determine whether it has oracle access to a randomly chosen function $f \in \mathcal{F}$ or a completely random function). Suppose that an entity, called the *verifier*, has oracle access to a randomly chosen function $f \in \mathcal{F}$. Suppose further that a second entity, known as the *prover*, wishes to prove to the verifier that it knows the description of the hash function's program code $[f]$ but doesn't want the verifier to gain any knowledge about $[f]$ from their exchange.

If the verifier is the only entity that has access to the oracle that computes f, then the prover can easily demonstrate knowledge of $[f]$ simply by computing $f(r)$ for values of r chosen by the verifier. This protocol is zero knowledge (i.e. the verifier doesn't gain any information about $[f]$) because the verifier only receives information that it could have computed.

However, if both the prover and the verifier have oracle access to f, then the situation becomes much more difficult. Of course, the prover could just release $[f]$ to the verifier, but this is not zero-knowledge as the verifier leans $[f]$. If there exists a cryptosystem that is provably secure (for a game-based notion of security) in the random oracle model, but insecure whenever the random oracle is instantiated with a real function, then we may construct a protocol for the prover to demonstrate knowledge of $[f]$. In this protocol, the verifier simply acts as the challenger in the security model, and the prover acts as an attacker. If the prover has no knowledge of the hash function description $[f]$, then the prover will be unable to break the cryptosystem as the cryptosystem is secure in the random oracle model. If the prover does know $[f]$ then it is able to break the scheme because the scheme is insecure when the scheme is instantiated with any real function.

Of course, we do not know if such a protocol is zero-knowledge or not. However, if the protocol is not zero-knowledge (for an honest verifier who faithfully follows the role of the challenger), then we know that any attacker who breaks the cryptosystem using knowledge of $[f]$ must leak some information about $[f]$ to the challenger. We conjecture that in such a situation, the attacker must be using a Canetti *et al.* style attack; attacks that can be effectively disregarded in practice. Hence, we conjecture that *either* there exists an honest-verifier zero-knowledge protocol for demonstrating knowledge of $[f]$ *or* security proofs in the random oracle model are sufficient in practice.

Another interesting point about Canetti *et al.* style attacks is that they all make use of the attacker's ability to make oracle queries in the security model, for example, decryption oracle queries in the security model for an asymmetric encryption scheme (see §3 *a*). We do not know of any example of any asymmetric encryption

scheme that is IND-CPA secure in the random oracle model, but insecure when the random oracle is instantiated with any hash function. If such a cryptosystem $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ existed and we assume, without loss of generality, that $\mathcal{M} = \{0, 1\}$ for all values of the security parameter, then the protocol for proving knowledge of $[f]$ reduces to:

1. The verifier computes $(pk, sk) = \mathcal{G}(1^k)$, generates a random bit $b \in \{0, 1\}$ and computes $C^* = \mathcal{E}(pk, b)$. The verifier sends $(1^k, pk, C^*)$ to the prover.

2. The prover executes the attacker algorithm that breaks the encryption scheme and recovers a guess b' for b. This value is sent to the verifier.

3. The verifier accepts the prover's claim if $b = b'$.

This is a zero-knowledge protocol for any honest verifier. Therefore, either there exists a two-round honest-verifier zero-knowledge protocol that demonstrates knowledge of $[f]$, or a proof of IND-CPA security in the random oracle model is sufficient to guarantee security.

4. The future?

The next decade will decide whether provable security has a future in practical cryptography, or whether it will be banished back to the realms of 'theoretically interesting' science. This will be largely determined by how well cryptographers overcome the fundamental problems that we have discussed.

It is clear that neither game-based, nor simulation-based, models of security are entirely adequate. The game-based models do not give the security guarantees that one requires, and the simulation-based models cannot be used to prove the security of certain types of scheme (even up to the barrier imposed by the question of whether P=NP or not). Since simulation-based security models were developed to overcome the problems in game-based models, it should be hoped that researchers will once again return to first principles in an attempt to produce a comprehensive model for security. Sadly, I am unaware of any group attempting to do this, and it is unclear whether this daunting line of research will be pursued.

A similar situation exists for the random oracle model; however, in this case, I do not believe the future is quite as bleak. While it is true that many researchers are still studying the negative aspects of the random oracle model, i.e. proving the inadequacy of the random oracle model in various situations, research is beginning on the more positive aspects of the model. Towards this end, this paper suggests that separation between the random oracle model and the standard model is intrinsically linked to certain problems connected with zero-knowledge proofs. Hopefully the cryptographic community will accept that the random oracle model will never disappear until we can construct simple and efficient cryptosystems that are provably secure in the standard model, and therefore begin to study the ways in which the model succeeds as well as the ways in which it fails.

One of the major stumbling blocks in constructing proofs of security for simple and efficient cryptosystems without using the random oracle model is the difficulties involved in responding to 'trivial' oracle queries. On the path towards eliminating the random oracle model entirely, solving the problem of trivial oracle queries would

be a major step forward. Researchers in this field are currently highly active, particularly in connection with the ideas presented by Bellare and Palacio (2004) on plaintext awareness. We will know within the next couple of years whether this is likely to be a fruitful future avenue of research. In an interesting twist, it seems possible that the problem of responding to trivial oracle queries may be connected to the problem of analysing the zero-knowledge protocols presented in §3.

Another area that I would expect to see explored over the next few years is the analysis of the fundamental problems on which many of the newer cryptosystems are based. The current trend of proposing new, unusual mathematical problems simply so that one can gain a proof of security must eventually come to a halt, and it seems most likely that this will occur because of a significant advance in the field of computational number theory and the subsequent collapse of several publicised 'provably secure' cryptosystems.

The author would like to thank Maria Petagna, Fred Piper, James Birkett, Nigel Smart, Marc Fischlin, Kenny Paterson, John Malone-Lee and Christine Swart for their comments and discussions on this paper. The author would also like to thank the EPSRC for their generous financial assistance.

References

Barak, B. 2001 How to go beyond the black-box simulation barrier. In *Proc. 42nd IEEE Annual Symp. on Foundations of Computer Science, Las Vegas, NV, USA, 14–17 October 2001*, pp. 106–115.

Barak, B., Goldrecih, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S. & Yang K. 2001 On the (im)possibility of obfuscating programs. In *Advances in Cryptology - CRYPTO 2001, Proc. 21st Annual International Cryptology Conference, Santa Barbara, California, USA, 19–23 August 2001* (ed. J. Kilian). Springer-Verlag Lecture Notes in Computer Science, no. 2139, pp. 1–18.

Bellare, M. 1997 Practice-oriented provable-security. In *Lectures on Data Security: Modern Cryptology in Theory and Practice* (ed. I. Damgård). Springer-Verlag Lecture Notes in Computer Science, no. 1561, pp. 1–15.

Bellare, M. & Palacio, A. 2004 Towards plaintext-aware public-key encryption without random oracles. In *Advances in Cryptology - ASIACRYPT 2004, Proc. 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, 5–9 December 2004* (ed. P. J. Lee). Springer-Verlag Lecture Notes in Computer Science, no. 3329, pp. 48–62.

Bellare, M. & Rogaway, P. 1993 Random oracles are practical: a paradigm for designing efficient protocols. In *Proc. 1st ACM conference on computer and communications security, Fairfax, VA, USA, 3–5 November 1993*, pp. 62–73.

Bellare, M., Desai, A., Jokipii, E. & Rogaway, P. 1997 A concrete security treatment of symmetric encryption. In *Proc. 38th Annual Symp. Foundations of Computer Science, Washington, DC, USA, 19–22 October 1997*, pp. 394–405.

Bellare, M., Desai, A., Pointcheval, D., & Rogaway, P. 1998 Relations among notions for public-key encryption schemes. In *Advances in Cryptology - CRYPTO '98, Proc. 18th Annual International Cryptology Conference, Santa Barbara, CA, USA, 23–27 August 1998* (ed. H. Krawczyk). Springer-Verlag Lecture Notes in Computer Science, no. 1462, pp. 26–45.

Boneh, D. & Boyen, X. 2004 Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2004, Proc. International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004* (eds

C. Cachin & J. Camenisch). Springer-Verlag Lecture Notes in Computer Science, no. 3027, pp. 56–73.

Canetti, R. 2001 Universally composable security: A new paradigm for cryptographic protocols. In *Proc. 42nd IEEE Annual Symp. on Foundations of Computer Science, Las Vegas, NV, USA, 14–17 October 2001*, pp. 136–145.

Canetti, R. & Fischlin, M. 2001 Universally composable commitments. In *Advances in Cryptology – CRYPTO 2001, Proc. 21st Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 August 2001* (ed. J. Kilian). Springer-Verlag Lectures Notes in Computer Science, no. 2139, pp. 19–40.

Cannetti, R., Goldreich, O. & Halevi, S. 2004 The random oracle methodology, revisited. *J. ACM* **51(4)**, 557–594.

Cramer, R. and Shoup, V. 2003 Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comp.* **33(1)**, pp. 167-226.

Dent, A. W. 2002 Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptoloyg – ASIACRYPT 2002, Proc. 8th International Conference on the Theory and Applicatino of Cryptology and Information Security, Queenstown, New Zealand, 1–5 December 2002* (ed. Y. Zheng). Springer-Verlag Lecture Notes in Computer Science, no. 2501, pp. 100-109.

Diffie, W. & Hellman, M. E. 1976 New directions in cryptography. *IEEE Trans. Inf. Th.* **22**, 644–654.

Goldwasser, S., Micali, S. & Rivest, R. 1988 A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing* **17(2)**, 281–308.

Koblitz, N. & Menezes, A. J. 2004 Another look at "provable security". Preprint, University of Washington & University of Waterloo.

Nielsen, J. B. 2002 Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *Advances in Cryptology – CRYPTO 2002, Proc. 22nd Annual International Cryptology Conference, Santa Barbara, CA, USA, 18-22 August, 2002* (ed. M. Yung). Springer-Verlag Lecture Notes in Computer Science, no. 2442, pp. 111 - 126.

Pfitzmann, B. & Waidner, M. 2000 Composition and integrity preservation of secure reactive systems. In *Proc. 7th ACM Conf. Computer and Communications Security, Athens, Greece, 1–4 November 2000*, pp. 245–254.

Rabin, M. O. 1979 Digitalized signatures and public-key functions as intractable as factorization. Report no. MIT/LCS/TR-212, MIT Laboratory for Computer Science, Massachusetts Institute of Technology.

Rackoff, C. & Simon, D. R. 1991 Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology – CRYPTO '91, Proc. 11th Annual International Cryptology Conference, Santa Barbara, CA, USA, 11-15 August 1991* (ed. J. Feigenbaum). Springer-Verlag Lectures Notes in Computer Science, no. 576, pp. 433–444.

Shannon, C. E. 1948 A mathematical theory of communication. *Bell System Technical J.* **27**, 379–423, 623–656.

Shannon, C. E. 1949 Communication theory of secrecy systems. *Bell System Technical J.* **28**, 656–715.

Shoup, V. 1997 Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology - EUROCRYPT '97, Proc. International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, 11–15 May 1997* (ed. W. Fumy). Springer-Verlag Lecture Notes in Computer Science, no. 1233, pp. 256–266.

Turing, A. M. 1936 On computable numbers, with an application to the Entscheidungsproblem. *Proc. Lon. Math. Soc. ser. 2* **42**, 230–265.

AUTHOR PROFILE

Alexander W. Dent

Alex Dent became interested in cryptography at the age of eight, after reading a Dangermouse book on codes and ciphers; however, his formal study of the subject didn't begin until his undergraduate career. Alex graduated with a first class M.Math degree from the University of Oxford, where his interest in cryptography was reawakened by a course in cryptography and complexity theory. Alex studied for his doctorate in mathematics at Royal Holloway, University of London, and graduated in 2002. After submitting his thesis, he was immediately employed as a post-doctoral research assistant for the European Union's cryptographic algorithm assessment project, NESSIE, and played a key role in assessing the security of several important asymmetric encryption schemes. At this point he also began working with the ISO and the IEC committees on the standardisation of secure cryptographic schemes. In 2004 he was awarded a prestigious EPSRC Junior Research Fellowship, and in 2005 he published a textbook (co-authored with Prof. Chris Mitchell) entitled "[A] User's Guide to Cryptography and Standards".

Alex is currently working as a lecturer in the Information Security Group at Royal Holloway, University of London, where he continues his research into both the practical and theoretical aspects of provable security. He claims the best thing about being an academic is the opportunity to travel, especially when it affords him the opportunity to visit his girlfriend in New York City, and he has presented his research on four continents. Currently, he is searching for a cryptographic conference based in Antarctica, as he believes that this will be the most difficult of the remaining continents to visit. He has also discovered that he rather enjoys talking about himself in the third person.