

An Brief Introduction to Provable Security

Dr. Alexander W. Dent
Information Security Group
Royal Holloway, University of London



Overview

- Different types of provable security
- Security models and win conditions
- Reductions
- Game hopping
- Arguments about proofs
- Conclusion



The different types of provable security



Royal Holloway
University of London

Purpose of provable security

- Whenever you propose a cryptosystem, you should provide some indication that it is secure.
- There are lots of ways to provide evidence that a scheme is secure.
- None of them guarantee security.



Royal Holloway
University of London

Purpose of provable security

- Provable security aims to give a mathematical guarantee that a scheme is secure.
- This can't be done.



Royal Holloway
University of London

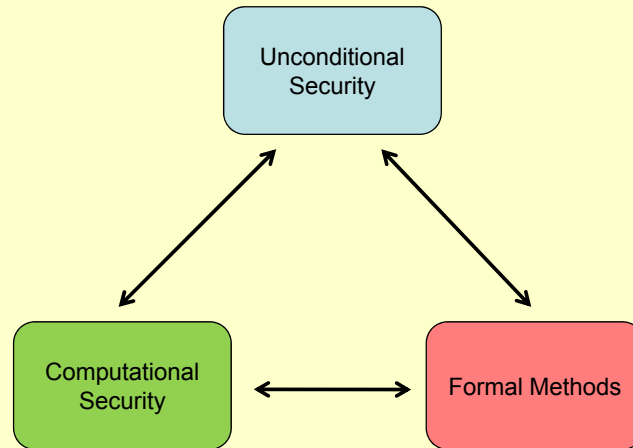
Purpose of provable security

- A mathematical guarantee that a scheme cannot be broken by a class of attackers in a mathematical model of reality.
 - A precise description of the scheme.
 - A precise description of the class of attackers.
 - A precise description of the model .
 - A precise description of the “win condition” .
 - A proof that no attacker can achieve the win condition for that scheme in that model.

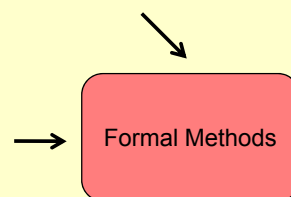


Royal Holloway
University of London

Three different types

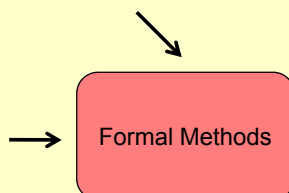


Three different types



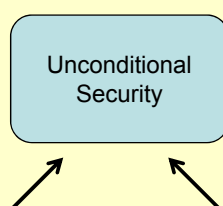
- Characterised by style of proof rather than be class of attackers.
- Use symbolic and logical representations of algorithms and security requirements.
- Proofs use symbolic manipulation.

Three different types



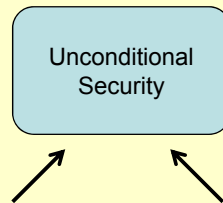
- Verification of protocol security
 - Assumes perfect underlying cryptography
- Verification of algorithmic correctness
- Slowly heading towards verification of algorithmic security

Three different types



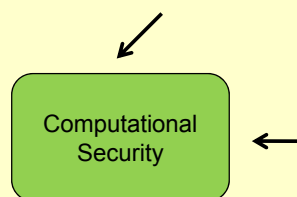
- Characterised by resisting all attackers.
- Began with Shannon's work (1949)
- Typically uses combinatorial techniques to prove theorems.

Three different types



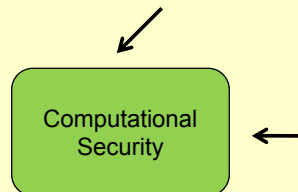
- Key pre-distribution schemes in large sensor networks.
- Frameproof and IPP codes.
- Secret sharing schemes.

Three different types



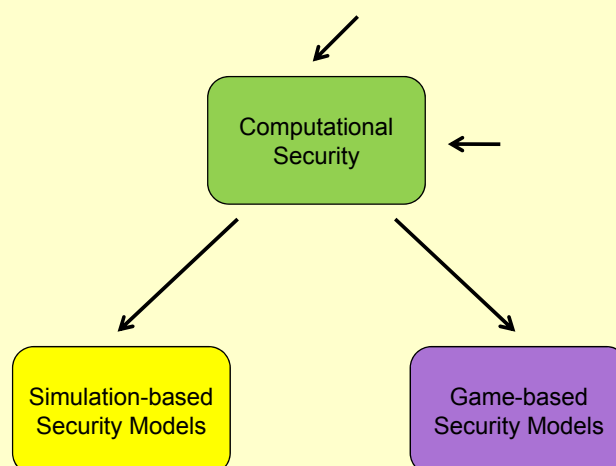
- Characterised by resisting attacks made by PPT algorithms or circuits.
- Most popular branch of provable security.
- Typically uses complexity theoretic techniques to prove security.

Three different types



- Algorithms vs. circuits occasionally a critically important issue.
- Non-uniform algorithms (circuits) are considered to be given a different “hint” for each value of the security parameter.

Five different types?



Five different types?

Game-based
Security Models

- Security is phrased as a game played between an attacker and a hypothetical challenger.
- Challenger's responses are completely defined and so define the security model.



Royal Holloway
University of London

Five different types?

Simulation-based
Security Models

- The attacker interacts with an arbitrary environment which uses a cryptosystem.
- The attacker's and environment's outputs must be simulatable when you replace the cryptosystem with an idealised version.



Royal Holloway
University of London

Five different types?



Simulation-based
Security Models

- Strong security guarantees – the cryptosystem is secure in any system.
- Canetti's UC framework
- Pfitzmann and Waider's composed systems

Security models and “win conditions”

Security models

- In order to prove security, we have to define security.
- This isn't always easy.
- Availability?
- Non-repudiation?



Royal Holloway
University of London

Security models

- The security model defines every possible interaction of the attacker with legitimate users of the cryptosystem.
- It should be strong enough to cover every possible interaction with the system.
- It should be as weak as possible respecting this constraint to allow for simple schemes to be proven secure.



Royal Holloway
University of London

Aren't security models easy?

- In theory each application should have its own security model.
- Easier to use one or two “supermodels”.



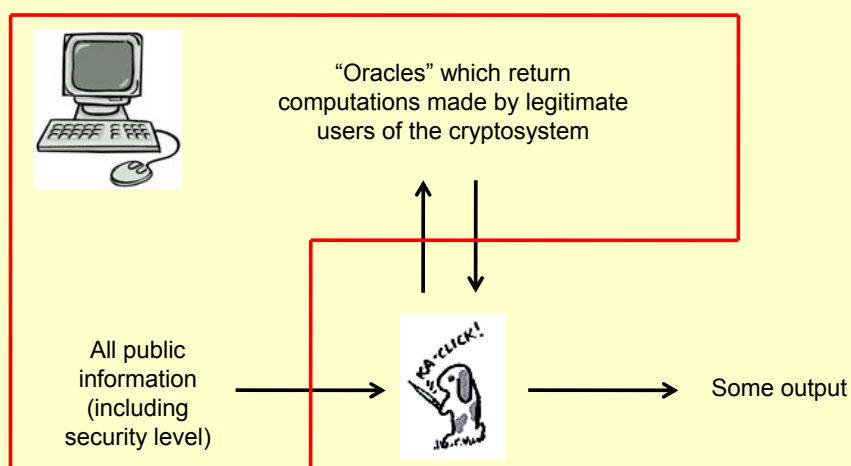
Aren't security models easy?

- In theory each application should have its own security model.
- Easier to use one or two “supermodels”.
- Applications should check whether these models capture practical attacks.
- Albrecht, Watson and Paterson attacks against the “provably secure” SSH protocol work due to poor model choices.

Aren't security models easy?

- On the other hand, security models should be as simple as possible.
- CLE has twenty security models.
- Much of the confusion stems from choices by the initial authors to use an unrealistically strong security model.
- This leads to unnecessarily complex and inefficient implementations.

“Normal” security models



Breaking the system

- The “model” defines the attacker’s possible interactions with the system.
- Also have to define what the attacker needs to do in order to “break” the cryptosystem.
- Typically we assess if “some output” satisfies some condition.

Breaking the system



→ Some output

- We may wish to minimise the probability that “some output” satisfies the condition.
- For example, we may want to show that the probability that the attacker outputs a new valid message and signature is small.

Breaking the system



→ Some output

- We may wish to show that the probability that the “some output” satisfies the condition is no better than by chance.

$$\text{Adv} = |\Pr[\mathcal{A} \text{ guesses } b] - \frac{1}{2}|$$



Royal Holloway
University of London

Breaking the system

- Sometimes an alternative formulation for advantage is used:

$$\text{Adv} = |\Pr[\mathcal{A} \text{ outputs } 1 \mid b=1] - \Pr[\mathcal{A} \text{ outputs } 1 \mid b=0]|$$

$$= 2 \times |\Pr[\mathcal{A} \text{ guesses } b] - \frac{1}{2}|$$



Royal Holloway
University of London

Breaking the system

- Again, the “win condition” defines the ways that the attacker can damage the system.
- It should be strong enough to capture all the ways that the attacker can cause damage.
- It should be as weak as possible respecting this condition to prevent the need for inefficient schemes.



Royal Holloway
University of London

Concrete vs Asymptotic security

- Big argument about whether security should be concrete or asymptotic.
- Asymptotic security considers PPT algorithms and “negligible” advantages.
- Concrete security considers t -time attackers with ϵ -advantages.
- Controversial statement:
- Personally, I don't really care.



Royal Holloway
University of London

Reductions



Royal Holloway
University of London

How to prove security?

- Proving the security of a scheme is hard.
- Have to show that the success probability is “small” for every PPT attacker.
- The great $NP \neq P$ barrier.

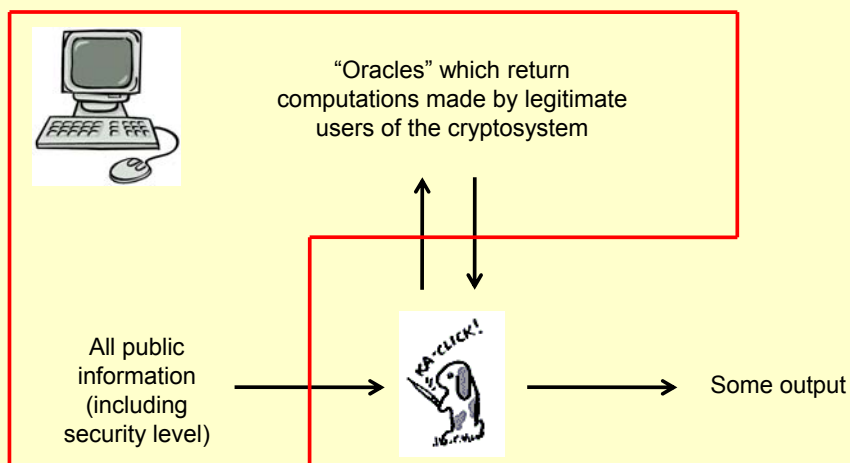


Royal Holloway
University of London

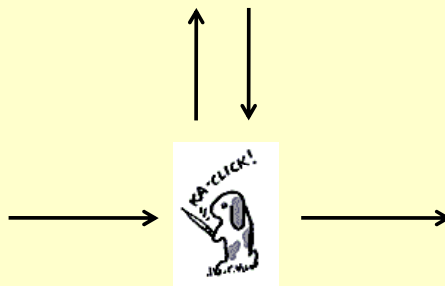
How to prove security?

- Given the $NP \neq P$ barrier, we can only reduce to a problem believed to be hard.
- Still has value: reduces a complex model to a very simple one which can be studied.
- Main weapon is complexity theoretic reductions.

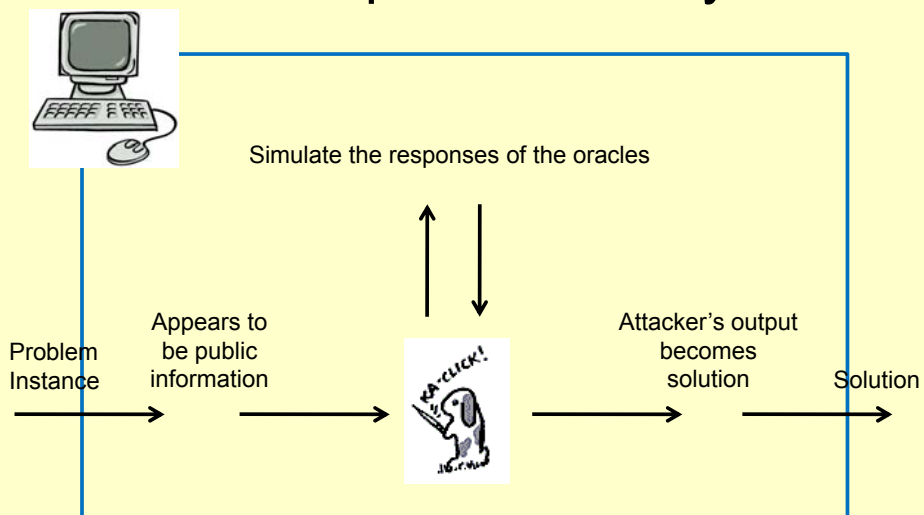
How to prove security?



How to prove security?



How to prove security?



How to prove security?

- For the proof to work, must show:
 - The inputs that the attacker receives look like the inputs it would get in the security model. Otherwise we cannot predict the attacker's solution.
 - If the attacker's succeeds in breaking the scheme, then the algorithm is likely to succeed in solving the problem.



Royal Holloway
University of London

How to prove security?

- Often see proofs in three parts:
 - A description of the simulator
 - A justification of why the simulator provides inputs which look (to the attacker) like those in the security model
 - A justification of why the simulator solves the problem whenever the attacker breaks the cryptosystem



Royal Holloway
University of London

Golden rules

- Every difference between simulator's responses and model's responses must be justified.
 - This includes changes to input distributions.
 - This includes error conditions.
 - Always write out the simulator!
- The attacker is just an algorithm.
 - Never assume that the attacker "knows" a value or that the value can be extracted.

Game hopping

Game hopping

- Proofs involved in a reduction argument:
 - The input distributions are similar to those provided by the challenger in the model.
 - The output of the attacker can be changed into an output for the problem.
- These arguments can become so complex that it's very hard to verify that they are true.



Royal Holloway
University of London

Game hopping

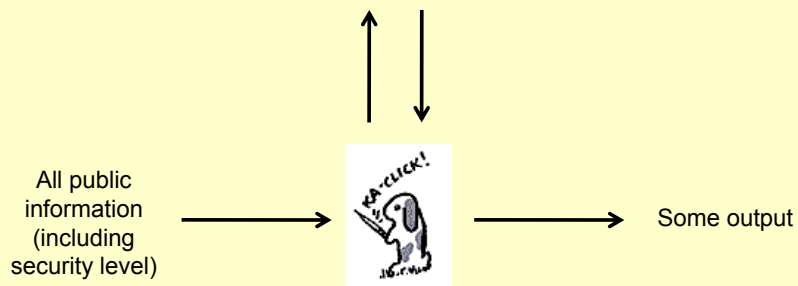
- It may be easy to prove the security of a scheme in an “incorrect” security model.
- It is hard to prove security of the scheme in the “correct” security model.
- Game hopping allows you to change the security model in a series of small steps.
- For each change, you show that the attacker's outputs stay (roughly) the same.



Royal Holloway
University of London

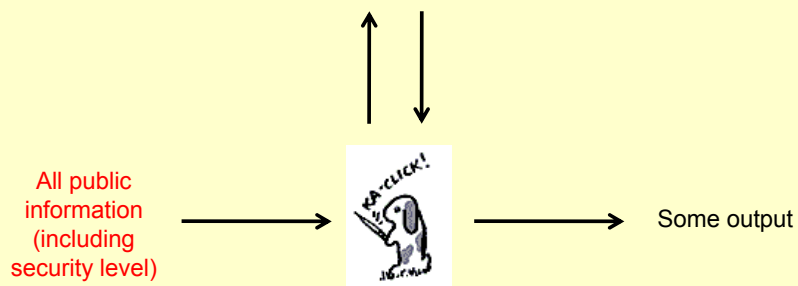
Game 1

“Oracles” which return computations made by legitimate users of the cryptosystem



Game 2

“Oracles” which return computations made by legitimate users of the cryptosystem



Game hopping

- If the attacker's output satisfies the "win condition" (with some probability) in Game 1 ... then it will satisfy the "win condition" (with similar probability) in Game 2.
- And it may be easier to prove that the probability that the attacker's output satisfies with the "win condition" is very small in Game 2.



Game hopping

- Let W_i be the event that the attacker wins in Game i .
- We use a lot of expressions like:

$$\Pr[W_1] = (\Pr[W_1] - \Pr[W_2]) + \Pr[W_2]$$



Game hopping

- In theory, games can change in any way.
- However, there are four recognised types:
 - Bridging steps
 - Distribution changes
 - “Small” error conditions
 - “Large” error conditions



Bridging steps

- All the inputs to the attacker remain the same in Game 1 and Game 2.
- The only difference is the way in which these inputs are calculated.
- For example, we may use the private key to “encrypt” a message rather than the public key.
- Useful for setting things up for later...



Distribution changes

- We may change the distribution of inputs to the attacker as long as the attacker will not notice the difference.
- Game 1 gives the attacker (g, g^a, g^b, g^{ab}) .
- Game 2 gives the attacker (g, g^a, g^b, g^c) .
- If the DDH problem is hard, then the attacker won't notice the difference (providing a, b, c aren't used anywhere)



Distribution changes

- If the distributions are computationally indistinguishable
 - It means that you can produce an algorithm (using the attacker) that will distinguish between the two distributions.
 - Write down the algorithm!

$$\Pr[W_1] \leq \Pr[W_2] + \text{AdvDist}(k)$$



Distribution changes

- If the distributions are statistically indistinguishable
 - Small statistical distance.
 - Allows hops even if the challenger cannot be written as a PPT algorithm.

$$\Pr[W_1] \leq \Pr[W_2] + \Delta(D_1, D_2)$$



Small error conditions

- The two games are identical unless a particular error condition E occurs.
- The probability of this error is small.
- Game 1 uses a randomly chosen “critical value” which is unknown to the attacker.
- Game 2 aborts if the attacker “guesses” this value.



Large error conditions

- The two games are identical unless a particular error condition E occurs.
- The probability that $\neg E$ is small but significant (often $\Pr[\neg E]$ is non-negligible).
- The attacker's actions are independent of the error condition E .

$$\Pr[W_2] = \Pr[\neg E] \cdot \Pr[W_1]$$



Large error conditions

- Game 1 is a multi-user model.
- Game 2 is identical to Game 1 except that the attacker loses unless it succeeds in breaking the scheme for a random user.

$$\Pr[W_2] = \frac{1}{\# \text{ users}} \Pr[W_1]$$

- Also, works for advantage.



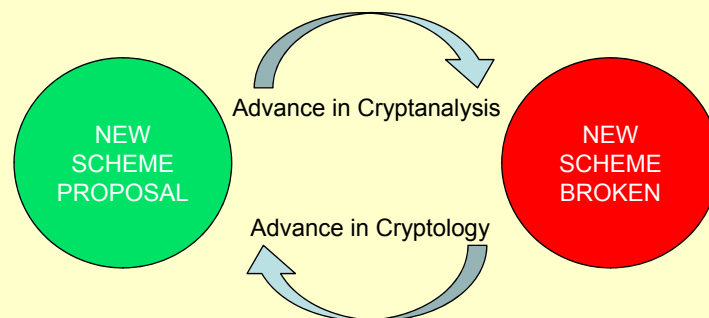
Hop tightness

- Different hops have different tightness:
 - Bridging steps incur no loss in the hop
 - Distribution changes and small error conditions incur a small loss in the hop
 - Large error conditions incur a large loss
- Try to minimise large error conditions
- To get tightest possible reduction, the order of the game hops matter.

Arguments about proofs

Evolution of cryptography

- Traditionally, cryptographic algorithms and protocols were produced by evolution.



Evolution of cryptography

- Evolution is a poor methodology
 - Too few lions thinning the herd
 - No concept of competitive advantage until an algorithm is deployed
 - The effects of a bad design can be extreme
 - Short periods of time for evolution to occur
 - Schemes often selected by reputation rather than by algorithmic design choices

Arguments about proofs

- The development of provable security has been rapid.
- The effects have been profound in terms of standardisation and implementation.
- Koblitz and Menezes produced a series of paper suggesting that our reliance on provable security guarantees is dangerous.

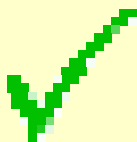


Royal Holloway
University of London

Arguments about proofs



- “There are two unfortunate connotations of “proof”... the first is the notion of 100% security”.
- Security proofs are reductions.
- Breaking the ‘hard’ problem may or may not break the security of the scheme.

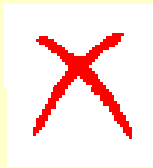


Royal Holloway
University of London

Arguments about proofs



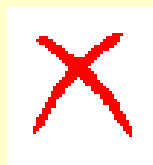
- “[T]he result is conditional in a strong sense ... the condition is of the sort ‘assuming that no one finds an improved algorithm for a certain math problem’”.
- The theorem means itself.
- Improved algorithms affect security regardless of proof.
- Security not a binary state.



Arguments about proofs



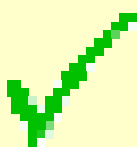
- “[A] provable security theorem only applies to attacks of a specified sort and says nothing about clever attacks that might not be included in the theorem”
- Security applies in the model.
- Model may be incorrectly interpreted or have inherent weaknesses.



Arguments about proofs



- “If the only way to obtain a proof of security ... is to concoct an interactive math problem that people have no desire to study, then it seems to us that the [proof] is not very convincing”.
- Unfortunately, on the rise.



Arguments about proofs



- “[Provable security experts] rarely read other authors’ papers carefully. As a result even the best authors sometimes publish papers with serious errors that go undetected for years”.

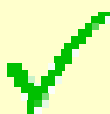


- “Do mistakes happen? Occasionally, though rarely.”

Arguments about proofs



- “[Provable security experts] rarely read other authors’ papers carefully. As a result even the best authors sometimes publish papers with serious errors that go undetected for years”.
- In some conferences, it is rare that a correct proof is presented.



Royal Holloway
University of London

Arguments about proofs

- [KM] argue that if we take the “tightness” of the security reduction into account when we choose key sizes, then the keys almost always become infeasible large.
- Alternatively, if we examine the proof at normal key sizes, then the proofs provide no security guarantees.



Royal Holloway
University of London

Arguments about proofs

- So how do we choose key sizes for a particular scheme?
- Ask an expert in the field!
- The looseness of a reduction does not mean that known techniques are more/less likely to break the scheme.
- Parameter sizes a quirk of history?



Royal Holloway
University of London

Arguments about proofs

Interpretive

- Proof is a poor choice of words for a security reduction.
- Tightness of bounds is unhelpful in determining parameter sizes.
- Apparently similar schemes have unexpectedly differing security results.

Societal

- Results are rarely checked and are often rushed for conferences.
- Schemes without security results are rarely considered by most cryptographers.



Royal Holloway
University of London

Conclusion

Should we have a section heading for one slide?



Royal Holloway
University of London

Conclusions



- “Instead of first setting up a natural cryptographic system and then trying to prove something about it, it has become increasingly common for protocol designers to start with a reductionist security objective that they want to achieve and use it to guide them in their construction of the scheme”.



Royal Holloway
University of London

Selected Bibliography

- Provable Security
 - A. W. Dent. Fundamental problems in provable security and cryptography. In Philosophical Trans of the Royal Society, Series A, vol. 364, no. 1849, pp. 3215--3230, 2006.
- Simulation-based security models
 - R. Canetti. Security and Composition of Cryptographic Protocols: A Tutorial. Available from <http://eprint.iacr.org/2006/465>, 2006.
 - B. Pfitzmann, and M. Waidner. Composition and integrity preservation of secure reactive systems. In Proc. 7th ACM Conference on Computer and Communications Security, pp. 245-254, 2000.
- Game-based security models
 - M. Bellare. Practice-Oriented Provable-Security. In Ivan Damgard, editor, Lectures in Data Security, LNCS 1561, pp. 1-15, 1999.



Royal Holloway
University of London

Selected Bibliography

- Game hopping
 - V. Shoup. Sequences of games: A tool for taming complexity in security proofs. Available from <http://shoup.net/papers>. 2006.
 - M. Bellare and P. Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In S. Vaudenay, editor, Advances in Cryptology – Eurocrypt 2006, LNCS 4004, pp. 409-426, 2006.
 - A. W. Dent. A note on game-hopping proofs. Available from <http://eprint.iacr.org/2006/260>. 2006.



Royal Holloway
University of London

Selected Bibliography

- Attitudes towards provable security
 - N. Koblitz and A. J. Menezes. Another look at “provable security”. *J. Cryptology*, vol. 20, no. 1, 2007.
 - N. Koblitz and A. J. Menezes. Another look at “provable security” II. In R. Barua and T. Lange, editors, *Progress in Cryptology – Indocrypt 2006*, LNCS 4329, pages 148–175, 2006.
 - N. Koblitz. The uneasy relationship between mathematics and cryptography. *Notices of the AMS*, vol. 54, no. 8, pages 972–979, 2007.
(See also responses in letters pages).
 - O. Goldreich. On post-modern cryptography. Available from <http://eprint.iacr.org/2006/461>. 2006.

