How to Evaluate the Security of Real-life Cryptographic Protocols?

The cases of ISO/IEC 29128 and CRYPTREC

Shin'ichiro Matsuo[1], Kunihiko Miyazaki[2], Akira Otsuka[3], and David Basin[4]

[1] NICT
[2] Hitachi Ltd.
[3] AIST
[4] ETH Zurich

Abstract. Governments and international standards bodies have established certification procedures for security-critical technologies, such as cryptographic algorithms. Such standards have not yet been established for cryptographic protocols and hence it is difficult for users of these protocols to know whether they are trustworthy. This is a serious problem as many protocols proposed in the past have failed to achieve their stated security properties. In this paper, we propose a framework for certifying cryptographic protocols. Our framework specifies procedures for both protocol designers and evaluators for certifying protocols with respect to three different assurance levels. This framework is being standardized as ISO/IEC 29128 in ISO/IEC JTC1 SC27/WG3, in which three of the authors are project co-editors. As a case study in the application of our proposal, we also present the plan for the open evaluation of entity-authentication protocols within the CRYPTREC project.

Keyword: Cryptographic protocols, formal verification, standardization

1 Introduction

1.1 Background

Over the past 20 years, many security technologies have been developed using cryptographic protocols. For example, the widely deployed Secure Socket Layer (SSL) protocol, uses a combination of digital signatures, public key cryptography, and symmetric key cryptography. From the viewpoint of users of such security technologies, a major concern is whether they should trust their security.

For cryptographic algorithms, such as block ciphers, stream ciphers, hash functions, and public key encryption, open competitions are held by NIST, NESSIE, ECRYPT, and CRYPTREC. Thanks to such procedures, national and international organizations can select standard cryptographic algorithms and have confidence in the trustworthiness of the results. Moreover, governments can make recommendations for particular application domains, such as e-government or military systems, where algorithms are selected that meet the domain-specific requirements. Again, these recommendations provide a starting point for companies and administrations building trustworthy systems.

In contrast to cryptographic *algorithms*, analogous evaluation procedures do not exist for cryptographic *protocols*. In practice, cryptographic protocols are often designed in industry by inexperienced engineers, who lack a deep knowledge of cryptography. Even for protocols that make their way to international standards, few cryptographers participate in the review process. The resulting protocols are often flawed, e.g., the vulnerabilities in the international standard ISO/IEC 11770-2 key-establishment protocol.

We propose here standardization activities for cryptographic protocols, analogous to those for cryptographic algorithms. Namely, a clearly defined evaluation process should be used where the evaluation results are certified by national and international organizations. By defining a clear evaluation process based on scientifically well-founded methods, the resulting protocols can be widely trusted and used as building blocks for security-critical systems. This would result in a substantial improvement over the current situation, where protocols are proposed and standardized (e.g., within organizations like the IETF and IEEE) without such a process. Moreover, once this process is standardized as a third party certification scheme, such as the Common Criteria or ISO/IEC 15408, newly developed cryptographic protocols may be certified to be secure under this process. This opens up the playing field for developing certified, internationally-recognized security protocols that can be widely accepted and deployed.

The starting points for our proposal are the different formal methods that currently exist for (symbolic) protocol verification. Experience shows that existing verification methods and associated tools can detect many flaws in standard cryptographic protocols. In doing so, the results can be used to improve the quality of the resulting protocols and ultimately to prove their correctness. These tools have become increasingly mature in recent years and can now provide a fine-grained analysis of the security of cryptographic protocols, which is lacking in less-principled engineering methods.

We believe such tools are now ready to be used to aid the design and, in particular, the certification of cryptographic protocols. Hence we propose a process based on the use of such tools to evaluate protocols with respect to different levels of assurance. Our evaluation process is generic: protocol designers and national organization should be able to apply it uniformly to certify a wide variety of cryptographic protocols.

1.2 Contributions

To begin with, we classify the state-of-the-art in security protocol analysis methods into three categories. Our classification is based on the capability of the method used, the skill required by the designer to use the method, and the security requirement of the protocol in question. Afterwards, we propose a certification process, which certifies the result of a security analysis performed by the protocol designer. Because the process of designing cryptographic protocols is similar to designing cryptographic products, our process is analogous to the Common Criteria. Moreover, three of the authors are project editors of the ISO standard, ISO 29128 "Verification of Cryptographic Protocols," which standardizes the above certification process. Currently this standardization is in the Committee Draft (CD) process and is under discussion. We outline this standard as well as some of the issues that have arisen during the standardization process.

We also report on a plan for the evaluation of cryptographic protocols by CRYPTREC, which is the Japanese governmental organization certifying cryptographic techniques. CRYPTREC is leading a standardization effort for entity-authentication protocols, which is taking place through 2013. The call for protocol contributions has already been made and CRYPTREC will use formal methods to evaluate the incoming proposals. Hence, this will be a good example of the application of ISO 29128. We take stock of the current plans for this evaluation.

2 Evaluation of Cryptographic Protocols

2.1 Formal Methods for cryptographic protocol analysis

Designing cryptographic protocols is a very challenging problem. In open networks, such as the Internet, protocols should work even under worst-case assumptions, namely messages may be eavesdropped or tampered with by an attacker (also called the intruder or adversary) or dishonest or careless principals. Surprisingly, severe attacks can be conducted even without attacking and breaking cryptography, but rather by attacking communication itself. These attacks exploit weaknesses in the protocol's design whereby protocols can be defeated by cleverly manipulating and replaying messages in ways not anticipated by the designer. This includes attacks such as: *man-in-the-middle attacks*, where an attacker is involved in two parallel executing sessions and passes messages between them; *replay attacks*, where messages recorded from previous sessions are played in subsequent ones; *reflection attacks*, where transmitted information is sent back to the originator; and *type flaw (confusion) attacks*, where messages of different types are substituted into a protocol (e.g., replacing a name with a key). Typically, these attacks are simply overlooked, as it is difficult for humans, even by a careful inspection of simple protocols, to determine all the complex ways that different protocol sessions could be interleaved together, with possible interferences coming from a malicious intruder.

What is needed are methods to speed up the development and analysis of cryptographic protocols. Moreover, if these methods are to be used to certify protocols, then they must be mathematically precise, so that exact statements are possible about the scope and significance of the analysis results. This role can be filled by formal methods.

Over the last two decades, the security community has made substantial advances in developing formal methods for analyzing cryptographic protocols and thereby preventing the kinds of attacks mentioned above. These methods and tools can be categorized by several points of view. Here we categorize them by

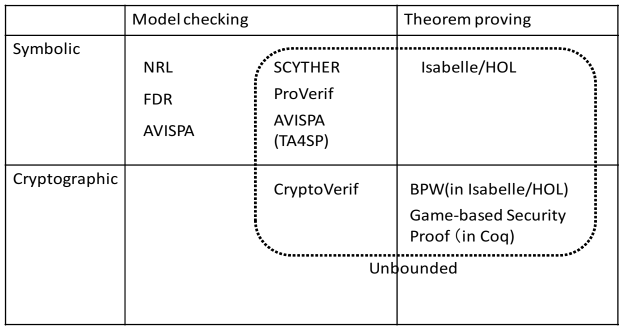

Fig. 1. Categorization of Formal Methods for cryptographic protocol analysis

"Symbolic versus Cryptographic", "Bounded versus Unbounded", and "Model checking versus Theorem proving" as follows (Fig.1)

Model Checking versus Theorem Proving Model checking establishes that a model M, typically formalized as a Kripke structure, has a property ϕ, i.e., $M \models \phi$. Model checking is a form of *algorithmic verification*, as opposed to *deductive verification*, in that $M \models \phi$ is established by executing an algorithm, rather than constructing a proof in some deductive system. Many model checking problems in security (e.g., secrecy and also authentication, see [1]) can be reduced to reachability problems, at which point the model checking algorithms amount to state enumeration. When the state space is finite, model checking constitutes a decision procedure. Initial work on model checking for cryptographic protocols began in the 1980s, starting with Kemmerer's InaTest tool [2]. Since then many successful methods and tools have been developed such as NRL [3], CSP and FDR [4, 5], OFMC [6, 7] and the AVISPA tool [8], ProVerif [9, 10, 1, 11], CryptoVerif [12], and SCYTHER [13].

In theorem proving, one reduces verification to proving a theorem in first-order or higher-order logic. The model M formalizes directly the semantics of the protocol as a set of traces, i.e., the sequence of communication events that result from interleaving runs of the protocol between different principals as well as interference from the intruder. The drawback is that inductive theorem proving requires considerable expertise as well as substantial time and effort. Still, in the hands of an experienced user, this approach has been shown to be effective for verifying protocols with respect to unbounded protocol models. In theorem proving, the *inductive approach* developed by Larry Paulson [14] has been used extensively.

Bounded versus Unbounded Protocols can often be attacked by cleverly manipulating and replaying messages in ways not anticipated by their developers. Such attacks can be quite complex and, in particular, they may require multiple parallel executing sessions. For this reason, it is necessary to model (in M) the possibility of principals participating in an *unbounded* number of protocol sessions. However, even with simple, abstract, term-based models, the general security problem is undecidable [15].

One strategy for handling this complexity is to carry out verification by interactive theorem proving, thereby shifting the complexity to the human who guides the theorem prover. This is the case, for example, when constructing proofs using Isabelle/HOL, as in Paulson's method.

Alternatively, if we are interested in automatic verification, then essentially two options are available: to bound the model so that the problem becomes decidable, or to attempt to produce, algorithmically, a finite characterization of the infinite set of reachable states (or traces) in the unbounded model. Most of model checkers choose the first option, but some advanced model checkers, such as AVISPA with TA4SP [16] backend, ProVerif [9, 10, 1, 11], CryptoVerif [12], and SCYTHER [13], realize the second option.

Symbolic versus Cryptographic The standard Dolev-Yao model is employed in most formal methods for analyzing cryptographic protocols. This model provides a strong idealization of actual cryptographic operations by representing them as term constructors (function symbols) in a term algebra with cancellation rules. This idealization, which we call here the *symbolic approach*, simplifies proof construction by freeing proofs from cryptographic details such as computational restrictions, probabilistic behavior, and error probabilities.

In contrast to this is the *cryptographic approach* (also called the *computational-complexity approach* or *provable security*), where proofs are constructed by reduction, as in complexity theory [17]. Under this approach, one reduces the security of the overall system to the security of the cryptographic primitives with respect to their cryptographic definitions (for example, adaptive chosen-message security for signature schemes). The cryptographic definitions themselves are defined in terms of probability theory and complexity theory. Proving schemes secure with respect to such definitions is a complex endeavor, but one has much stronger guarantees than under the symbolic approach. In [18, 19], a formalization of the BPW model is presented that is a very general model that provides cryptographic guarantees (cryptographic soundness) with respect to the cryptographic approach. This model is formalized in Isabelle/HOL and, using this model, the security (authenticity) of the (corrected) Needham-Schroeder protocol is verified. This is the first such formalization, in logic, of this model and its first application to formal, machine-checked proofs. [20] presents a refinement of the game-based approach to security proofs and its implementation using the proof assistant Coq. Another tool following the cryptographic approach is CryptoVerif [12], which is an automatic protocol prover developed by Bruno Blanchet.

3 Framework for Protocol Certification

3.1 Objectives

As we mentioned in the last section, there are many formal methods that are effective for verifying (or falsifying) the security of cryptographic protocols. The problem today is not that there is a shortage of formal methods for cryptographic protocol analysis, but rather that there are too many! There is no consensus on which methods should be used and the scope of their effectiveness. Moreover, the relationships between the different methods is not yet well understood.

This situation is problematic for practitioners who design or use cryptographic protocols because they can neither select appropriate methods to verify their protocols nor have sufficient confidence in their results. Hence we propose a framework for protocol certification whose objectives are to establish means to provide defined levels of confidence (or assurance) concerning the security of the cryptographic protocols.

3.2 ISO/IEC 29128 verification of cryptographic protocols

ISO/IEC JTC 1/SC 27 has started in 2007 the project "Verification of cryptographic protocols (ISO/IEC 29128)" to provide a technical basis for the assessment of the security of cryptographic protocols. This project is on ballot to proceed at the Committee Draft (CD) stage at the time of writing this article and will become an International Standard by 2011 after several revisions and further ballots.

The current draft text of ISO/IEC 29128 does not specify precisely what proof methods or tools shall be used, but instead only specifies their properties. This encourages protocol designers to use state-of-the-art approaches for protocol verification in terms of models, methods, and tools. It also encourages tool designers to develop better tools.

The draft defines minimal requirements for specifying cryptographic protocols and different protocol assurance levels. To certify a cryptographic protocol, this standard requires a document that covers the following four aspects.

protocol specification: specification of the cryptographic protocol
adversarial model: specification of the adversarial model
security properties: specification of the objectives and security properties that the protocol should satisfy
self-assessment evidence: evidence that the specification of the cryptographic protocol in its adversarial model achieves its objectives and satisfies its security properties

The different protocol assurance levels lead to different requirements for these four aspects as shown in next subsection. The protocol designer prepares a document describing these four aspects of the protocol and provides it to the evaluator. The evaluator then checks whether these requirements are satisfied by the document in the sense defined for each protocol assurance level.

3.3 Cryptographic protocol assurance levels

Table 1 presents the three levels of our assurance requirements and the associated requirements for each of the four protocol aspects. These levels provide increasingly strong guarantees about the security of cryptographic protocols.

Table 1. Cryptographic protocol assurance levels

Protocol assurance levels	Protocol Assurance Level 1 (PAL1)	Protocol Assurance Level 2 (PAL2)	Protocol Assurance Level 3 (PAL3)
Protocol Specification	Semiformal description of protocol specification	Formal description of protocol specification in a tool-specific specification language, whose semantics is mathematically defined	
Adversarial model	Informal description of adversarial model	Formal description of adversarial model	
Security property	Informal description of security property	Formal description of security property	
Self-assessment evidence	Informal argument or mathematically formal paper-and-pencil proof that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model	Tool-aided bounded verification that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model	Tool-aided unbounded verification that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model

The difference between PAL1 and PAL2 is whether all aspects of the protocol description, such as the specification, security properties, and adversarial model, are formally described or not. If these are not sufficiently formal, a rigorous analysis is not possible and the designer cannot search for attacks or construct correctness proofs. At best, the designer can search for typical weaknesses and evaluate the protocol with respect to those attacks that she has thought of. Hence, PAL1 gives only minimal guarantees about the protocol's security. However, PAL1 may be sufficient for some closed network environment, such as a company intranet, lacking committed adversaries.

In contrast, in PAL2, the protocol designer provides a formal specification. Thus she can capture all traces consistent with the specification within some bound specified for the verification. Designers are typically poor at anticipating all possible (interleaved) traces and hence these traces will usually include complex ones, not considered in advance by the protocol designer. PAL2 generally gives reasonable guarantees that there does not exist any other successful adversary within some bound on the number of protocol sessions. We recommend PAL2 for open network environment such as the Internet.

The difference between PAL2 and PAL3 is whether or not the analysis (and hence the evidence presented) is for unbound verification. Verification in PAL2 is bounded and thus the designer cannot prove a protocol secure when complex attacks lie outside of the given bound. In contrast, PAL3 gives strong guarantees on that no successful (symbolic, Dolev-Yao) adversary exists, even allowing for unbounded numbers of sessions. With unbounded verification, a protocol designer can prove her protocol secure against all adversaries, even those willing to carry out complex and expensive attacks. PAL3 is effective for critical information systems, such as those providing social infrastructures or financial systems.

3.4 Discussion during the standardization process

Before the Committee Draft (CD) stage, ISO/IEC 29128 has been revised three times in the Working Draft (WD) stage. We have received various comments, which we have taken into account in the revisions. The following three points are the most important ones considered.

Neutrality to specific methods and tools. Since the state-of-the-art for protocol verification is progressing rapidly in terms of models, methods, and tools, this standard should not focus on specific methods or tools. Hence the draft standard provides only minimal requirements for specifying cryptographic protocols to keep them as general as possible.

Computational model. In very early stages of the standard, the highest assurance level required protocol verification in the computational model. However, because very few tools currently support this model and this approach requires both a very high degree of expertise and effort, cryptographic approaches based on the computational model (with unbounded verification) was included in PAL3 in the current draft. In the future, the use of the computational model might be defined at a higher level, such as PAL4, when verification tools are up to the task and usable by practitioners.

Paper-and-pencil proof. Both informal arguments and mathematically formal paper-and-pencil proofs are allowed under PAL1 in the current draft. Although formal proofs usually provides much more confidence than informal arguments, proofs by hand can be error-prone. Moreover, it is very difficult for protocol evaluators to confirm whether the proof is correct or not. Hence this standard requires mechanized proof for higher levels than PAL1.

3.5 Is our framework effective?

Currently, when a non-expert user uses a standardized cryptographic protocol, he cannot evaluate its security by himself. Instead he trusts the standardization body, which evaluated the protocol. In other words, the security of a cryptographic protocol is reduced to trusting the standardization body. As noted previously, this trust is not always well placed. In contrast, our proposed framework will provide the practitioner with trustworthy results based on sound, scientifically verifiable evidence.

So far, not all useful and practical protocols can be evaluated in the framework. One of the reason is the immaturity of tools. Although there are many tools as mentioned in Section 2, each tool has its own limitations. For example, few of the existing tools can effectively handle all of the different algebraic properties required to formalize the different cryptographic operators used in protocols. As a result, we may not be able to prove and certify the security of some protocols in the framework, even if they are actually secure because of lacking tool support. Another problem for the framework to be practical is the lack of experts. Each tool requires some expertise, but currently only a limited number of researchers have such expertise.

These problems could be improved by progress within the research community working on formal methods for protocol verification. They can also be improved by educating developers on existing formal methods. Note that, as we mentioned in Section 3.4, the proposed framework is open with respect to future progress in methods and tools.

4 Protocol Evaluation in a National Project: the CRYPTREC Case

4.1 Overview

As explained in the previous section, ISO/IEC 29128 is a framework for certifying cryptographic protocols using formal verification methods. To improve this framework and increase its usability, we must gather experience using it in actual evaluations. Afterwards we can revise the framework based on our experience. One of the authors is involved in a Japanese national project on the selection of cryptographic protocols within CRYPTREC.[5] Within this project, formal methods are being applied to verify selected entity-authentication protocols based on the certification framework described in Section 3. We plan to use the project results and experience gained there to evaluate and improve our framework.

The CRYPTREC project aims to evaluate and monitor the security of ciphers recommended for e-Government applications, as well as to study the establishment of evaluation criteria for cryptographic modules. In 2002, CRYPTREC produced an "e-Government Recommended Ciphers List"[21]. CRYPTREC is now conducting a renewal of this list. This includes recommending a list of entity-authentication protocols. In this renewal, CRYPTREC is asking for entity-authentication protocols that use cryptographic algorithms given in the

[5] CRYPTREC abbreviates "CRYPTography Research and Evaluation Committees". See http://www.cryptrec.go.jp/english/index.html

e-Government Recommended Ciphers List or that use cryptographic algorithms that have a security reduction to computationally difficult problems.

The submitted entity-authentication protocols should assure the correctness of the communication partners. In particular, the protocol designer can specify the protocol property as being mutual authentication or unilateral authentication. Examples of international standard protocols are:

– ISO/IEC 9798 series, which contain protocols based on symmetric encryption algorithms (9798-2), digital signature techniques (9798-3), cryptographic check functions (9798-4), zero knowledge techniques, (9798-5) and manual data transfer (9798-6),
– Kerberos and SASL (IETF), and
– One-time passwords.

In the evaluation by CRYPTREC, protocols are evaluated assuming that, if the cryptographic algorithms used are in the e-Government Recommended Ciphers List, then they are ideally secure. If other cryptographic algorithms are used, then the protocol is evaluated without their idealization. We describe the reasons for this below.

As indicated in Sections 2 and 3, there are many mature verification tools and different ways (corresponding to different assurance levels) that these tools can be used. When considering the maturity of current tools, CRYPTREC mainly considers verification without computational soundness. CRYPTREC already has recommended symmetric and asymmetric cryptographic algorithms, a hash function, and a pseudorandom generator. Thus, in the verification, we assume that the cryptographic algorithms in the list are ideal cryptographic algorithms. Hence, we do not require complicated computationally-sound proofs in this case. If the submitted protocol uses only cryptographic algorithms in the list, the efforts required of the protocol designer and the evaluator are reduced. Of course, this does not rule out entity-authentication protocols that use cryptographic algorithms not in the list. For example, we expect that many protocols will use a variant of the Diffie-Hellman protocol or the Fiat-Shamir heuristic. In this case, their security must be proven in a computationally-sound sense.

When submitting a proposal, the protocol designer gives an informal description of the proposed protocol, the desired properties, and the adversarial model to the evaluator who conducts the verification. The protocol designer also provides the evaluator with a formal description of the proposed protocol, the adversarial model, the result of executing the verification tool, and information on the tool itself. The protocol designer can use a formal verification tool that is publicly available or a proprietary (private) one. If the protocol designer uses a publicly available tool, he must provide the name of the tool and its version number. If he uses a proprietary verification tool, he must provide access to the verification tool itself as well as its specification.

The evaluator in CRYPTREC will investigate the correctness of the protocol description and the effectiveness of the verification tool. Then the evaluator performs verification using the same tool and compares the result with those submitted by the protocol designer.

4.2 Discussion

The main issue in the certification process is how to confirm the soundness of the verification tool. To obtain reliable verification results, the tool must not contain bugs that could lead to erroneous results. In practice, however, tools often do contain bugs. In some cases, different versions of the same tool may even produce different outputs. This is a serious problem for evaluators.

Solutions discussed within the CRYPTREC project are as follows.

- The evaluator collects information about reliable tools and their stable versions. Then the evaluator provides a list of them after obtaining consensus by experts. International consensus about the reliability of different tools is therefore needed.
- Alternatively, CRYPTREC provides a single standard verification tool so that the protocol designer and the evaluator can work using this (trusted) tool.

Eliminating bugs and producing stable versions of verification tools is quite important for certification. However, the current situation is insufficient for evaluating the correctness of the tools themselves. We expect to see methods available for evaluting the correctness of tools or their results in near future. We see three possibilities here. The first option is white-box testing of the tool by several experts. To carry out the tests, the protocol designer prepares

- a documented formal model underlying the tool,
- documentation on how the formalism is implemented in the tool,
- and the tool's source code.

The evaluator checks the soundness of the formalism from the description of the formal model, then checks if this is properly implemented by referencing the implementation document and the source code itself. This type of evaluation takes substantial time and efforts.

If time and effort are limited, a second option is to perform black-box tests on the tool. For cryptographic algorithms, the "test vectors" are a widely trusted tool for verification. To verify protocol tools, a test vector would consist of three parts: a test protocol, an adversarial model, and the expected verification result.

A final option is to use model-checking tools that produce proof scripts that can be independently checked. For example, the model-checking tool could generate a proof that can be checked using a standard theorem prover, e.g., one for higher-order logic like Isabelle/HOL. Recent research results suggest that this is a promising option.

Once the verification tool (or its output) is assured to be sound, the evaluator must still verify whether the formal description of the protocol specification correctly models the actual protocol. Moreover, it must be checked that security property correctly formalizes the actual security requirements and that the adversarial model is realistic for the protocol's intended application. Hence, even with this framework, the reasonableness of the security notion is finally assured by human experts.

5 Conclusion

In this paper, we have presented two activities related to the evaluation of cryptographic protocols, the ISO/IEC 29128 project, and the CRYPTREC project, which are being conducted in parallel.

ISO/IEC 29128 is a newly proposed international standard that uses formal methods to improve the security assurance of cryptographic protocols based on mathematically rigorous, machine-checkable, security proofs. Once a cryptographic protocol is certified with ISO/IEC 29128, in particular under its highest assurance level, the protocol is absolutely secure up to the assumption under which the security proofs are made and the soundness of the underlying verification tool. Thus, with this new standard, we should enjoy substantially higher levels of security than at present time.

The CRYPTREC project is a Japanese government project that evaluates the security of cryptographic algorithms and protocols as described in this paper. CRYPTREC is now planing to conduct a security evaluation for entity-authentication protocols using formal methods. We plan to combine the findings of the CRYPTREC project with the development of ISO/IEC 29128. In this way we hope to further improve ISO/IEC 29128 and its practicality.

References

1. Blanchet, B.: From secrecy to authenticity in security protocols. In: SAS '02: Proceedings of the 9th International Symposium on Static Analysis, London, UK, Springer-Verlag (2002) 342–359
2. Kemmerer, R.: Using formal methods to analyze encryption protocols. IEEE Journal of Selected Areas in Communication **7**(2) (1989) 448–457
3. Meadows, C.: The NRL protocol analyzer: An overview. Journal of Logic Programming **19** (1994)
4. Hoare, C.: Communicating sequential processes. CACM **21** (1978) 666–677
5. Hoare, C.A.: Communicating Sequential Processes. Prentice-Hall, Englewood Cliffs, NJ (1995)
6. Basin, D.: Lazy infinite-state analysis of security protocols. In Baumgart, R., ed.: Secure Networking — CQRE (Secure)'99. LNCS 1740, Springer-Verlag (1999) 30–42
7. Basin, D., Mödersheim, S., Viganò, L.: OFMC: A symbolic model checker for security protocols. International Journal of Information Security **4**(3) (2005) 181–208
8. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuellar, J., Drielsma, P.H., Heám, P.C., Kouchnarenko, O., Mantovani, J., Mödersheim, S., von Oheimb, D., Rusinowitch, M., Santiago, J., Turuani, M., Viganò, L., Vigneron, L.: The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In: Proceedings of CAV'2005. LNCS 3576. Springer-Verlag (2005) 281–285
9. Abadi, M., Blanchet, B.: Analyzing Security Protocols with Secrecy Types and Logic Programs. Journal of the ACM **52**(1) (January 2005) 102–146
10. Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In: Proceedings of CSFW'01, IEEE Computer Society Press (2001) 82–96

11. Blanchet, B.: A computationally sound mechanized prover for security protocols. In: IEEE Symposium on Security and Privacy, Oakland, California (May 2006) 140–154
12. Blanchet, B.: A computationally sound automatic prover for cryptographic protocols. In: Workshop on the link between formal and computational models, Paris, France (June 2005)
13. Cremers, C.: Scyther — Semantics and Verification of Security Protocols. PhD thesis, University of Eindhoven (2006)
14. Paulson, L.C.: The inductive approach to verifying cryptographic protocols. Journal of Computer Security **6** (1998) 85–128
15. Durgin, N., Lincoln, P.D., Mitchell, J.C., Scedrov, A.: Undecidability of Bounded Security Protocols. In: Proceedings of the FLOC'99 Workshop on Formal Methods and Security Protocols (FMSP'99). (1999)
16. Boichut, Y., Heam, P.C., Kouchnarenko, O., Oehl, F.: Improvements on the Genet and Klay Technique to Automatically Verify Security Protocols. In: Automated Verification of Infinite States Systems (AVIS'04). ENTCS (2004)
17. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences **28** (1984) 270–299
18. Sprenger, C., Backes, M., Basin, D., Pfitzmann, B., Waidner, M.: Cryptographically sound theorem proving. In: 19th IEEE Computer Security Foundations Workshop, Venice, Italy, IEEE Computer Society (July 2006) 153–166
19. Sprenger, C., Basin, D.: Cryptographically-sound protocol-model abstractions. In: Computer Security Foundations (CSF 08), Los Alamitos, CA, USA, IEEE Computer Society (2008) 115–129
20. Nowak, D.: A framework for game-based security proofs. In Qing, S., Imai, H., Wang, G., eds.: ICICS. Volume 4861 of Lecture Notes in Computer Science., Springer (2007) 319–333
21. CRYPTREC: e-government recommended ciphers list. `http://www.cryptrec.go.jp/english/images/cryptrec_01en.pdf` (2003)