Bruce Christianson
Bruno Crispo
James A. Malcolm
Michael Roe (Eds.)

LNCS 3364

Security Protocols

11th International Workshop
Cambridge, UK, April 2003
Revised Selected Papers

Springer

Bruce Christianson   Bruno Crispo
James A. Malcolm   Michael Roe (Eds.)

Security Protocols

11th International Workshop
Cambridge, UK, April 2-4, 2003
Revised Selected Papers

Springer

Volume Editors

Bruce Christianson
James A. Malcolm
University of Hertfordshire
Computer Science Department
Hatfield AL10 9AB, UK
E-mail: {b.christianson,J.A.Malcolm}@herts.ac.uk

Bruno Crispo
Vrije Universiteit
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
E-mail: crispo@cs.vu.nl

Michael Roe
Microsoft Research Ltd
7 J.J. Thomson Avenue, Cambridge CB3 0FB, UK
E-mail: mroe@mircosoft.com

Library of Congress Control Number: 2005931713

CR Subject Classification (1998): E.3, F.2.1-2, C.2, K.6.5, J.1, K.4.1, D.4.6

ISSN 0302-9743
ISBN-10 3-540-28389-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-28389-8 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11542322 06/3142 5 4 3 2 1 0

Preface

Greetings. These are the proceedings of the 11th in our series of International Workshops on Security Protocols. Our theme this time was "Where have all the Protocols gone?" Once upon a time security protocols lived mainly in the network and transport layers. Now they increasingly hide in applications, or in specialised hardware. Does this trend lead to better security architectures, or is it an indication that we are addressing the wrong problems?

The intention of the workshops is to provide a forum where incompletely worked out ideas can stimulate discussion, open up new lines of investigation, and suggest more problems. The position papers published here have been revised by the authors in the light of their participation in the workshop. In addition, we publish edited transcripts of some of the discussions, to give our readers access to some of the roads ahead not (yet) taken. We hope that these revised position papers and edited transcripts will give you at least one interesting idea of your own to explore. Please do write and tell us what it was.

Our purpose in publishing these proceedings is to produce a conceptual map which will be of enduring interest, rather than to be merely topical. This is perhaps just as well, given the delay in production. This year we moved to new computer-based recording technology, and of course it failed completely. Fortunately various domestic recorders had been smuggled into the venue, but picking the signal bits out of the noise has taken a long time, and we have had to insert more than the usual number of epicycles to make the discussions come out right.

Our thanks to Sidney Sussex College Cambridge for the use of their facilities, to Lori Klimaszewska of the University of Cambridge Computing Service for the even worse than usual task of transcribing the audio tapes (in which the reverse use of "two rings" provided a test for creative intelligence) and to Johanna Hunt at the University of Hertfordshire for helping us with the Ptolemeic editing of the results.

Lent 2005

Bruce Christianson
Bruno Crispo
James Malcolm
Michael Roe

Previous Proceedings in This Series

The proceedings of previous International Workshops on Security Protocols have also been published by Springer as Lecture Notes in Computer Science, and are occasionally referred to in the text:

10th Workshop (2002), LNCS 2845, ISBN 3-540-20830-5
9th Workshop (2001), LNCS 2467, ISBN 3-540-44263-4
8th Workshop (2000), LNCS 2133, ISBN 3-540-42566-7
7th Workshop (1999), LNCS 1796, ISBN 3-540-67381-4
6th Workshop (1998), LNCS 1550, ISBN 3-540-65663-4
5th Workshop (1997), LNCS 1361, ISBN 3-540-64040-1
4th Workshop (1996), LNCS 1189, ISBN 3-540-63494-5

Lecture Notes in Computer Science 3364

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Lecture Notes in Computer Science

For information about Vols. 1–3573

please contact your bookseller or Springer

Vol. 3703: F. Fages, S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning. VIII, 163 pages. 2005.

Vol. 3697: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005, Part II. XXXII, 1045 pages. 2005.

Vol. 3696: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Biological Inspirations - ICANN 2005, Part I. XXXI, 703 pages. 2005.

Vol. 3687: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Image Analysis, Part II. XXV, 809 pages. 2005.

Vol. 3686: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Data Mining, Part I. XXVI, 689 pages. 2005.

Vol. 3674: W. Jonker, M. Petković (Eds.), Secure Data Management. X, 241 pages. 2005.

Vol. 3672: C. Hankin, I. Siveroni (Eds.), Static Analysis. X, 369 pages. 2005.

Vol. 3671: S. Bressan, S. Ceri, E. Hunt, Z.G. Ives, Z. Bellahsène, M. Rys, R. Unland (Eds.), Database and XML Technologies. X, 239 pages. 2005.

Vol. 3670: M. Bravetti, L. Kloul, G. Zavattaro (Eds.), Formal Techniques for Computer Systems and Business Processes. XIII, 349 pages. 2005.

Vol. 3665: K. S. Candan, A. Celentano (Eds.), Advances in Multimedia Information Systems. X, 221 pages. 2005.

Vol. 3664: C. Türker, M. Agosti, H.-J. Schek (Eds.), Peer-to-Peer, Grid, and Service-Orientation in Digital Library Architectures. X, 261 pages. 2005.

Vol. 3663: W.G. Kropatsch, R. Sablatnig, A. Hanbury (Eds.), Pattern Recognition. XIV, 512 pages. 2005.

Vol. 3662: C. Baral, G. Greco, N. Leone, G. Terracina (Eds.), Logic Programming and Nonmonotonic Reasoning. XIII, 454 pages. 2005. (Subseries LNAI).

Vol. 3660: M. Beigl, S. Intille, J. Rekimoto, H. Tokuda (Eds.), UbiComp 2005: Ubiquitous Computing. XVII, 394 pages. 2005.

Vol. 3659: J.R. Rao, B. Sunar (Eds.), Cryptographic Hardware and Embedded Systems – CHES 2005. XIV, 458 pages. 2005.

Vol. 3658: V. Matoušek, P. Mautner, T. Pavelka (Eds.), Text, Speech and Dialogue. XV, 460 pages. 2005. (Subseries LNAI).

Vol. 3654: S. Jajodia, D. Wijesekera (Eds.), Data and Applications Security XIX. X, 353 pages. 2005.

Vol. 3653: M. Abadi, L.d. Alfaro (Eds.), CONCUR 2005 – Concurrency Theory. XIV, 578 pages. 2005.

Vol. 3649: W.M.P. van der Aalst, B. Benatallah, F. Casati, F. Curbera (Eds.), Business Process Management. XII, 472 pages. 2005.

Vol. 3648: J.C. Cunha, P.D. Medeiros (Eds.), Euro-Par 2005 Parallel Processing. XXXVI, 1299 pages. 2005.

Vol. 3646: A. F. Famili, J.N. Kok, J.M. Peña, A. Siebes, A. Feelders (Eds.), Advances in Intelligent Data Analysis VI. XIV, 522 pages. 2005.

Vol. 3645: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part II. XIII, 1010 pages. 2005.

Vol. 3644: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part I. XXVII, 1101 pages. 2005.

Vol. 3642: D. Ślezak, J. Yao, J.F. Peters, W. Ziarko, X. Hu (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part II. XXIII, 738 pages. 2005. (Subseries LNAI).

Vol. 3641: D. Ślezak, G. Wang, M. Szczuka, I. Düntsch, Y. Yao (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part I. XXIV, 742 pages. 2005. (Subseries LNAI).

Vol. 3639: P. Godefroid (Ed.), Model Checking Software. XI, 289 pages. 2005.

Vol. 3638: A. Butz, B. Fisher, A. Krüger, P. Olivier (Eds.), Smart Graphics. XI, 269 pages. 2005.

Vol. 3637: J. M. Moreno, J. Madrenas, J. Cosp (Eds.), Evolvable Systems: From Biology to Hardware. XI, 227 pages. 2005.

Vol. 3636: M.J. Blesa, C. Blum, A. Roli, M. Sampels (Eds.), Hybrid Metaheuristics. XII, 155 pages. 2005.

Vol. 3634: L. Ong (Ed.), Computer Science Logic. XI, 567 pages. 2005.

Vol. 3633: C. Bauzer Medeiros, M. Egenhofer, E. Bertino (Eds.), Advances in Spatial and Temporal Databases. XIII, 433 pages. 2005.

Vol. 3632: R. Nieuwenhuis (Ed.), Automated Deduction – CADE-20. XIII, 459 pages. 2005. (Subseries LNAI).

Vol. 3630: M.S. Capcarrere, A.A. Freitas, P.J. Bentley, C.G. Johnson, J. Timmis (Eds.), Advances in Artificial Life. XIX, 949 pages. 2005. (Subseries LNAI).

Vol. 3629: J.L. Fiadeiro, N. Harman, M. Roggenbach, J. Rutten (Eds.), Algebra and Coalgebra in Computer Science. XI, 457 pages. 2005.

Vol. 3628: T. Gschwind, U. Aßmann, O. Nierstrasz (Eds.), Software Composition. X, 199 pages. 2005.

Vol. 3627: C. Jacob, M.L. Pilat, P.J. Bentley, J. Timmis (Eds.), Artificial Immune Systems. XII, 500 pages. 2005.

Vol. 3626: B. Ganter, G. Stumme, R. Wille (Eds.), Formal Concept Analysis. X, 349 pages. 2005. (Subseries LNAI).

Vol. 3625: S. Kramer, B. Pfahringer (Eds.), Inductive Logic Programming. XIII, 427 pages. 2005. (Subseries LNAI).

Vol. 3624: C. Chekuri, K. Jansen, J.D.P. Rolim, L. Trevisan (Eds.), Approximation, Randomization and Combinatorial Optimization. XI, 495 pages. 2005.

Vol. 3623: M. Liśkiewicz, R. Reischuk (Eds.), Fundamentals of Computation Theory. XV, 576 pages. 2005.

Vol. 3621: V. Shoup (Ed.), Advances in Cryptology – CRYPTO 2005. XI, 568 pages. 2005.

Vol. 3620: H. Muñoz-Avila, F. Ricci (Eds.), Case-Based Reasoning Research and Development. XV, 654 pages. 2005. (Subseries LNAI).

Vol. 3619: X. Lu, W. Zhao (Eds.), Networking and Mobile Computing. XXIV, 1299 pages. 2005.

Vol. 3618: J. Jedrzejowicz, A. Szepietowski (Eds.), Mathematical Foundations of Computer Science 2005. XVI, 814 pages. 2005.

Vol. 3617: F. Roli, S. Vitulano (Eds.), Image Analysis and Processing – ICIAP 2005. XXIV, 1219 pages. 2005.

Vol. 3615: B. Ludäscher, L. Raschid (Eds.), Data Integration in the Life Sciences. XII, 344 pages. 2005. (Subseries LNBI).

Vol. 3614: L. Wang, Y. Jin (Eds.), Fuzzy Systems and Knowledge Discovery, Part II. XLI, 1314 pages. 2005. (Subseries LNAI).

Vol. 3613: L. Wang, Y. Jin (Eds.), Fuzzy Systems and Knowledge Discovery, Part I. XLI, 1334 pages. 2005. (Subseries LNAI).

Vol. 3612: L. Wang, K. Chen, Y. S. Ong (Eds.), Advances in Natural Computation, Part III. LXI, 1326 pages. 2005.

Vol. 3611: L. Wang, K. Chen, Y. S. Ong (Eds.), Advances in Natural Computation, Part II. LXI, 1292 pages. 2005.

Vol. 3610: L. Wang, K. Chen, Y. S. Ong (Eds.), Advances in Natural Computation, Part I. LXI, 1302 pages. 2005.

Vol. 3608: F. Dehne, A. López-Ortiz, J.-R. Sack (Eds.), Algorithms and Data Structures. XIV, 446 pages. 2005.

Vol. 3607: J.-D. Zucker, L. Saitta (Eds.), Abstraction, Reformulation and Approximation. XII, 376 pages. 2005. (Subseries LNAI).

Vol. 3606: V. Malyshkin (Ed.), Parallel Computing Technologies. XII, 470 pages. 2005.

Vol. 3604: R. Martin, H. Bez, M. Sabin (Eds.), Mathematics of Surfaces XI. IX, 473 pages. 2005.

Vol. 3603: J. Hurd, T. Melham (Eds.), Theorem Proving in Higher Order Logics. IX, 409 pages. 2005.

Vol. 3602: R. Eigenmann, Z. Li, S.P. Midkiff (Eds.), Languages and Compilers for High Performance Computing. IX, 486 pages. 2005.

Vol. 3599: U. Aßmann, M. Aksit, A. Rensink (Eds.), Model Driven Architecture. X, 235 pages. 2005.

Vol. 3598: H. Murakami, H. Nakashima, H. Tokuda, M. Yasumura, Ubiquitous Computing Systems. XIII, 275 pages. 2005.

Vol. 3597: S. Shimojo, S. Ichii, T.W. Ling, K.-H. Song (Eds.), Web and Communication Technologies and Internet-Related Social Issues - HSI 2005. XIX, 368 pages. 2005.

Vol. 3596: F. Dau, M.-L. Mugnier, G. Stumme (Eds.), Conceptual Structures: Common Semantics for Sharing Knowledge. XI, 467 pages. 2005. (Subseries LNAI).

Vol. 3595: L. Wang (Ed.), Computing and Combinatorics. XVI, 995 pages. 2005.

Vol. 3594: J.C. Setubal, S. Verjovski-Almeida (Eds.), Advances in Bioinformatics and Computational Biology. XIV, 258 pages. 2005. (Subseries LNBI).

Vol. 3593: V. Mařík, R. W. Brennan, M. Pěchouček (Eds.), Holonic and Multi-Agent Systems for Manufacturing. XI, 269 pages. 2005. (Subseries LNAI).

Vol. 3592: S. Katsikas, J. Lopez, G. Pernul (Eds.), Trust, Privacy and Security in Digital Business. XII, 332 pages. 2005.

Vol. 3591: M.A. Wimmer, R. Traunmüller, Å. Grönlund, K.V. Andersen (Eds.), Electronic Government. XIII, 317 pages. 2005.

Vol. 3590: K. Bauknecht, B. Pröll, H. Werthner (Eds.), E-Commerce and Web Technologies. XIV, 380 pages. 2005.

Vol. 3589: A M. Tjoa, J. Trujillo (Eds.), Data Warehousing and Knowledge Discovery. XVI, 538 pages. 2005.

Vol. 3588: K.V. Andersen, J. Debenham, R. Wagner (Eds.), Database and Expert Systems Applications. XX, 955 pages. 2005.

Vol. 3587: P. Perner, A. Imiya (Eds.), Machine Learning and Data Mining in Pattern Recognition. XVII, 695 pages. 2005. (Subseries LNAI).

Vol. 3586: A.P. Black (Ed.), ECOOP 2005 - Object-Oriented Programming. XVII, 631 pages. 2005.

Vol. 3584: X. Li, S. Wang, Z.Y. Dong (Eds.), Advanced Data Mining and Applications. XIX, 835 pages. 2005. (Subseries LNAI).

Vol. 3583: R.W. H. Lau, Q. Li, R. Cheung, W. Liu (Eds.), Advances in Web-Based Learning – ICWL 2005. XIV, 420 pages. 2005.

Vol. 3582: J. Fitzgerald, I.J. Hayes, A. Tarlecki (Eds.), FM 2005: Formal Methods. XIV, 558 pages. 2005.

Vol. 3581: S. Miksch, J. Hunter, E. Keravnou (Eds.), Artificial Intelligence in Medicine. XVII, 547 pages. 2005. (Subseries LNAI).

Vol. 3580: L. Caires, G.F. Italiano, L. Monteiro, C. Palamidessi, M. Yung (Eds.), Automata, Languages and Programming. XXV, 1477 pages. 2005.

Vol. 3579: D. Lowe, M. Gaedke (Eds.), Web Engineering. XXII, 633 pages. 2005.

Vol. 3578: M. Gallagher, J. Hogan, F. Maire (Eds.), Intelligent Data Engineering and Automated Learning - IDEAL 2005. XVI, 599 pages. 2005.

Vol. 3577: R. Falcone, S. Barber, J. Sabater-Mir, M.P. Singh (Eds.), Trusting Agents for Trusting Electronic Societies. VIII, 235 pages. 2005. (Subseries LNAI).

Vol. 3576: K. Etessami, S.K. Rajamani (Eds.), Computer Aided Verification. XV, 564 pages. 2005.

Vol. 3575: S. Wermter, G. Palm, M. Elshaw (Eds.), Biomimetic Neural Learning for Intelligent Robots. IX, 383 pages. 2005. (Subseries LNAI).

Vol. 3574: C. Boyd, J.M. González Nieto (Eds.), Information Security and Privacy. XIII, 586 pages. 2005.

¥490.88元

Table of Contents

* Speakers.

Where Have All the Protocols Gone?

Bruce Christianson

University of Hertfordshire, UK

There was a time when security protocols lived mainly in the network and transport layers. Where are they now?

Some have moved downstairs, towards the physical layer. What used to be a wide-area authentication or session establishment protocol is now a very local interaction with a trusted device, such as a tamper-evident smartcard, or a biometric token.

Indeed, in some cases a piece of mobile hardware has actually replaced altogether the security protocol that we used to find. Now in the strict sense, there is still a security protocol here: we use a set of rules to construct an artefact which will then be moved into a different context and interpreted in accordance with a shared set of conventions. But the individual protocol run no longer involves the same kind of electronic message-passing that we used to see or rather, as Marshall McLuhan would have said, the medium *is* now the message.

Other security protocols have moved upwards, to the application layer. The increasing deployment of Trusted Third Parties or TTPs is actually encouraging this trend. The part of the protocol that is visible from the point of view of the network, or of the system infrastructure, is now just the first, not so interesting, half of the protocol.

The quantum computing people regard entanglement as a resource. They have protocols for creating entanglement, but that's not what is interesting. The interesting question is what is done with the entanglement once it's created. How does it get used up?

The security protocol that we see in action across the network is often being used to create a cryptographic entanglement between a number of geographically separated entities. But again, the interesting thing is not how this entanglement got there, it is what happens afterwards[1]. What are these entities going to do with that entangled cryptographic resource? What happens next to the fresh session key after it gets handed off to the application? How does it get used up?

We don't see that backend to the protocol anymore. It's usually happening after the point where the analysis of the protocol has stopped, and it now tends to happen where we can't get at it to analyse it anyway. In many cases the really interesting part of the security protocol has moved all the way upstairs to the penthouse, above the API, where it is effectively hidden.

Still other security protocols have moved to a completely different level of abstraction. This is particularly the case for the protocols which used to be

[1] For more on this point, see Roger Needham's keynote address "Mobile Computing vs Immobile Security", LNCS 2467, 1–3.

B. Christianson et al. (Eds.): Security Protocols 2003, LNCS 3364, pp. 1–2, 2005.

used to enforce a security policy in a foreign domain. Many of these have now been replaced with service level agreeements, or SLAs[2]. Now an SLA is also a protocol, but at a much more abstract level. Just as with the biometric tokens, the security protocol itself now no longer involves the traditional type of explicit instance-based message passing. So nor do the threats.

Even the protocols that still seem to look much the way they always did are frequently not doing what they used to do. Partly this is because of the natural tendency to use security protocols for things other than those for which they were originally designed[3]. It's so hard to get a security protocol right, it always seems a shame not to reuse it. Although this isn't often a sensible thing to do.

But this is only part of the reason. The context in which security protocols are operating has also changed dramatically[4]. For example, twenty years ago it was an axiom that the primary purpose of security was to protect the system from the user. Most interesting failure stories now involve some kind of failure to protect the user from the system. I think this is actually the emerging security requirement which is least well met today.

To take another example, a lot of early security protocols were deliberately designed so as not to assume the existence of persistent shared state. Twenty years ago, constructing global state was a terribly difficult technical problem[5]. It was actually less bother to go to extreme lengths to construct protocols in such a way that they didn't rely upon any stable kind of global state. Nowadays the ease with which we can construct reliable, global state is very frightening. It makes various types of privacy invasion a real threat at the purely technical level, leaving aside any personal or political security dimensions[6].

And so finally, a lot of old security protocols have gone over to the dark side of the force, and are now effectively working for the bad guys. Many emerging security requirements are therefore, largely direct consequences of the success with which we've addressed the security problems of ten or twenty years ago.

Modern attacks tend to be built upon, and to exploit the security features of, the very infrastructure with which we ourselves have provided the attacker. Consequently we now need to turn a lot of our threat models inside out. Many countermeasures will become threats, and hopefully vice versa.

Is this trend for security protocols to migrate (and mutate) an opportunity which could lead to better security architectures, or is it an indication that we are addressing the wrong problems?

[2] Federated Identity Management is an example of this process, see B. Pfitzmann, these proceedings.

[3] See Roger's keynote address "Security Protocols and the Swiss Army Knife", LNCS 2133, 1–4.

[4] See Roger's keynote address "The Changing Environment for Security Protocols", LNCS 1796, 1–5.

[5] Conversely, constructing a global trust infrastructure was believed to be a problem which would soon be solved. Now we have learned to do without: see Roger's keynote address "Discerning the Protocol Participants", LNCS 2845, 1–3.

[6] For examples see the concluding discussion in LNCS 2467, 229–238.

A Protocol's Life After Attacks...

Giampaolo Bella[1,2], Stefano Bistarelli[3,4], and Fabio Massacci[5]

[1] Computer Laboratory, University of Cambridge, UK
[2] Dip. di Matematica e Informatica, Università di Catania, Italy
giamp@dmi.unict.it
[3] Dip. di Scienze, Università "G. D'annunzio" di Chieti-Pescara, Italy
bista@sci.unich.it
[4] Istituto di Informatica e Telematica, CNR, Pisa, Italy
stefano.bistarelli@iit.cnr.it
[5] Dip. di Informatica e TLC, Università di Trento, Italy
massacci@ing.unitn.it

Abstract. In the analysis of security protocols, it is customary to stop as soon as we find an attack. Tons of ink can be spilled on whether an "attack" is really an attack, but it goes without saying that there is no life after that, hence no interest in continuing the analysis. If the protocol is broken, then we ought to fix it.

Yet, fixing things is expensive and other measures may be more effective. In the physical world, most ATM safes would not resist heavy shelling with anti-tank bazookas, but banks don't worry about that. The attack will be noisy enough that cops will come within seconds from its start. To secure ourselves, we rely on a mixture of measures including the protection from attacks but also countermeasures after detection.

In the light of these considerations, the following question becomes of interest: *what can happen after an attack?* Does the villain leave enough traces that we can retaliate it on-the-fly? Or, if we can't or won't, does a subsequent forensic analysis allow us to discover who did it (and send the cops behind him)? If even this is impossible, can we discover that we have been hacked by looking at the logs?

To address these issues, we introduce the notions of *retaliation*, *detection*, and *suspicion*, which can be applied after an attack. These properties introduce more sophisticated formal relations between traces of actions, which go beyond the simple existentials that formal methods have made us used to.

These concepts should allow for a more comprehensive evaluation of security protocols. A protocol may well be vulnerable to an attack, but if we can retaliate afterwards, maybe fixing it isn't that necessary: the concrete possibilities of retaliation or detection may be enough to convince potential hackers to refrain from mounting the attack.

1 Introduction and Motivations

What is a security protocol, if we set technology aside? It is just a social behavior that principals of a distributed system must follow to obtain some important collective benefits. For the good guys, we just set up clear, understandable,

B. Christianson et al. (Eds.): Security Protocols 2003, LNCS 3364, pp. 3–10, 2005.

and acceptable rules describing this behaviour: execute the security protocol correctly, namely by the book. Because they are good guys, they will conform to the rules, and behave as we wanted. The bad guys, by definition, will not conform to the rules and execute the protocol incorrectly, namely arbitrarily.

Classical research in distributed systems and security starts from the need to counter the disruptive behavior of the nasty ones. In classical distributed algorithms, the main focus has been to design the protocol so that if the good guys outnumber the bad ones, the collective benefits will be achieved, no matter what the bad guys do (alternatively, prove that the good guys are doomed, no matter how many they are and how smart they are [4]). The security standpoint is to find a design such that, no matter what the bad guys do and no matter how many they are, they can prevent the good guys from achieving the desired collective benefits.

Each time an attack is published, the implied corollary is that the security experts have failed: the protocol has flaws, the good guys cannot achieve their ultimate goal, and we should go back to the drawing board. Yet, before abandoning the protocol, it is worth looking at what is left after the attack. This can lead us to more comprehensive evaluation of security protocols even if it requires continuation of a protocol analysis after an attack is found.

1.1 Our Contribution

It has never been considered whether it is at all possible to threaten the bad guys in case they execute the protocol incorrectly. In the real world, we impose a virtuous behavior on people by not letting them sin first (the classical security approach), and by making them repent of their sins ever since (sending them to jail). They would therefore weigh up the benefits of an incorrect execution with the consequent threats, and might choose to execute the protocol correctly if the threats were heavier.

Let's consider Lowe's example in his paper on Needham-Schroeder [6]. After the end of the attack, the bad guy asks for a transfer of money. Would he steal 1.000 Euro if the threat that 2.000 Euro could be stolen to him on the next day were significant? Would he transfer the money if the chances of being caught were significant?

In this paper, we propose a number of notions for the analysis of protocols beyond attacks: we introduce the notions of *retaliation*, *detection*, and *suspicion*. For instance, in a peer-2-peer environment or in network games, *direct retaliation* can be an effective threat to force the bad guys to play by the rules.

In the next section we show how Lowe's attack on Needham-Schroeder can be retaliated [1]. Then, we introduce the notion of retaliation, followed by those of detection and suspicion.

2 Retaliation in Needham-Schroeder

Let's start from a classical case: the (in)famous Needham-Schroeder public key protocol represented in Figure 1. We use the classical notation for security protocols [2]:

- keys are denoted by K, possibly extended with subscripts expressing the principals knowing them; the $^{-1}$ superscript expresses the inverse of a key;
- nonces are denoted by N;
- concatenation is denoted by a comma;
- encryption is denoted by a pair of curly braces with the key as a subscript; the type of the key determines the type of encryption.

$$\begin{array}{ll} 1. & A \rightarrow B : \{\!|Na, A|\!\}_{Kb} \\ 2. & B \rightarrow A : \{\!|Na, Nb|\!\}_{Ka} \\ 3. & A \rightarrow B : \{\!|Nb|\!\}_{Kb} \end{array}$$

Fig. 1. The asymmetric Needham-Schroeder protocol

The goal of the protocol is *authentication*: at completion of a session initiated by A with B, A should get evidence to have communicated with B and, likewise, B should get evidence to have communicated with A. Assuming that encryption is perfect and that the nonces are truly random, authentication is achieved here by confidentiality of the nonces. Indeed, upon reception of Na inside message 2, A would conclude that she is interacting with B, the only principal who could retrieve Na from message 1. In the same fashion, when B receives Nb inside message 3, he would conclude that A was at the other end of the network because Nb must have been obtained from message 2, and no-one but A could perform this operation.

Lowe discovers [6] that the protocol suffers the "attack" described in Figure 2, whereby a malicious principal C masquerades as a principal A with a principal B, after A initiated a session with C. The attack, which sees C interleave two sessions, indicates failure of the authentication of A with B which follows from failure of the confidentiality of Nb.

$$\begin{array}{ll} 1. & A \rightarrow C : \{\!|Na, A|\!\}_{Kc} \\ 1'. & C \rightarrow B : \{\!|Na, A|\!\}_{Kb} \\ 2'. & B \rightarrow A : \{\!|Na, Nb|\!\}_{Ka} \\ 2. & C \rightarrow A : \{\!|Na, Nb|\!\}_{Ka} \\ 3. & A \rightarrow C : \{\!|Nb|\!\}_{Kc} \\ 3'. & C \rightarrow B : \{\!|Nb|\!\}_{Kb} \end{array}$$

Fig. 2. Lowe's attack to the Needham-Schroeder Protocol

Let's examine this protocol after the attack took place:

- B is the subject of the attack — we call him the *good* agent;
- C is the *bad* guy;

- A is just playing by the rules — we call him the *ugly* participant;
- the trace T of the protocol as in Figure 2 describes an attack

The predicate $\mathit{Attack}(T, \mathcal{G} := \{B\}, \mathcal{B} := \{C\}, \mathcal{U} := \{A\})$ can represent the previous fact with the meaning:

- B executes a run of the protocol apparently with A (in fact he receives/sends the following messages:$[\{\!|Na, A|\!\}_{Kb}, \{\!|Na, Nb|\!\}_{Ka}, \{\!|Nb|\!\}_{Kb}] \in T$);
- C knows the nonce Nb, which was meant to be known to A and B only by receiving the message $\{\!|Nb|\!\}_{Kc} \in T$ in step 3 (so he can complete the run with B and perform the attack).
- A plays according to the rules and is just a spectator in the battle between B and C.

By using the authentication attack if B is a bank for example, C can steal money from A's account as reported by Lowe [6]:

$$4. \quad C \rightarrow B : \{\!|Na, Nb, \text{“Transfer £1000 from } A\text{'s account to } C\text{'s”}|\!\}_{Kb}$$

The bank B would honour the request believing it came from the account holder A.

Notice however, that the same predicate *Attack* also holds if instantiated in a different way [1]. In fact, we have also $\mathit{Attack}(T, \mathcal{G} := \{A\}, \mathcal{B} := \{B\}, \mathcal{U} := \{C\})$:

- A executes a run of the protocol apparently with C (in fact he receives/sends the following messages:$[\{\!|Na, A|\!\}_{Kc}, \{\!|Na, Nb|\!\}_{Ka}, \{\!|Nb|\!\}_{Kc}] \in T$);
- B knows the nonce Na, which was meant to be known to A and C only, by receiving the message $\{\!|Na, A|\!\}_{Kb} \in \mathcal{T}$ in step $1'$ (so he can send in step 2 the message $\{\!|Na, Nb|\!\}_{Ka}$ also without having C sending it to A).
- C is in this run just playing according to the rules.

In this case B is the bad guy. As C did previously, B can equally decide to illegally exploit his knowledge of Na. If also A is a bank, B can rob the robber as follows:

$$2''. \quad B \rightarrow A : \{\!|Na, Nb|\!\}_{Ka}$$
$$4'' \quad B \rightarrow A : \{\!|Na, Nb, \text{“Transfer £2000 from } C\text{'s account to } B\text{'s”}|\!\}_{Ka}$$

The bank A would honour the request believing it came from the account holder C. After we found the attack $\mathit{Attack}(T, \mathcal{G} := \{B\}, \mathcal{B} := \{C\}, \mathcal{U} := \{A\})$ in the trace T, we could have continued the analysis and find out that the bad guy could be easily punished, as in the example given above.

3 Vendetta...

To introduce the notion of retaliation we must identify the role played by each principal in the protocol execution. We are not interested in notions such as responder or initiator; rather, we intend to provide a behavioural characterization.