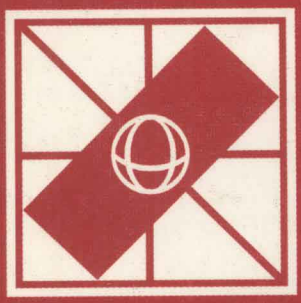
David de Frutos-Escrig
Manuel Núñez (Eds.)

Formal Techniques for Networked and Distributed Systems – FORTE 2004

LNCS 3235

24th IFIP WG 6.1 International Conference
Madrid, Spain, September 2004
Proceedings



IFIP – WG6.1

Springer

David de Frutos-Escrig Manuel Núñez (Eds.)

Formal Techniques for Networked and Distributed Systems – FORTE 2004

24th IFIP WG 6.1 International Conference
Madrid, Spain, September 27-30, 2004
Proceedings

Springer

Volume Editors

David de Frutos-Escrig
Manuel Núñez
Universidad Complutense de Madrid
Dept. Sistemas Informáticos y Programación
Madrid 28040, Spain
E-mail:{defrutos/mn}@sip.ucm.es

Library of Congress Control Number: Applied for

CR Subject Classification (1998): C.2.4, D.2.2, C.2, D.2.4-5, D.2, F.3, D.4

ISSN 0302-9743
ISBN 3-540-23252-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by DA-TeX Gerd Blumenstein
Printed on acid-free paper SPIN: 11321293 06/3142 5 4 3 2 1 0

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

Editorial Board

Lecture Notes in Computer Science

For information about Vols. 1–3162

please contact your bookseller or Springer

Vol. 3274: R. Guerraoui (Ed.), Distributed Computing. XIII, 465 pages. 2004.

Vol. 3273: T. Baar, A. Strohmeier, A. Moreira, S.J. Mellor (Eds.), <<UML>> 2004 - The Unified Modelling Language. XIII, 449 pages. 2004.

Vol. 3271: J. Vicente, D. Hutchison (Eds.), Management of Multimedia Networks and Services. XIII, 335 pages. 2004.

Vol. 3270: M. Jeckle, R. Kowalczyk, P. Braun (Eds.), Grid Services Engineering and Management. X, 165 pages. 2004.

Vol. 3266: J. Solé-Pareta, M. Smirnov, P.V. Mieghem, J. Domingo-Pascual, E. Monteiro, P. Reichl, B. Stiller, R.J. Gibbens (Eds.), Quality of Service in the Emerging Networking Panorama. XVI, 390 pages. 2004.

Vol. 3263: M. Weske, P. Liggesmeyer (Eds.), Object-Oriented and Internet-Based Technologies. XII, 239 pages. 2004.

Vol. 3260: I. Niemegeers, S.H. de Groot (Eds.), Personal Wireless Communications. XIV, 478 pages. 2004.

Vol. 3258: M. Wallace (Ed.), Principles and Practice of Constraint Programming – CP 2004. XVII, 822 pages. 2004.

Vol. 3256: H. Ehrig, G. Engels, F. Parisi-Presicce, G. Rozenberg (Eds.), Graph Transformations. XII, 451 pages. 2004.

Vol. 3255: A. Benczúr, J. Demetrovics, G. Gottlob (Eds.), Advances in Databases and Information Systems. XI, 423 pages. 2004.

Vol. 3254: E. Macii, V. Paliouras, O. Koufopavlou (Eds.), Integrated Circuit and System Design. XVI, 910 pages. 2004.

Vol. 3253: Y. Lakhnech, S. Yovine (Eds.), Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems. X, 397 pages. 2004.

Vol. 3250: L.-J. (LJ) Zhang, M. Jeckle (Eds.), Web Services. X, 300 pages. 2004.

Vol. 3249: B. Buchberger, J.A. Campbell (Eds.), Artificial Intelligence and Symbolic Computation. X, 285 pages. 2004. (Subseries LNAI).

Vol. 3246: A. Apostolico, M. Melucci (Eds.), String Processing and Information Retrieval. XIV, 332 pages. 2004.

Vol. 3245: E. Suzuki, S. Arikawa (Eds.), Discovery Science. XIV, 430 pages. 2004. (Subseries LNAI).

Vol. 3244: S. Ben-David, J. Case, A. Maruoka (Eds.), Algorithmic Learning Theory. XIV, 505 pages. 2004. (Subseries LNAI).

Vol. 3242: X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tiňo, A. Kabán, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature - PPSN VIII. XX, 1185 pages. 2004.

Vol. 3241: D. Kranzlmüller, P. Kacsuk, J.J. Dongarra (Eds.), Recent Advances in Parallel Virtual Machine and Message Passing Interface. XIII, 452 pages. 2004.

Vol. 3240: I. Jonassen, J. Kim (Eds.), Algorithms in Bioinformatics. IX, 476 pages. 2004. (Subseries LNBI).

Vol. 3239: G. Nicosia, V. Cutello, P.J. Bentley, J. Timmis (Eds.), Artificial Immune Systems. XII, 444 pages. 2004.

Vol. 3238: S. Biundo, T. Frühwirth, G. Palm (Eds.), KI 2004: Advances in Artificial Intelligence. XI, 467 pages. 2004. (Subseries LNAI).

Vol. 3236: M. Núñez, Z. Maamar, F.L. Pelayo, K. Pousttchi, F. Rubio (Eds.), Applying Formal Methods: Testing, Performance, and M/E-Commerce. XI, 381 pages. 2004.

Vol. 3235: D. de Frutos-Escrig, M. Núñez (Eds.), Formal Techniques for Networked and Distributed Systems – FORTE 2004. X, 377 pages. 2004.

Vol. 3232: R. Heery, L. Lyon (Eds.), Research and Advanced Technology for Digital Libraries. XV, 528 pages. 2004.

Vol. 3229: J.J. Alferes, J. Leite (Eds.), Logics in Artificial Intelligence. XIV, 744 pages. 2004. (Subseries LNAI).

Vol. 3225: K. Zhang, Y. Zheng (Eds.), Information Security. XII, 442 pages. 2004.

Vol. 3224: E. Jonsson, A. Valdes, M. Almgren (Eds.), Recent Advances in Intrusion Detection. XII, 315 pages. 2004.

Vol. 3223: K. Slind, A. Bunker, G. Gopalakrishnan (Eds.), Theorem Proving in Higher Order Logics. VIII, 337 pages. 2004.

Vol. 3222: H. Jin, G.R. Gao, Z. Xu, H. Chen (Eds.), Network and Parallel Computing. XX, 694 pages. 2004.

Vol. 3221: S. Albers, T. Radzik (Eds.), Algorithms – ESA 2004. XVIII, 836 pages. 2004.

Vol. 3220: J.C. Lester, R.M. Vicari, F. Paraguaçu (Eds.), Intelligent Tutoring Systems. XXI, 920 pages. 2004.

Vol. 3219: M. Heisel, P. Liggesmeyer, S. Wittmann (Eds.), Computer Safety, Reliability, and Security. XI, 339 pages. 2004.

Vol. 3217: C. Barillot, D.R. Haynor, P. Hellier (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2004. XXXVIII, 1114 pages. 2004.

Vol. 3216: C. Barillot, D.R. Haynor, P. Hellier (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2004. XXXVIII, 930 pages. 2004.

Vol. 3215: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVII, 906 pages. 2004. (Subseries LNAI).

Vol. 3214: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVIII, 1302 pages. 2004. (Subseries LNAI).

Vol. 3213: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVIII, 1280 pages. 2004. (Subseries LNAI).

Vol. 3212: A. Campilho, M. Kamel (Eds.), Image Analysis and Recognition. XXIX, 862 pages. 2004.

Vol. 3211: A. Campilho, M. Kamel (Eds.), Image Analysis and Recognition. XXIX, 880 pages. 2004.

Vol. 3210: J. Marcinkowski, A. Tarlecki (Eds.), Computer Science Logic. XI, 520 pages. 2004.

Vol. 3209: B. Berendt, A. Hotho, D. Mladenic, M. van Someren; M. Spiliopoulou, G. Stumme (Eds.), Web Mining: From Web to Semantic Web. IX, 201 pages. 2004. (Subseries LNAI).

Vol. 3208: H.J. Ohlbach, S. Schaffert (Eds.), Principles and Practice of Semantic Web Reasoning. VII, 165 pages. 2004.

Vol. 3207: L.T. Yang, M. Guo, G.R. Gao, N.K. Jha (Eds.), Embedded and Ubiquitous Computing. XX, 1116 pages. 2004.

Vol. 3206: P. Sojka, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue. XIII, 667 pages. 2004. (Subseries LNAI).

Vol. 3205: N. Davies, E. Mynatt, I. Siio (Eds.), UbiComp 2004: Ubiquitous Computing. XVI, 452 pages. 2004.

Vol. 3203: J. Becker, M. Platzner, S. Vernalde (Eds.), Field Programmable Logic and Application. XXX, 1198 pages. 2004.

Vol. 3202: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Knowledge Discovery in Databases: PKDD 2004. XIX, 560 pages. 2004. (Subseries LNAI).

Vol. 3201: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Machine Learning: ECML 2004. XVIII, 580 pages. 2004. (Subseries LNAI).

Vol. 3199: H. Schepers (Ed.), Software and Compilers for Embedded Systems. X, 259 pages. 2004.

Vol. 3198: G.-J. de Vreede, L.A. Guerrero, G. Marín Raventós (Eds.), Groupware: Design, Implementation and Use. XI, 378 pages. 2004.

Vol. 3195: C.G. Puntonet, A. Prieto (Eds.), Independent Component Analysis and Blind Signal Separation. XXIII, 1266 pages. 2004.

Vol. 3194: R. Camacho, R. King, A. Srinivasan (Eds.), Inductive Logic Programming. XI, 361 pages. 2004. (Subseries LNAI).

Vol. 3193: P. Samarati, P. Ryan, D. Gollmann, R. Molva (Eds.), Computer Security – ESORICS 2004. X, 457 pages. 2004.

Vol. 3192: C. Bussler, D. Fensel (Eds.), Artificial Intelligence: Methodology, Systems, and Applications. XIII, 522 pages. 2004. (Subseries LNAI).

Vol. 3191: M. Klusch, S. Ossowski, V. Kashyap, R. Unland (Eds.), Cooperative Information Agents VIII. XI, 303 pages. 2004. (Subseries LNAI).

Vol. 3190: Y. Luo (Ed.), Cooperative Design, Visualization, and Engineering. IX, 248 pages. 2004.

Vol. 3189: P.-C. Yew, J. Xue (Eds.), Advances in Computer Systems Architecture. XVII, 598 pages. 2004.

Vol. 3188: F.S. de Boer, M.M. Bonsangue, S. Graf, W.-P. de Roever (Eds.), Formal Methods for Components and Objects. VIII, 373 pages. 2004.

Vol. 3187: G. Lindemann, J. Denzinger, I.J. Timm, R. Unland (Eds.), Multiagent System Technologies. XIII, 341 pages. 2004. (Subseries LNAI).

Vol. 3186: Z. Bellahsène, T. Milo, M. Rys, D. Suciu, R. Unland (Eds.), Database and XML Technologies. X, 235 pages. 2004.

Vol. 3185: M. Bernardo, F. Corradini (Eds.), Formal Methods for the Design of Real-Time Systems. VII, 295 pages. 2004.

Vol. 3184: S. Katsikas, J. Lopez, G. Pernul (Eds.), Trust and Privacy in Digital Business. XI, 299 pages. 2004.

Vol. 3183: R. Traunmüller (Ed.), Electronic Government. XIX, 583 pages. 2004.

Vol. 3182: K. Bauknecht, M. Bichler, B. Pröll (Eds.), E-Commerce and Web Technologies. XI, 370 pages. 2004.

Vol. 3181: Y. Kambayashi, M. Mohania, W. Wöß (Eds.), Data Warehousing and Knowledge Discovery. XIV, 412 pages. 2004.

Vol. 3180: F. Galindo, M. Takizawa, R. Traunmüller (Eds.), Database and Expert Systems Applications. XXI, 972 pages. 2004.

Vol. 3179: F.J. Perales, B.A. Draper (Eds.), Articulated Motion and Deformable Objects. XI, 270 pages. 2004.

Vol. 3178: W. Jonker, M. Petkovic (Eds.), Secure Data Management. VIII, 219 pages. 2004.

Vol. 3177: Z.R. Yang, H. Yin, R. Everson (Eds.), Intelligent Data Engineering and Automated Learning – IDEAL 2004. XVIII, 852 pages. 2004.

Vol. 3176: O. Bousquet, U. von Luxburg, G. Rätsch (Eds.), Advanced Lectures on Machine Learning. IX, 241 pages. 2004. (Subseries LNAI).

Vol. 3175: C.E. Rasmussen, H.H. Bülthoff, B. Schölkopf, M.A. Giese (Eds.), Pattern Recognition. XVIII, 581 pages. 2004.

Vol. 3174: F. Yin, J. Wang, C. Guo (Eds.), Advances in Neural Networks - ISNN 2004. XXXV, 1021 pages. 2004.

Vol. 3173: F. Yin, J. Wang, C. Guo (Eds.), Advances in Neural Networks – ISNN 2004. XXXV, 1041 pages. 2004.

Vol. 3172: M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, T. Stützle (Eds.), Ant Colony, Optimization and Swarm Intelligence. XII, 434 pages. 2004.

Vol. 3171: A.L.C. Bazzan, S. Labidi (Eds.), Advances in Artificial Intelligence – SBIA 2004. XVII, 548 pages. 2004. (Subseries LNAI).

Vol. 3170: P. Gardner, N. Yoshida (Eds.), CONCUR 2004 - Concurrency Theory. XIII, 529 pages. 2004.

Vol. 3166: M. Rauterberg (Ed.), Entertainment Computing – ICEC 2004. XXIII, 617 pages. 2004.

Vol. 3163: S. Marinai, A. Dengel (Eds.), Document Analysis Systems VI. XI, 564 pages. 2004.

Preface

This volume contains the proceedings of the 24th IFIP TC 6/WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2004), held in Madrid, Spain, September 27–30, 2004. FORTE denotes a series of international working conferences on formal description techniques applied to computer networks and distributed systems. The conference series started in 1981 under the name PSTV. In 1988 a second series under the name FORTE was set up. Both series were united to FORTE/PSTV in 1996. Three years ago the conference name was changed to its current form. The last five meetings of this well-established conference series were held in Beijing, China (1999), Pisa, Italy (2000), Cheju Island, Korea (2001), Houston, USA (2002), and Berlin, Germany (2003).

The scope of the papers presented at FORTE 2004 covered semantic models and application of formal description languages (in particular, automata and Petri Nets), as well as the verification and testing of communication and distributed systems. The conference was preceded by 2 half-day tutorials by Roberto Gorrieri and Farn Wang. The proceedings contain the 20 regular papers accepted and presented at the conference. They were selected from 54 submitted papers in a careful selection procedure based on the assessment of three referees for each paper. The proceedings also include the papers contributed by the three invited speakers: Martín Abadi, Tommaso Bolognesi, and Juan Quemada.

FORTE 2004 was organized under the auspices of IFIP TC 6 by Universidad Complutense de Madrid. We would like to express our gratitude to the numerous people who contributed to the success of FORTE 2004. The reviewing process was one of the major efforts during the preparation of the conference, including not only PC members but additional reviewers. Finally, we would like to thank the local organizers for the excellent running of the conference. In particular, Luis Llana (as Web master), Natalia López (as organizing chair), and Fernando Rubio (as publicity chair) deserve a special mention. Last, but not least, we are in debt to Richard van de Stadt, the author of CyberChair, the tool we used to deal with the whole organization process.

We would like to mention that this is the first time that FORTE had three colocated workshops:

- TheFormEMC: 1st International Workshop on Theory Building and Formal Methods in Electronic/Mobile Commerce
- EPEW: 1st European Performance Engineering Workshop
- ITM: 1st International Workshop on Integration of Testing Methodologies

The proceedings of these workshops, which took place on the 1st and 2nd of October in Toledo, are also published by Springer in the LNCS series.

September 2004

David de Frutos-Escrig
Manuel Núñez

Organizing Entities

Universidad Complutense de Madrid

Universidad de Castilla La Mancha

International Federation for Information Processing

Other Sponsors

Ministerio de Educación y Ciencia, Junta de Comunidades de Castilla–La Mancha, Excmo. Ayuntamiento de Madrid, Comunidad Autónoma de Madrid, Facultad de Informática (UCM).

Executive Committee

Conference Chairs	David de Frutos-Escrig
	Manuel Núñez
Publicity Chair	Fernando Rubio
Organizing Chair	Natalia López
Workshops Chair	Valentín Valero

Organizing Committee

Alberto de la Encina	Luis Llana	Ismael Rodríguez
Carlos Gregorio	Natalia López	Fernando Rubio
Mercedes Hidalgo	Olga Marroquín	Alberto Verdejo

Steering Committee

Gregor v. Bochmann	University of Ottawa, Canada
Tommaso Bolognesi	Istituto di Elaborazione della Informazione, Italy
Guy Leduc	University of Liege, Belgium
Kenneth Turner	University of Stirling, UK

We want to express our gratitude for the collaboration of the former members of the Steering Committee: Ed Brinksma, Stan Budkowski, Elie Najm, and Richard Tenney.

Program Committee

Gregor von Bochmann	Univ. of Ottawa, Canada
Tommaso Bolognesi	IEI Pisa, Italy
Mario Bravetti	University of Bologna, Italy
Ana Cavalli	INT Evry, France
Jean Pierre Courtiat	LAAS Toulouse, France
David de Frutos-Escrig	Complutense University of Madrid, Spain
Rachida Dssouli	Concordia University, Montreal, Canada
Reinhard Gotzhein	University of Kaiserslautern, Germany
Holger Hermanns	Saarland University, Germany
Teruo Higashino	Osaka University, Japan
Dieter Hogrefe	University of Göttingen, Germany
Gerard J. Holzmann	Bell Labs, USA
Claude Jard	IRISA, France
Myungchul Kim	ICU Taejon, Korea
Hartmut König	Brandenburg University of Technology, Germany
Maciej Koutny	University of Newcastle upon Tyne, UK
Guy Leduc	University of Liege, Belgium
David Lee	Bell Labs, China
Elie Najm	ENST, France
Manuel Núñez	Complutense University of Madrid, Spain
Doron A. Peled	University of Warwick, UK
Alexandre Petrenko	CRIM Montreal, Canada
Kenji Suzuki	Kennisbron Co., Japan
Ken Turner	University of Stirling, UK
Hasan Ural	University of Ottawa, Canada
Ümit Uyar	City University of New York, USA
Valentín Valero	Universidad Castilla-La Mancha, Spain
Farn Wang	National Taiwan University, Taiwan
Jianping Wu	Tsinghua University, Beijing, China
Nina Yevtushenko	Tomsk State University, Russia

Additional Reviewers

Baptiste Alcalde
Gerd Behrmann
Bernard Berthomieu
Sergiy Boroday
Diego Cazorla
Robert Clark
Fernando Cuartero Gómez
Raymond Devillers
Chen Dongluo
Nicola Dragoni
A. Dury
Michael Ebner
Abdeslam En-Nouaary
Xiaoming Fu
Stefania Gnesi
Cyril Grepet
Claudio Guidi
Serge Haddad
Hesham Hallal
Toru Hasegawa
Klaus Havelund
May Haydar
Loïc Hélouët
Jia Le Huo
Akira Idoue
David N. Jansen
Thierry Jéron
Cai Jianpeng
Sungwon Kang
Tatjana Kapus
Victor Khomenko
Hanna Klaudel
Anton Kolomeets
Vitali Kozioura
Jean Leneutre
Keqin Li
Natalia López
Roberto Lucchi
Luis Llana
Hermenegilda Macià Soler
Savi Maharaj
Olga Marroquín Alonso
Narciso Martí-Oliet
Michael Meier
Nicola Mezzetti
Gethin Norman
Tomohiko Ogishi
Jean-Marie Orset
Miguel Palomino
Juan José Pardo Mateo
Svetlana Prokopenko
Ismael Rodríguez
Elisangela Rodriguez Vieira
Fernando Rubio
Vlad Rusu
Claudio Sacerdoti Coen
Joern Schneider
Christel Seguin
Koushik Sen
Soonuk Seol
Leslie Smith
Rene Soltwisch
Natalia Spitsyna
Alexandre Tauveron
Dong Wang
Yin Xia
Shi Xingang
Wang Zhiliang
Li Zhongjie
Bachar Zouari

Table of Contents

A Logical Account of NGSCB

Martín Abadi[1] and Ted Wobber[2]

[1]University of California at Santa Cruz
[2]Microsoft Research, Silicon Valley

Abstract. As its name indicates, NGSCB aims to be the "Next-Generation Secure Computing Base". As envisioned in the context of Trusted Computing initiatives, NGSCB provides protection against software attacks. This paper describes NGSCB using a logic for authentication and access control. Its goal is to document and explain the principals and primary APIs employed in NGSCB.

1 Introduction

NGSCB ("Next-Generation Secure Computing Base", formerly known as "Palladium") integrates hardware and software components that aim to help in protecting data and processes against software attacks [8,9,14,15]. The hardware includes a cryptographic co-processor that contains keys and offers basic cryptographic services. The software includes new, trusted operating system components.

While the architecture and the implementation of NGSCB continue to evolve, quite a few of its features have been discussed in public. We believe that it is worthwhile to elucidate them further. Many of these features seem likely to remain important as NGSCB matures, and also appear in other projects and research efforts in the area of Trusted Computing [1,10,12,13,16].

In this paper, we present an attempt to understand the fundamentals of NGSCB in terms of a logic for authentication and access control. This formalism had its origins in the context of the Taos operating system and of the Digital Distributed System Security Architecture [11,12,18]. In this application to NGSCB, we use the logic for describing relationships between principals while abstracting away most of the details of the underlying cryptographic protocols. Although it may be feasible and perhaps attractive, we do not relate the logic to concrete implementations, nor base new implementations on the logic. Our goal is to document the components and primary APIs employed in NGSCB, and to provide concise and principled explanations for them.

At present, one may view all work on NGSCB as "work in progress". This paper is no exception. Because the specifics of NGSCB remain subject to change, we are less concerned with giving a detailed and up-to-the-minute account than with providing a consistent explanation of important concepts and techniques.

The next section reviews the logic. Section 3 reviews the basics of NGSCB. Sections 4, 5, and 6 describe principals, derived authorities, and the certification infrastructure (which is external to NGSCB but necessary for its applications).

D. de Frutos-Escrig and M. Núñez (Eds.): FORTE 2004, LNCS 3235, pp. 1-12, 2004.

Section 7 deals with the main system services in logical terms. Section 8 briefly addresses privacy. Section 9 discusses an example. Section 10 concludes.

2 A Brief Logic Review

The logic enables us to describe a system in terms of principals and their statements. The logical formula *P says s* means that principal *P* makes or supports statement *s*. The principal may for example be a user, a piece of hardware, the combination of some hardware and some software, or a cryptographic key.

We also allow compound principals, particularly of the form $P \mid C$. The meaning of $P \mid C$ is "*P* quoting *C*"; we have that $P \mid C$ says *s* when *P* says that *C* says *s*. For instance, *P* may represent a piece of hardware, and *C* a piece of code or a user.

In addition, the logical formula $P \Rightarrow Q$ means that *P* speaks for *Q*, so if *P says s* then *Q says s*, for every *s*. We generally assume the *hand-off axiom* which says that if *Q says* $P \Rightarrow Q$ then indeed $P \Rightarrow Q$. We use the "speaks for" relation for many purposes. For instance, we often write that a key *K* speaks for a principal *P* when *K* is *P*'s public signature key. Typically only *P* knows the corresponding signing key (*K*'s inverse) and can produce signatures that can be checked with *K*. We may also write that a principal speaks for any group of which it is a member; thus, when we write that *P* speaks for a group we typically mean that *P* is or speaks for some member of the group, not necessarily all members of the group. (It would be easy to extend the logic with a membership relation, and to replace "speaks for" with membership in these uses; whether this extension is worth the trouble remains open to debate.) We may represent an access control list (ACL) as the group of the principals authorized by the list. If G_X is the ACL for accessing an object *X*, then *P* speaks for G_X when *P* is authorized to access *X*.

Although the logic does not lead to correctness proofs of the kind expected in high-assurance systems, this logic and its relatives have been useful in several ways in the past. Much as in this paper, the logic has served for describing and documenting the workings of a system, what the system achieves, and on what assumptions it relies, after the fact or in the course of development. In this respect, formal notations do not accomplish anything beyond the reach of careful, precise prose, but they are helpful. The logic has also served in validating particular techniques for authorization, reducing them to logical reasoning, and also as a basis for new techniques; research on stack inspection and proof-carrying authorization exemplify this line of work [3,6,17,18]. Finally, the logic has served as a foundation for languages for writing general security policies [7].

We refer to previous papers for more detailed descriptions of the logic and its applications.

3 Assumptions on NGSCB

We assume that at the root of any NGSCB node there exists a hardware-based security facility that implements cryptosystems, random number generation, and key storage. We use the term *Security Support Component* (*SSC*) to describe this facility

since it has been previously used in related literature. The Trusted Platform Module (TPM) from the Trusted Computing Group [16] may be the main current example of an SSC.

We further assume that the hardware has the capability to load an operating system that can be reliably identified by taking a hash (or *code-id*) of the initial operating system image and data. This operating system, so loaded, will reside in a protected area of memory that cannot be accessed by untrusted code that the operating system might load. Therefore, the hardware has reason to believe that the statements made by the securely loaded operating system can be attributed to the principal identified by the code-id. In turn, the operating system can load a child process and attribute statements made by that child process to the principal identified by the hash of its code and data. Following one existing nomenclature for NGSCB, we call the securely loaded operating system the *Nexus*, and we call any child process that it loads a *Nexus Computing Agent* (*NCA*). The Nexus may be implemented, for example, by combining a virtual-machine hypervisor with a trusted guest operating system [10,15]. For simplicity, we focus on situations with one distinguished Nexus and one distinguished NCA; of course, other software may be running on the hardware at the same time.

Finally, we assume trusted input and output paths for communication with users. In particular, the hardware may guarantee that only a particular Nexus receives input from the keyboard and can send output to a display. The Nexus may in turn provide a similar guarantee to a particular NCA.

These assumptions are consistent with previous, public descriptions of NGSCB, such as the ones found in papers and on the Web. Those descriptions, like most informal descriptions, are however incomplete and imprecise in some respects. One of the goals of the present logical account is to complement those descriptions, with additional details (some of them validated in private conversations with the NGSCB team, and some of them conjectured rather than based on an official NGSCB design), and with a partial rationale for the workings of NGSCB.

4 Principals

Next we enumerate principals relevant for NGSCB security.

The following principals are particular to each NGSCB node. Each node will have different instances of them.

K_0	the permanent public key of the SSC
K_T	a per-boot public key of the SSC
S_0	the master symmetric key of the SSC
S_T	a per-boot symmetric key derived from S_0

The inverse of the key K_0 and the symmetric key S_0 never leave the SSC hardware; the inverse of the key K_T and the symmetric key S_T may or may not leave the hardware, as discussed below. We rely on asymmetric cryptography (public-key operations with K_0, K_T, and their inverses) primarily for digital signatures, rather than for public-key encryption. When encryption is needed, we indicate it explicitly. The

symmetric key S_0 may be replaced with a pair of keys for asymmetric encryption, with only minor changes.

The following principals represent software images. There can, of course, be many different images in which we might be interested. For simplicity of exposition, we will be concerned with only two:

C_{NEX}	the code-id of a particular Nexus
C_{NCA}	the code-id of a particular NCA

The following principals complete the cast; they provide the context outside an NGSCB system:

M	a manufacturer of SSC hardware
V	a vendor or author of Nexus software
A	a vendor or author of NCA software
K_M	M's public signature key
K_V	V's public signature key
K_A	A's public signature key
G_M	a group of public signature keys for SSCs produced by M
G_V	a group of code-ids for Nexus software images produced by V
G_A	a group of code-ids for NCA software images produced by A
CA	a trusted certification authority

5 Derived Authorities

It would be possible for an SSC to make statements only with its permanent public key K_0. However, it is desirable to sign as few certificates as possible with this key. Therefore, we assume that, at boot time, the SSC generates a temporary key pair, consisting of the public key K_T and its inverse. Then K_0 transfers all of its authority to K_T. This hand-off of authority is captured in the following statement:

$$K_0 \text{ says } K_T \Rightarrow K_0$$

The certificate described here, and all subsequent certificates mentioned in this paper, should be considered valid only for a limited period of time. The logic does not directly model time, so we do not represent time formally; one could probably add it with a modest effort.

In this formulation, we assume that the SSC holds the temporary secret key (the inverse of K_T) in hardware and uses the key for signing statements on behalf of other principals on the local machine. This key could instead reside in the Nexus and be accessed through a similar interface. In this case, the key would no longer be protected within the SSC, so the two arrangements entail different security properties.

Because of the secure loading steps described above, a successfully loaded Nexus will have the authority of the compound principal $K_T \mid C_{NEX}$. An SSC can run any Nexus software, but the rights of a specific Nexus instance are exactly those of the SSC parameterized by the code-id of the Nexus. Similarly, an NCA loaded on top of a Nexus would speak as $K_T \mid C_{NEX} \mid C_{NCA}$.

6 Certification Infrastructure

In order to deduce anything useful about statements made by the software running on an NGSCB node, we must have trust assumptions. We hypothesize the presence of a certification authority *CA* that makes statements that are globally trusted. In particular, we trust *CA* to specify the set of acceptable NGSCB nodes and the set of trusted Nexus and NCA software images; we express this trust as follows:

$$CA \Rightarrow G_M$$
$$CA \Rightarrow G_V$$
$$CA \Rightarrow G_A$$

We simplify a bit here: in practice, *CA* will almost always be implemented by a hierarchy of certification authorities and there will be multiple subgroups of G_M, G_V, and G_A, according to the intended applications and trust relationships.

Next, we must give some key (or set of keys) the authority to certify membership in the groups G_M, G_V, and G_A. We represent such statements in the following certificates:

$$CA \text{ says } K_M \Rightarrow G_M$$
$$CA \text{ says } K_V \Rightarrow G_V$$
$$CA \text{ says } K_A \Rightarrow G_A$$

Finally, we use the signing keys that correspond to K_M, K_V, and K_A for making membership certificates for the specific hardware/software stack that we intend to construct:

$$K_M \text{ says } K_0 \Rightarrow G_M$$
$$K_V \text{ says } C_{NEX} \Rightarrow G_V$$
$$K_A \text{ says } C_{NCA} \Rightarrow G_A$$

Combining the certificates and trust assumptions, we can derive:

$$K_0 \Rightarrow G_M$$
$$C_{NEX} \Rightarrow G_V$$
$$C_{NCA} \Rightarrow G_A$$

Note that if K_0 is a member of G_M (so $K_0 \Rightarrow G_M$ in our model) then K_0 can also define new group members of G_M. In particular, $K_0 \Rightarrow G_M$ and $K_0 \text{ says } K_T \Rightarrow K_0$ imply that $K_T \Rightarrow G_M$. Using a primitive membership relation rather than "speaks for" would remove this possibility.

7 Programmatic Interface

The programmatic interface of NGSCB supports the sealing of information and hardware-based attestation. Next we explain those functions in terms of the logic and of the definitions of the previous sections.