Army Secure Operating System: Information Security for Real Time Systems

Eric R. Anderson
Ben L. Di Vito
Ruth M. Hart

TRW Defense Systems Group
One Space Park
Redondo Beach, CA 90278

Abstract

The Army Secure Operating System (ASOS) project, under the management of the U.S. Army CECOM organization, will provide real time systems software necessary for fielding modern Battlefield Automation Systems. The ASOS objective is to develop a family of operating systems for tactical data system applications. ASOS will support real time applications software coded in Ada. In addition, ASOS will incorporate information security technology to protect classified data processed by Army tactical systems. The security technology is based on the National Computer Security Center's Trusted Computer System Evaluation Criteria (TCSEC). Current plans include two members of the ASOS family: a Dedicated Secure Operating System at the TCSEC C2 level and a Multilevel Secure Operating System at the TCSEC A1 level. In the paper we present an overview of the ASOS concept and TRW's solution to the ASOS design and analysis problem.

1 Introduction

Fielding tactical data systems is a problem of great importance to the U.S. Army. Battlefield Automation Systems are desired that utilize the most modern computer technology and capitalize on the wide variety of product offerings available today. The increasing power and decreasing cost of microcomputers in particular are making possible significant advances in tactical data applications. Less visible but equally important advances in software technology are making valuable contributions as well.

A significant problem facing commands that field tactical data systems is information security. National recognition of a growing threat to information security is eliciting an ever stronger Government response [1]. Tactical systems face a threat because they process data classified at different sensitivity levels. Security is achieved in practice, but current methods of protecting classified information are costly and awkward, relying heavily on operation in dedicated mode or system-high mode. A real need for multilevel security exists for a variety of reasons:

- Message traffic at different security levels requires automated ha

- Intelligence sources are often classified higher than message contents.
- Databases contain both classified and unclassified (or less highly classified) information. Often the highly classified material is only a small fraction of the total data stored.
- Systems containing data at different classification levels require interconnection, forming a larger system containing the union of all the data and users.

Thus, to meet the needs of many tactical information systems and keep pace with both mission and technology trends, operation in multilevel mode is sought by the Army.

Technology for implementing computer security has reached a certain state of maturity. Systems designed to meet the Trusted Computer System Evaluation Criteria (TCSEC) [2] have been developed and new products continue to emerge. Of particular importance is the demonstrated feasibility of product development at the A1 level [3]. Using products evaluated in the B2–A1 range, true multilevel operating modes are possible. The ASOS program is designed to yield two certifiable operating system products that address the needs of information security for tactical data systems:

- A Dedicated Secure (DS) operating system evaluated at the C2 level. This system will be optimized for real time performance and limits security considerations to those of discretionary controls needed for operation in dedicated or system-high mode.
- A Multilevel Secure (MLS) operating system evaluated at the A1 level. This system will stress state-of-the-art information security for operation in multilevel mode.

Both systems will accommodate the functional needs of real time applications developed in Ada.

The presentation that follows includes a description of the ASOS concepts and requirements, the ASOS architecture, and a discussion of the analysis activities employed to assure system security. Emphasis is placed on the multilevel secure version of ASOS in the remainder of the paper.

2 ASOS Requirements

ASOS satisfies the Army's need for a family of tactical operating systems. The role of operating systems is to provide a bridge between applications software, which is responsible for performing mission-specific functions, and the computer system hardware. Operating systems provide hardware resource management and a host of other services to relieve applications software from such low-level chores. Many of the same low-level functions are required for every application. Therefore, by putting these services into a set of common operating systems, they can serve a broad class of mission-specific applications.

The ASOS concept is to provide a family of operating systems to support tactical mission applications. Essential features of this family are the following:

- Two operating systems are planned for certification at the C2 and A1 levels of the Trusted Computer System Evaluation Criteria.

Accession Number: 5086

Title: Army Secure Operating System: Information Security for Real Time Systems

Personal Author: Anderson, E.R.; DiVito, B.L.; Hart, R.M.

Corporate Author Or Publisher: TRW Defense Systems Group, One Space Park, Redondo Beach, CA 90278

Descriptors, Keywords: Army Security Operating System Real Time ASOS Software Battlefield Automation Trusted Computer

Pages: 00011

Cataloged Date: Aug 10, 1994

Document Type: HC

Number of Copies In Library: 000001

Record ID: 29348

- They are optimized to support applications software written in Ada.
- They are designed to be transportable to different instruction set architectures and computer systems.
- They are designed for real time efficiency.
- The family will be extendible to distributed system environments having multiprocessor or network based architectures.

These general requirements address the needs of many current and future Army programs. More detailed requirements have been distilled from studies of these applications and have been incorporated into the ASOS program. The result will be a widely applicable family of operating systems.

The advent of the Ada programming language and its associated software development technology has broadened the role of operating systems somewhat. Unlike most programming languages, Ada provides a very rich environment for developing applications, including facilities for concurrent programming. This is an area that traditionally has been outside the bounds of the language and its compiler, resulting in applications software that often is heavily dependent on its particular operating system features for correct execution. Ada, however, provides the language features and run-time support for multitasking. This situation leads to an overlap of the traditional domains of operating systems and programming language systems. Much of the ASOS functionality is concerned with this special kind of support required by Ada programs.

Figure 1 shows the abstract environment for developing and executing Ada applications software. Programs are developed on a host computer and compiled for a particular target computer. Bound together with the compiled program is a subset of the Runtime Support Library (RSL) provided by the compiler vendor. The total package is loaded on the target computer for execution under the control of ASOS. During execution, ASOS manages the hardware resources, controls the interface to peripherals and input/output devices, and provides additional RSL functions. All the while, ASOS ensures that programs and users adhere to the appropriate security policy in force for the particular mission environment and operating mode.

The MLS version of ASOS must implement the notion of *mandatory access control (MAC)*. This concept involves controlling access to information *objects* (passive entities) by system *subjects* (active entities). Subjects are associated with clearances and objects with classifications, and the DoD security policy on access to classified information is enforced. Mandatory controls in ASOS are based on a mathematical policy formulation due to Bell and LaPadula [4], from which the ASOS *formal security model* has been derived.

A subordinate security policy known as *discretionary access control (DAC)* is provided by both the dedicated and multilevel secure versions of ASOS. This policy is intended for controlling access on a need-to-know basis. It allows access privileges and restrictions to be specified for named individuals and information objects, with subsequent alteration of the access rights possible. Naturally, the MAC policy always takes precedence over the DAC policy.

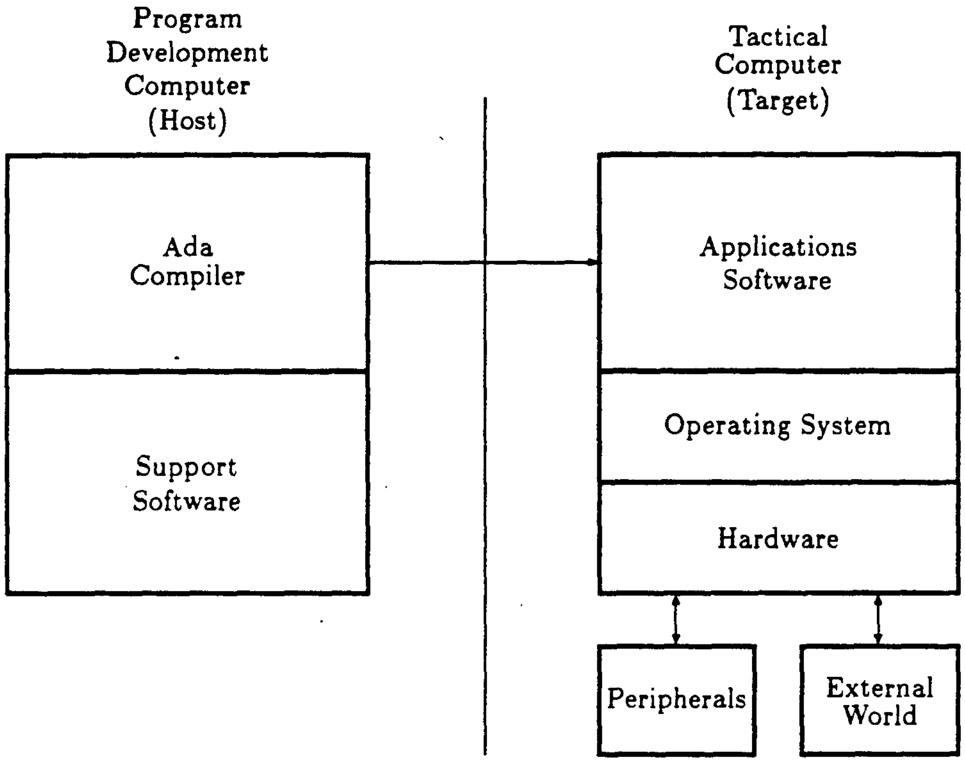


Figure 1: ASOS components on host and target computers.

3 ASOS Architecture

TRW's solution to the ASOS design problem is based on a careful integration of conventional operating system design principles and computer security principles. Our architecture has been developed to meet the performance and functionality demands of real time tactical systems while simultaneously offering high levels of information security. Tradeoffs among these goals have been exercised to arrive at the C2 and A1 members of the ASOS family. Both systems, however, are based on a common ASOS design.

Our solution to the information security problem is based on sound and proven principles of computer security, centering on the implementation of a *Trusted Computing Base (TCB).* The TCB includes any portion of the system design necessary to enforce the security policy or whose incorrect operation could inadvertently lead to a compromise of classified information. Central to the TCB concept is the *reference monitor*, which is the function responsible for performing checks on every access attempt and for making decisions to grant or deny access requests.

In ASOS, the reference monitor is realized by a combination of hardware and software mechanisms. The hardware mechanisms consist of protection features provided by the target computer hardware. These include multidomain computer architectures for control

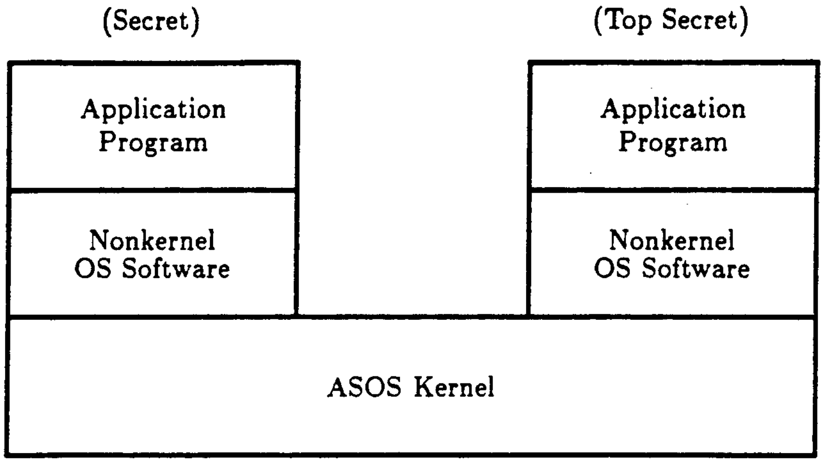


Figure 2: ASOS software architecture.

of privileges, memory management units to restrict access to physical memory, and a means to control access to input/output devices. Such features are generally found on virtually all modern processors and do not impose any unusual requirements on the hardware. ASOS will use the Motorola M68020 microprocessor as its initial target hardware, which has an adequate set of hardware protection features.

At the heart of the ASOS TCB is its *security kernel*, which is the software portion of the reference monitor. ASOS relies heavily on a kernelized design as the basis for its TCB architecture. ASOS also relies on a largely software implementation of the TCB. Unlike the alternative approach of embedding large amounts of TCB functionality in specialized hardware [5], the ASOS approach has the advantage of capitalizing on commercially available hardware and ASOS software transportability to realize performance gains as computer vendors improve their products. In today's climate of very rapid advances in hardware speed, this strategy will guard against premature obsolescence.

A security kernel is similar to the conventional notion of an operating system kernel, except that it emphasizes the retention of a minimal set of functions so that only security relevant software is contained in the kernel. This notion is essential to achieving a high degree of trust in the design and implementation. A security kernel is also different in the sense that it adheres to a set of rules that embodies the enforcement of a security policy. Thus, the kernel enforces discretionary, and in the case of multilevel security, mandatory access controls.

Figure 2 displays a high level view of the ASOS software architecture. This architecture provides for three layers of software, two of which are in the operating system. In this design, the kernel is the lowest layer, closest to the hardware, and executes in the most privileged domain provided by the target computer. At a higher level of abstraction is the nonkernel OS software, which provides the interface to application programs and issues service requests to the kernel. The nonkernel OS software in the ASOS design is *not* part of the TCB. A separate copy of nonkernel OS data is maintained for each program. Together, an application program and its copy of the nonkernel OS software are considered a single security subject that executes at a specific security level. It is this level that determines what objects are accessible and in what modes. ASOS ensures that all subjects and objects are isolated from each other and that accesses are allowed only if they adhere to the security policy.

Security kernels are designed in accordance with three engineering principles to make them viable as reference monitor implementations:

- **Completeness.** All accesses must be mediated so that no subject can circumvent the reference monitor controls.
- **Isolation.** The kernel must be tamper-proof and self-protecting.
- **Verifiability.** The kernel must be small and well structured so that its correctness can be assured.

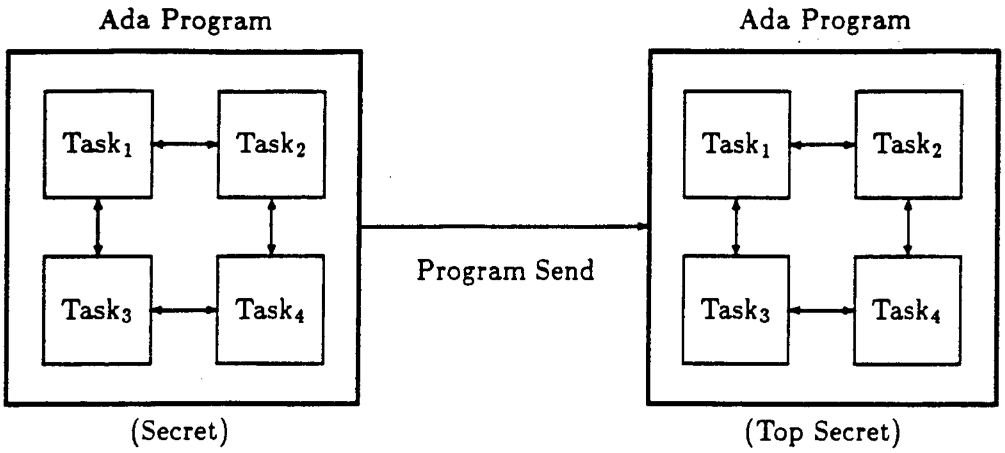
ASOS is being designed with these principles explicitly in mind to ensure the implementation of a trustworthy and high quality product.

Besides the kernel, there are several other operating system functions that must be placed within the TCB. Nonkernel trusted software is provided for in the ASOS design. These functions are required to support such activities as user authentication, operator command processing, and line printer spooling. Trusted programs are often granted exemptions from some of the security policy rules normally enforced by the kernel. In addition, they execute with a reduced set of system services to minimize the amount of trusted software required, which is all subject to design verification.

Support for the Ada concurrent programming features poses some special problems for the ASOS design. Due to the generality of the Ada tasking mechanisms and the relatively unconstrained manner in which Ada tasks can communicate with one another, it would be highly impractical to make Ada tasks separate security subjects. To do so would either allow flaws, because intertask communication is, in general, bidirectional and information might flow from a high level task to a low level one, or would require complex and expensive runtime checks to be sure no such flows were occurring.

TRW's solution makes a separate security subject out of an entire Ada program instance. Ordinary Ada tasking may still be used within a program, but all tasks within the same program instance execute at the same security level. Consequently, intertask communication and data sharing may be permitted according to the rules of Ada with impunity because of the grouping of tasks into programs. To allow information sharing among separate programs, ASOS provides message services for interprogram communication. Figure 3 illustrates this concept. Of course, these new program level services are subject to security policy restrictions and have been designed to perform appropriate runtime checks. These and other issues relating to the secure execution of Ada programs have been previously investigated and reported by TRW [6].

Overall, our ASOS design approach emphasizes strong architectural features to enforce security in a trustworthy manner. It incorporates access control at the Ada program level



Interprogram communication restricted by security policy.

Figure 3: Ada programs and tasks within ASOS.

and a modest set of services to complement Ada tasking. Real time efficiency is addressed and factored into the design. Modularity and good structure are stressed to enhance the understandability and verifiability of the design. Finally, a high degree of commonality between the C2 and A1 designs will promote transportability and maintainability.

4 A1 Level Assurance

In order to assure trustworthy system operation in multilevel mode, several unconventional development activities are required by the A1 criteria. Most of these involve performing special analyses to ensure that the system as designed and implemented does indeed enforce the system security policy. Substantial evidence must be presented to argue that a system deserves the kind of trust ascribed to the A1 level of evaluation.

Figure 4 illustrates the overall integration of the A1 assurance methodology with the conventional development activities required for ASOS. The cornerstone of A1 assurance techniques is the application of *formal methods*, that is, mathematically rigorous techniques designed to prove that a system properly enforces access control. The A1 criteria require a mathematical proof that the top level design of the TCB adheres to the constraints of a formal statement of security policy. Key steps in this process are as follows:

- Develop a *formal security model* that precisely captures the policy and essential security requirements.

- Express the top level TCB design in a *Formal Top Level Specification (FTLS)* using a suitable formal language.

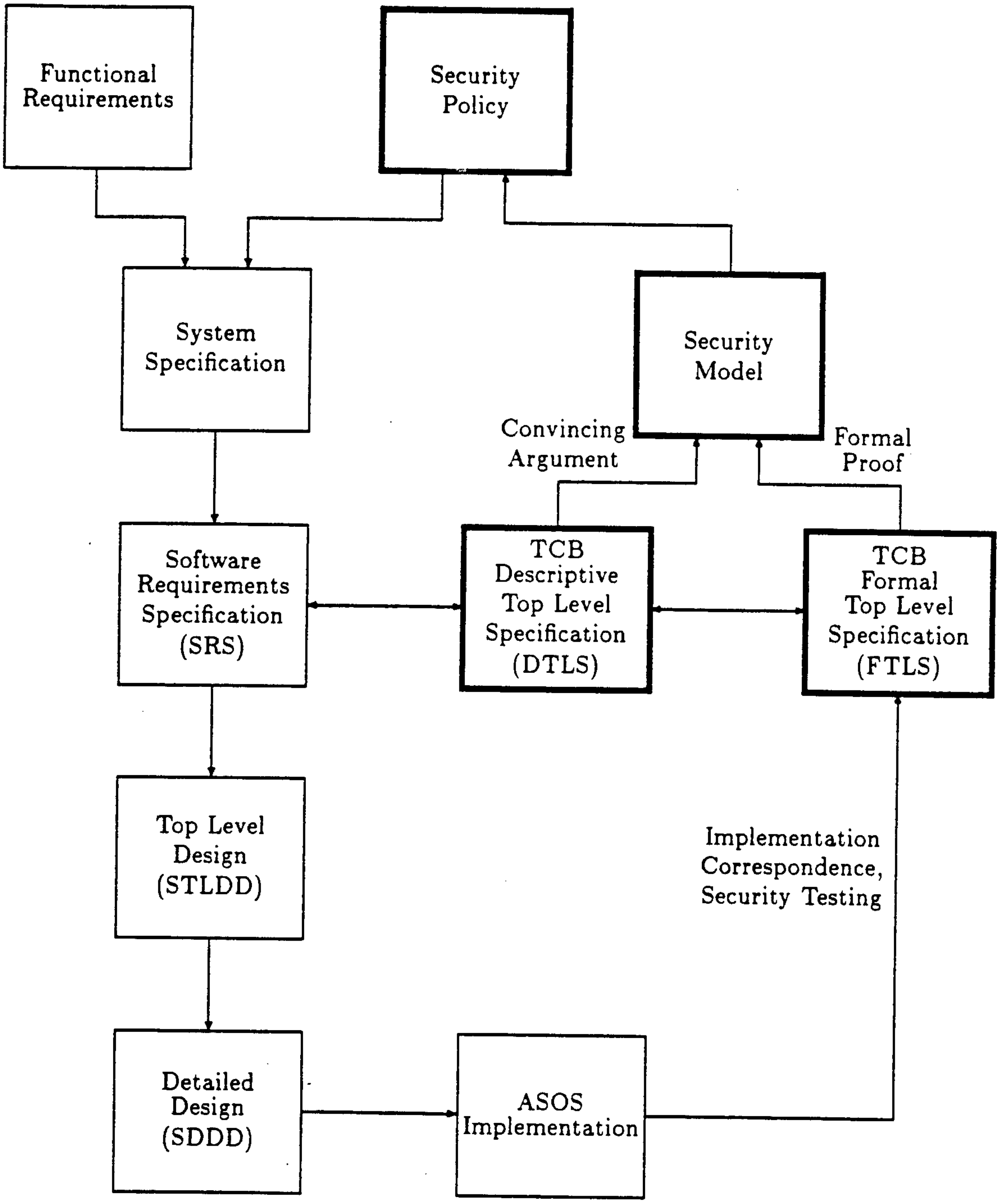

Functional Requirements
Security Policy
System Specification
Security Model
Convincing Argument
Formal Proof
Software Requirements Specification (SRS)
TCB Descriptive Top Level Specification (DTLS)
TCB Formal Top Level Specification (FTLS)
Top Level Design (STLDD)
Implementation Correspondence, Security Testing
Detailed Design (SDDD)
ASOS Implementation

Figure 4: A1 assurance methodology.

- Construct a rigorous proof, preferably with the aid of automated tools, that the FTLS complies with the security model.
- Analyze the FTLS and TCB design for the presence of covert information channels (leakage paths) that might be exploited to contravene the security policy.

Successful completion of these tasks provides strong evidence of a trustworthy design, and hence, of the delivered system's secure operation.

Additional assurance tasks required include developing a *Descriptive Top Level Specification (DTLS)* for the TCB, showing the correspondence between the TCB implementation and the FTLS, conducting a thorough security testing program, and following strong configuration management procedures. All the techniques mentioned above will be applied during the ASOS development effort. The end result will be a highly trustworthy product that has be carefully scrutinized to ensure correct and secure design.

TRW will be using the Gypsy methodology [7] to conduct the formal verification activities in our ASOS work. A set of tools known as the Gypsy Verification Environment (GVE) are available for carrying out the detailed steps to achieve formal design verification. These tools have been endorsed by the National Computer Security Center for use in developing A1 trusted computer systems.

Development of the formal proofs on ASOS will proceed as depicted in Figure 5. Increasingly detailed descriptions of the TCB will be subjected to proof against a set of abstract security properties. The first layer shows that the ASOS-extended Bell-LaPadula rules of operation are security preserving. At the second layer, a more detailed version of the rules is introduced and verified. Finally, the third layer and its proof constitute verification of FTLS compliance with the security model.

Our overall assurance approach applies state-of-the-art technology to ensure a high degree of confidence in the ASOS protection mechanisms. The Gypsy Verification Environment is the result of continuing research into the technology of trusted systems. TRW is currently working with the research community to enhance the automation provided by the GVE. We expect ASOS to benefit from further developments in verification tools and techniques, such as Ada verification tools, when they become available.

5 Conclusion

The Army Secure Operating System project is a concrete effort to develop a usable multilevel secure operating system for tactical data applications. It will provide real time support for executing and controlling Ada applications programs. The operating system itself is being written in Ada and will benefit from many of the latest advances in software technology. Transportability is a major goal of ASOS, which will allow it to keep up with the rapidly changing computer product offerings.

By emphasizing a family of systems with different characteristics and tradeoffs for performance and security, the Army will be better able to serve its tactical data system needs over a wide range of applications. Across the family, however, information security remains a significant consideration and driver of the ASOS designs. As mission environments move

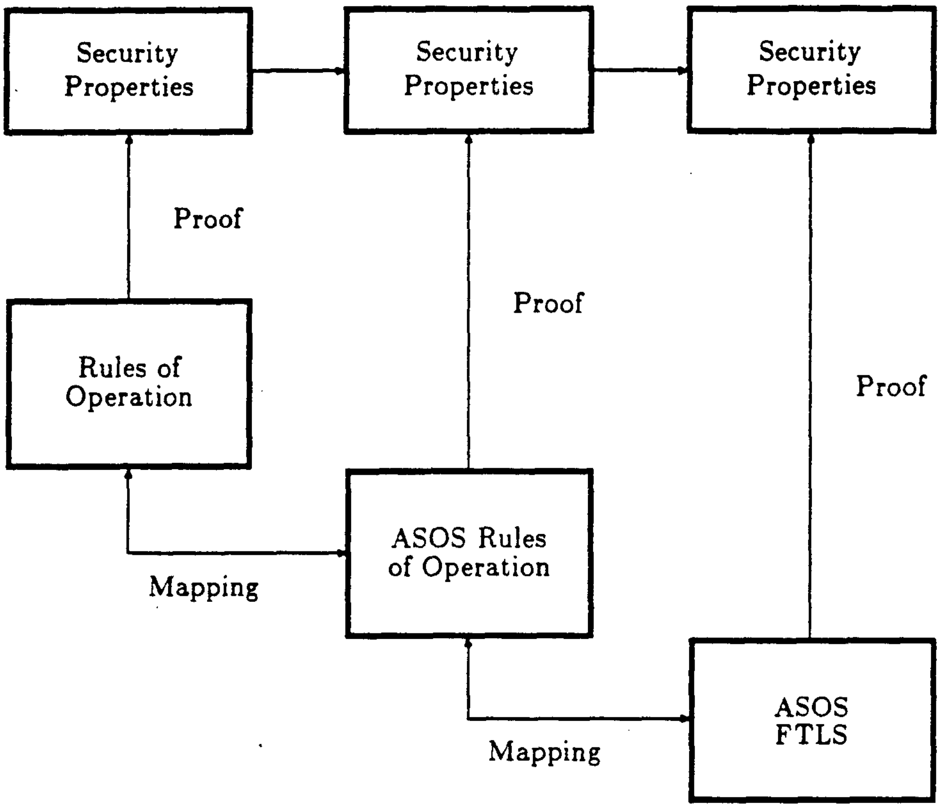


Figure 5: Layered approach to formal verification.

in the direction of higher levels of automation, incorporation of information system security will grow in importance. ASOS will provide a solid base on which to build tomorrow's mission applications and protect vital national security information.

Acknowledgements

This paper is the result of work conducted as part of the Army Secure Operating System (ASOS) project, which is sponsored by the United States Army Communications-Electronics Command and the National Computer Security Center, under contract No. DAAB07-86-CA032.

References

1. National Security Decision Directive 145, "National Policy on Telecommunications and Automated Information Systems Security," The White House, Washington, September 1984.

2. National Computer Security Center, "Department of Defense Trusted Computer System Evaluation Criteria," DoD 5200.28-STD, December 1985.

3. L. Fraim, "Scomp: A Solution to the Multilevel Security Problem," Computer, IEEE, July 1983.

4. D. E. Bell and L. J. LaPadula, "Secure Computer System: Unified Exposition and Multics Interpretation," Technical Report ESD-TR-75-306, Mitre Corporation, Bedford, Mass., March 1976.

5. W. E. Boebert, et al, "Secure Ada Target: Issues, System Design, and Verification," Proc. 1985 Symposium on Security and Privacy, IEEE, April 1985.

6. E. R. Anderson, "Ada's Suitability for Trusted Computer Systems," Proc. 1985 Symposium on Security and Privacy, IEEE, April 1985.

7. D. I. Good, B. L. DiVito, and M. K. Smith, "Using the Gypsy Methodology," Institute for Computing Science, University of Texas at Austin, June 1984.