NPS-CS-02-003

# NAVAL POSTGRADUATE SCHOOL
## Monterey, California


**A National Trusted Computing Strategy**

by

Cynthia E. Irvine
Timothy E. Levin
George W. Dinolt

May 2002



Prepared for: Center for MFOSEC Studies and Research

NAVAL POSTGRADUATE SCHOOL
Monterey, California **93943**-5000

RADM Admiral David R. Ellison
Superintendent

R. Elster
Provost

This report was prepared for and funded by the Naval Postgraduate School Center for Information Systems Security (INFOSEC) Studies and Research (NPS CISR).

This report was prepared by:

Cynthia E. Irvine
Associate Professor

Timothy E. Levin
Research Associate Professor

George W. Dinolt
Associate Professor

Reviewed by:

Neil C. Rowe
Professor
Department of Computer Science

Christopher Eagle, Chair
Department of Computer Science

Released by:

D. W. Netzer
Associate Provost and
Dean of Research

# REPORT DOCUMENTATION PAGE

Form **approved**

OMB **No 0704-0188**



| 1. AGENCY USE ONLY (Leave blank) | 2. REPORT DATE<br>May 2002 | 3. REPORT TYPE AND DATES COVERED<br>Technical Report |
|---|---|---|

**4. TITLE AND SUBTITLE**

A National Trusted Computing Strategy

**5. FUNDING**

**6. AUTHOR(S)**

Cynthia E. Irvine, Timothy E. Levin and George W. Dinolt

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Center for Information Systems Security Studies and Research (NPS CISR)
Naval Postgraduate School, 833 Dyer Road, Monterey, CA 93943

**8. PERFORMING ORGANIZATION REPORT NUMBER**

NPS-CS-02-003

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release; distribution is unlimited

**12b. DISTRIBUTION CODE**

**13. ABSTRACT (Maximum 200 words.)**

Through neglect, the national capability to design and construct trusted computers and networks has begun to atrophy. Not only has the information infrastructure been built weakly, but also our capability to strengthen it continues to decline. The Nation is now lacking in both the research and development talent to produce trusted computing systems and the educational infrastructure to create this talent. The Center for INFOSEC Studies and Research (CISR) in Monterey, California, proposes a three-pronged approach to strengthen the national information infrastructure and reinvigorate the national capability to produce trustworthy computing systems. First, we describe our Trusted Computing Exemplar project as a ***worked example*** of how trusted computing systems and components can be constructed. Second, we define a national ***research initiative*** to advance the theoretical foundations for trusted computing and to produce a set of automated tools to support the development of high assurance systems; and third, we define an ***educational initiative*** based on nascent Information Assurance education programs and the Trusted Computing Exemplar to provide a framework for Trusted Computer Development education. The result of this multi-faceted approach will be to increase the security of the national Information Infrastructure by increasing the availability of: Trusted Computer systems and components, Trusted Computer development tools, and Trusted Computer developers, evaluators and educators.

**14. SUBJECT TERMS**

Information assurance, computer security, trusted computing, high assurance

**15. NUMBER OF PAGES**
13

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | Unclassified |

# A National Trusted Computing Strategy

Cynthia E. Irvine, Timothy E. Levin, George W. Dinolt

Center for Information Systems Security Studies and Research
Computer Science Department
Naval Postgraduate School
Monterey, California 93943

## Introduction

Entire sectors of the critical national infrastructure rely upon the vast systems of computers and networks that comprise our *Information Infrastructure*. The ongoing physical and economic well being of our country is based on the ability of this infrastructure to process data securely and reliably. Yet, daily we learn more and more about vulnerabilities in the networks, operating systems, and applications that form the very fabric of the Information Infrastructure. The nation is clearly at risk in its lack of readiness for cyber attacks.

In our zeal to capitalize on the explosive advances in commercial computer technology over the last 10 to 15 years, we have ignored the requirements to deploy computing systems with the ability to protect data according to its criticality and value to us. During this time, the US Government and Military focus on commercial *off the shelf* procurements helped to fuel the technology explosion, but it also contributed to the lack of advances in the ability of commercial systems to appropriately protect themselves and the data with which they are entrusted. While industry has been driven to supply the latest technology at the fastest pace, it has not been motivated, either internally or externally via customer demand, to produce trustworthy computing systems. Through neglect, the national capability to design and construct trusted computers and networks has begun to atrophy. Not only has the information infrastructure been built weakly, but also our capability to strengthen it continues to decline. The Nation is now lacking in both the research and development talent to produce trusted computing systems and the educational infrastructure to create this talent.

The Center for INFOSEC Studies and Research (CISR) in Monterey, California, proposes a three-pronged approach to strengthen the national information infrastructure and reinvigorate the national capability to produce trustworthy computing systems.

- First, we describe our Trusted Computing Exemplar project as a ***worked example*** of how trusted computing systems and components can be constructed.

- Second, we define a national ***research initiative*** to advance the theoretical foundations for trusted computing and to produce a set of automated tools to support the development of high assurance systems; and
- Third, we define **an *educational initiative*** based on nascent Information Assurance education programs and the Trusted Computing Exemplar to provide a framework for Trusted Computer Development education.

The result of this multi-faceted approach will be to increase the security of the national Information **Infrastructure** by increasing the availability of:

- Trusted Computer systems and components
- Trusted Computer development tools, and
- Trusted Computer developers, evaluators and educators

We begin with a more detailed description of the need for strong trusted computing and an overview of the degraded state of trusted computing in the United States.

## Current lack of Trusted Computing Infrastructure

Trusted computing is special. It addresses not only the problems of frontal attack to systems and malicious software, but also that of subversion: the equivalent of software "moles" in the system. Such "moles" can be trap doors or other inserted artifices. They can be triggered by conditions determined by the adversary and, when active, can be used to launch any attack the adversary desires. The safeguards required to protect systems from trap doors go beyond those required to protect either against frontal attack or Trojan Horses. Trusted computing encompasses the science and engineering required to specify, design, implement, and maintain components in which we have a high level of confidence against system subversion.

To protect against frontal attacks, systems must be designed and implemented without exploitable flaws and must be designed to constrain access to information and confine any damage resulting from the execution of malicious software. This approach acknowledges the mathematically proven limits of computability by accepting the fact that there is no way to automatically inspect all software to determine whether its behavior is benign. Instead most software is considered to be potentially malicious and its execution is circumscribed such that the effects of any malicious activity can be controlled and audited. To protect against trap doors requires an approach to system lifecycle that is structured so that systems can be subjected to analysis that demonstrates the absence of internal subversion. It is the proof of absence of unspecified functionality that distinguishes trusted computing from safety and other forms of high confidence computing.

The science and discipline of trusted computing has been neglected for well over a decade. For example, in one national laboratory where one might expect such expertise, scientists and engineers freely admit that those who have joined the field of computer and network security over the past **15** years are ignorant of the science and discipline that must be applied in the area of highly trusted computing.

This section is intended to provide an overview of the current state of trusted computing in the United States. Topics to be covered include: education, practitioners, tools, trusted systems and components, decline in **U.S.** capabilities, and observations regarding non-US capabilities. We will see that there is a strong interdependence between several of these areas and that weakness in one weakens others.

### Education

Colleges and universities produce the teachers and practitioners of scientific and technical disciplines. In traditional computer science departments, computer and network security have not been considered mainstream topics. **As** a consequence, senior faculty often lack an appreciation for this discipline and do not provide adequate encouragement to junior faculty or graduate students to pursue it. The recent surge in recognition of the need for improved computer and network security has attracted new faculty to the area. Unfortunately, these faculty are often unaware of the developments in trusted computing that have preceded them. Consequently they teach what they know and attempt to adapt it to the problem of computer security. The overall pathetic state of computer security in all sectors has resulted in a constant barrage of frontal, direct attacks on systems and their exploitation by a variety of viruses and simple malicious software instances. Thus, instead of addressing the

underlying problems of flawed design and inadequate partitioning of execution domains, much less the problem of proof of the absence of artifices and trapdoors, most current pedagogy focuses on superficial fixes. Students are taught to construct and configure systems in the hope that by creating a maze of weak defenses, adversaries will be thwarted. Substantial teaching and research effort are applied to the construction of adaptive, but superficial, protection systems; provision of security functions lacking concomitant assurances that the functions cannot be bypassed; attempts to solve theoretically undecideable problems; etc.

With a few notable exceptions, the entire approach to teaching trusted computing misses the mark. Students are unaware that systems can be constructed that do not present the flaws that are so easily exploited in direct attacks. They do not know how to construct systems that confine and control malicious software. They are not taught the techniques that must be used to ensure the absence of trapdoors and that do ensure that protections are correct and complete.

Today, even if faculty wanted to learn about the science, tools, development methods, and processes required for trusted system construction, most of the information they need is unavailable. Indeed there are papers in the research literature describing the theory of trusted computing and there are summaries providing an overview of the results of a number of trusted computing initiatives. But the reality is that much of the science and technology behind those early initiatives was folded into proprietary documents and may forever be unavailable to the academic community.

**Practitioners**

Two decades ago relatively large commercial teams were engaged in the development of strong trusted systems. It was a time when practitioners were learning the scientific and social processes required for the development. It was a time when that which worked in practice was distilled from a broad range of less effective techniques, a time when the first tools for trusted systems development were being created and used. It was also a time when the US Government encouraged vendors to develop trusted systems. Thus many of the practitioners of trusted computing were developing proprietary systems.

Commercial computing blossomed. Office automation proceeded with great rapidity, personal computers were becoming ubiquitous, and use of the Internet was expanding. Eventually, trade-offs began to be made during system acquisitions. Insecure systems were chosen rather than secure ones for a number of reasons. Government acquisitions ceased to specify the need for strong trusted computing, and vendors lost confidence in a process that had initially encouraged them to invest in development of strong trusted systems. A downward spiral in trusted computing began and continues to this day. Thus, as support of trusted computing waned, the core of scientists and engineers who could build highly trusted systems dispersed to find opportunities elsewhere.

Not only were developers impacted, but the lack of requests for highly trusted computing systems resulted in a dwindling need for those capable of conducting independent evaluations of such systems. The commercial laboratories that have since been established to conduct secure product evaluations are focused on the analysis of less trustworthy products.

**Tools**

A rigorous methodology is required to construct highly trusted systems. All non-trivial systems require the use of tools to support the development effort and to provide proofs and evaluation evidence regarding the absence of subversion.

The current availability of tools for trusted computing development programs is affected by two factors. First, many of the tools previously developed for trusted computing efforts have disappeared or have become obsolete as a result of the absence of platforms and operating systems upon which to run them. Second, because there has been, in effect, a 15-year hiatus in the area of trusted computing, little research and development has been applied to the improvement and extension of the available toolset.

Current activity in computer science and engineering that addresses assurance is focused on the correct function of systems and software: that the systems will function as specified. This has resulted in advances in system safety and reliability. Thus, where one can describe (viz., mathematically specify) the safety and/or reliability properties of a system, it is possible to provide various levels of assurance that these properties will be satisfied in specified environments. But these systems make one important assumption, that all of the users and developers want the system to do the right thing. They assume that system failures are a result of random errors in the external environment and not the result of malicious activities addressed to subvert the system. In computer security, we must address the specification, design, implementation, and maintenance of a system in which the assumption must be that there will be ongoing attempts to insert functionality into the system that is unspecified and will perform clandestinely on behalf of an adversary. The malicious activity can occur during any phase of the system with the goal to either modify the system during its construction (subversion) or to take advantage of weaknesses in the design, implementation, configuration or management of the system to subvert its intended security functionality. Thus, in trusted computing, we are faced with an ongoing malicious threat to the integrity of the system.

To address the problem of subversion, the insertion of unwanted functionality during construction, special tools are required to (1) describe the desired security properties of the system and (2) guarantee that the implementation of the system has these desired security properties and that the properties cannot be subverted. To achieve the first goal one needs languages to describe the security properties. To achieve the second, one needs to be able to mathematically ensure that the implementations, the architecture, the detailed design, the software, the management plan, including the hardware where practical, actually implement exactly the security properties described in part one and no more. In this way one can ensure that no subversion has taken place.

During the era of government support for strong trusted computing, there was research and development of tools to support the formal, mathematical demonstration of the absence of subversion. These tools supported description of security policies, specifications of designs, verification that the designs implemented security policies, and a mapping of the design to the code. There was limited support for "code" proofs.

In addition, tools were developed to support the connections between the formal descriptions of the system and the testing that the components and systems underwent. As

part of this testing and analysis, some work was done to identify and limit "covert channels," which could be used by a malicious insider to communicate, using system resources, with a colleague on the outside.

Various U.S. Government agencies supported the development of these tools. The National Computer Security Center maintained an Endorsed Tools List of formal verification systems for security modeling. These verification systems encompassed mathematical specification languages that allow correctness conditions to be expressed, and reasoning mechanisms that included functions for parsing specifications, verifying specification legality, and providing verified conclusions regarding the satisfaction of the correctness conditions. Each system was characterized by its rules and organization; features and functionality; the level of trust that could be ascribed to the verification system itself; and its documentation.

The hardware engineering community, for example Intel and to a lesser extent, AMD, have both invested heavily in the use of "formal methods" and have developed tools to assist in the analysis of complex chip designs and implementations. This work has been spurred on by the discovery that these methods are cost effective in the chip design process.

In contrast, the software development community has not made significant use of these tools; instead, they have tried to address the issues of software complexity and assurance by advocating various development "methodologies." Since software is relatively "easy" to update if bugs are found, there is no incentive to provide up-front correctness that has driven the use of formal methods in the hardware world, The assumption is that since software is "so complex," its properties cannot be specified and hence any attempts at formalisms are doomed to failure at the outset. The software development community is generally unaware of the "formal methods" successes of the past and hence may see specification and verification as being an additional cost burden, with no apparent gain.

Hence, in the United States, over the last 15 years, there has been little development of the mathematics, science, engineering, technology and tools that could be used to provide a high degree of assurance that software meets not only its safety and reliability goals, but also its security goals.

**Trusted Systems and Components**

Few strong trusted systems are currently available as commercial products and those that are available are inadequate for modem use. What happened to all of the government-sponsored projects to build highly trusted systems? There was a lack of appreciation on the part of decision makers of the value of trusted computing. As a result, although trusted systems were available in the late 1980s and early 1990s, few such systems were purchased. In fact, no major government program of record ever mandated the acquisition of trusted systems that provided strong confidence against the threat of subversion.

Because they were categorized as "munitions", strong trusted systems were subjected to export controls. These controls further limited the market for highly trusted commercial products and decreased the incentive for vendors to pursue the development of such products. In fact, one major vendor canceled the roll out of a much-anticipated highly trusted product after all development and analysis were completed. The artificially limited market and complexity associated with international sales were certainly factors working

against continued corporate investment in products for which high quality lifecycle management would be required.

**Decline in US Capability as Contrasted with non-US Capability**

If the United States were to mount an effort to construct a large highly trusted computing system today, there would be no sufficiently large or experienced commercial team to carry out the work. **A** vendor attempting such an effort would need to either assemble a team largely from personnel responsible for the early trusted computing work, or teach an entire new team all of the science and technology that was previously brought to bear. Because colleges and universities are not preparing students in the area of strong trusted computing, no recent graduates would be able to participate without a significant period devoted to additional education.

The consensus holds that confidence in the trust properties of a system must be provided by independent review and evaluation rather than the often hyperbolic and misleading claims of product vendors. Thus, in order to effectively deploy trusted computing systems, a national capability for their evaluation must be available. Evaluators must have the same range of scientific and technical background as those who construct highly trusted systems, thus they are faced with similar problems: any laboratory or agency charged with the independent evaluation of highly trusted systems.would have to reconstitute the necessary professional staff to conduct the work.

How do capabilities in the United States compare to those in the rest of the world? It is well known that educational rigor in the United States has declined and that **U.S.** high school and university students rate poorly with respect to mathematics and science students in other countries. This is reflected in the lack rigor applied to analysis and specification of our computer systems and, naturally, in trusted computing as well. The mindset is to throw the systems together and hope that they will run well enough so that flaws will not be a major impediment to the systems' use in the popular market.

Elsewhere, the requirements for careful analysis and formal rigor in the design and development of systems is better appreciated and practiced. Rather than rely on questionable commercially available systems, several research and development programs in other countries apply rigor to the area of trusted computing and have programs to construct their own computers. For example, in 2001 China had approximately 350 companies with over 500 products.[1] Just in the area of cryptography, a 2001 report gave a total of 1521 cryptographic products produced and developed in **76** countries.[2] These products included both commercial and end-user products in the areas of hardware, software, firmware, and combinations thereof. Germany alone produced a total of 118 of these products. Despite an active hacking culture, the scientists and engineers of the former Soviet Union have a deep appreciation of the foundations of and techniques for strong

---

[1] China's Computer Security Products Exceed 500, **9** Sept 2001, http://www 1.chinadaily.com.cn/itchina/2001-09-05/31182.html

[2] World Wide Survey of Cryptographic Products, **NAI** Labs, *6* Sept 2001, http://download.nai.com/products/media/pgp/pdf/NAI-Labs-Crypto-Survey-9-6-01.pdf

trusted computing. Their collaboration with other non-U.S. partners is likely to produce formidable results.

This leads to the question: if key elements of the information systems upon which critical national infrastructures rely must be constructed using strong trusted computing components, will the United States need to depend upon foreign sources and foreign evaluations for those components? What level of comfort would we have, with guarantees from those who may be fair weather friends, that these components are free from trap doors and other mechanisms that could be controlled by unfriendly adversaries?

**Summary**

Less than a decade ago, the United States possessed the most advanced capabilities in the world for the construction of highly trusted computing components. Today, that capability is rapidly dwindling and within the next five to ten years will have disappeared altogether. The National Strategy for Trusted Computing must not permit our capability in the area of strong trusted computing to be lost; instead that capability must be revitalized.

The remainder of this paper will present a strategy for reviving strong trusted computing. It includes system development, education, and research. To be successful, the program will require courage and confidence. Courage to move forward despite the claims of those who would ignore the fragility of the infrastructure due to inadequate trusted computing; and confidence to build and deploy strong trusted computing products where they are desperately needed.

## Strategy for Developing a National Trusted Computing Capability

In this section we present three interrelated initiatives for developing a National trusted computing capability. The first initiative, our Trusted Computing Exemplar project, provides a focal point for defining *research* and *education* initiatives and a test bed for interpreting research results.

### Trusted Computing Exemplar Project

The purpose of the Trusted Computing Exemplar project proposed by CISR is to provide a worked example of how trusted computing systems and components can be constructed. The project has four activities:

- Create a framework for rapid high assurance system development
- Develop a reference trusted computing component
- Evaluate the component for high assurance
- Disseminate related deliverables via open methodology

**A** trusted computing *component* will be developed utilizing the high assurance development *framework.* A third party evaluation of the component will commence during development (e.g., once the high-level design documentation is written). The documentation, source code, and other deliverables will be made openly available as they are produced. Co-located teams composed of a combination of seasoned trusted computing veterans and uninitiated "apprentices" will perform these activities.

The combination of open methodology applied to all project documents and deliverables and the mentoring of project apprentices will help provide for transfer of trusted computing technical know-how to a new generation of trusted computing developers (see also "Educational Initiative," below). Furthermore, the public availability of the high assurance development framework and the reference trusted computing component will provide technology transfer of key enabling technologies to the commercial, government, and open-source communities.

Framework for Rapid High Assurance Development

The framework for rapid high assurance development will provide a set of interoperable tools and define a set of efficient, repeatable procedures for constructing trusted computing systems and components.

The toolset will provide support for automated management of high assurance development throughout a product's lifecycle. The toolset will support:

- Configuration management of the developed software, tools and processes
- Specification of the security properties and the design
- Verification that the design meets the security properties
- Code development
- Verification that the code implements the design (and only the design)
- Specification based testing
- Teamwork and training support

The tools and procedures will also support the open dissemination of project deliverables using a philosophy and mechanisms similar to the "open source" approaches.

We expect that the development framework for the Trusted Computing Exemplar project will be scaled to the size of the project. It will reveal those aspects of such frameworks that must be handcrafted to meet specific project requirements. Generalization of this framework to support different target technologies and larger projects is a research topic (see "Research Initiative," below).

Trusted Computing Reference Component

We will develop a small-scale high assurance trusted computing component as a reference implementation. Candidates for implementation include a high-assurance, embedded micro kernel to support partitioning of users and information, and a high integrity public library for web applications. Because the product as well as the process will be showpieces for trusted computing development, high assurance methodologies and techniques will be applied during the entire lifecycle (viz., design, implementation, distribution, and maintenance phases). The goal is to produce a very small, portable component that will take advantage of modern hardware support, where applicable, and that will provide users with an *a priori* assurance against system subversion.

Reference Component Evaluation

As noted above, independent evaluation is required to ensure confidence in the assurance claims made for a trusted component. The reference component will be subjected to a third-party evaluation to ensure that it provides security with high assurance. There are two options for the evaluation: the Common Criteria and the Trusted Computer Evaluation Criteria (TCSEC). With the Common Criteria, products are evaluated with respect to a "protection profile." There do not currently exist any generally accepted protection profiles for high assurance systems or components. Therefore, if the Common Criteria were to be used as part of the Trusted Computing Exemplar Project, a high assurance protection profile would need to be developed, first. The TCSEC is a set of criteria that preceded the Common Criteria, and that includes specifications for high assurance products (viz., classes B3 and A1). While evaluation according to the provisions of the TCSEC is perhaps more predictable as a result of historical precedent, the TCSEC is not an active document and the results might be considered old fashioned.

Open thodolog

Utilizing the open methodology tools and procedures developed in the High Assurance Development Framework (see above), the deliverables and outputs of the Trusted Computing Exemplar Project will be made available to the public. This open methodology will take advantage of the approaches used by the various "open source" movements to provide a continuous set of mechanisms for contribution, evaluation and distribution of the various parts of the science and technology. As part of the open methodology activity, the development framework will be documented, including methods, techniques, and social model, and distributed in an open web-based format. Other deliverables to be made available include source code, project plans, and evaluation evidence and reports. Thus, by making available these internal documents, the project will provide previously unavailable examples of how-to for high assurance trusted computing.

**Research Initiative**

The National research initiative defines a set of research activities to ensure that the nation possesses necessary and sufficient national capacity to develop and maintain highly trusted computing systems. **A** key effect of this work will be to reduce dependence of the national Information Infrastructure on high maintenance security mechanisms that are provably insufficient, such as firewalls and intrusion detection systems.

The research initiative has two primary thrusts. The first is the development of theory that can be used to both describe the security properties of components and systems and to show how these properties "compose" to form larger units. Some work has been done in this area but there are remarkably few results that have proved useful. The second primary thrust is the development of tools that can be used to automatically reason about the component properties and the software that is used to build the components. There are a number of methodologies that purport to support this and several companies make tools that support the methodologies, but the commercial methodologies are not based on a firm theoretical foundation so that it is not possible to determine whether the outputs of the methodology and the software or systems developed from these outputs have the desired properties. The rest of this section provides discussion regarding specific issues and activities of the research initiative.

Modular composition is central to the construction of complex computing systems. The advantages of the principles of modularity apply, whether constructing monolithic or distributed systems. Yet, work needs to be done to provide a clear scientific underpinning for understanding which type of compositions will result in secure or more secure systems. For example, the concepts of *defense in depth* and *construction of trusted systems from untrusted components* need to be clarified. Precisely when do these concepts apply, and how is one to assess the result; and conversely, exactly when do these concepts not apply? How do we know when a *depth of defenses* is better than a simple, unified mechanism? What are the principles or methodologies under which components can be composed to yield a level of trust in their group behavior that is greater than our trust in the behavior of the individual components? Similar results from the field of fault tolerance may be encouraging at an anecdotal level, but we lack a scientific basis for their applicability to security.

Modular composition for one particular type of security property, "non-interference," has been the subject of past investigations, but there remains a research challenge to determine whether the results can be useful in the construction of trusted computing systems. Also, research is needed to determine whether any of the analytical methods utilized for non-interference are applicable to understanding the composition of other, more general types of security policies. If non-interference based composition proves to be too narrow - or worse, useless - then alternatives must be investigated.

Another question regards the composition of application-enforced policies together with infrastructure-based policies (e.g., policies enforced by the network fabric, hardware or operating systems). For example, to what degree is it possible for an application to enforce its own policy regarding entities it creates, if the application's behavior depends on a weak infrastructure? We need a scientific understanding of the requirements on the infrastructure for effective application-enforced security policies, so that we know how to construct such compositions, and know when they can be considered to be effective.

Regarding the construction of "hybrid-assurance" systems out of components with disparate levels of trust, research is needed to understand the precise circumstances under which low assurance components can contribute (e.g., to enforcement of security policy). Data integrity is a related issue; for example, under what arrangements can low assurance components be trusted to handle or manage high integrity data? For what purposes would a Trusted System be useful if it did not have the capability to protect the integrity of data entrusted to it?

The engineering principles for construction of monolithic systems, such as layering, modularity and the role of global variables have a much longer history than those for more recent distributed and object-based systems. More basic research needs to be applied to understanding how these and other principles for system building can be applied to secure distributed component constructs. For example, what should be the role of layering in the prevention of functional interdependencies between distributed modules? Can the concept of ***protection domains*** help in the design of secure distributed control channels? Could system 'securityproperties be enhanced if distributed modules are interconnected such that each module only depends on modules of equal or higher assurance?

**Our** ability to understand the effectiveness of trusted computing systems and our approaches for constructing them, such **as** modular composition, is limited by our lack of ability to measure security or trust. How do we know how much security is provided by a system? How can the level of trust be specified?

How do we measure the overall behavioral properties of a set of remote, interconnected modules? What security properties are preserved or lost when a module is interconnected with a weaker module? Similarly, we lack a theoretical basis for building efficient and integrated tools for analyzing trusted computing systems and architectures. To arrive at a high degree of trust that an architecture, system or component functions securely, formal verification may be required. We lack a theory for how to apply formal verification to complex distributed composite systems. Also, given a system of components, each of which have been formally verified, can the verification results themselves be composed to achieve or assist in the formal verification of the system? And finally, the automatic derivation of efficient code from high-level security specifications continues to elude us.

Research is required to provide automated development environments and tools to manage the rapid creation of high assurance trusted computing systems. These environments need to be generalizable to support the production of a wide range of systems and components, so that they are applicable, for example, for constructing high assurance secure networks, applications and operating systems, **as** well as trusted embedded components and specialized high assurance interfaces.

Tools are needed for automating all aspects of the trusted computing development process, including design, specification, testing, analysis, verification and configuration management. We need to develop tools that will automatically (mechanically) reason about the properties of systems. The tools should be usable by a large class of system developers. Systems developers should be able to specify "high level" properties of systems that can be checked for proper implementation at each stage of development, integration, testing and deployment. The tools need to be scalable, such that they can be used for construction of large as well **as** small systems, and they need to be interoperable, so that the different tools

work with each other (e.g., compilers and configuration management systems) and are integrated with the system development and project management processes.

**Education Initiative**

The purpose of the proposed Education Initiative is two-fold. The first purpose is to increase the national ***capacity to create trusted computing systems*** by educating developers and evaluators in trusted computing development science, methods, and techniques. The second, is to increase the national ***capacity to create developers in trusted computing*** by educating a new generation of teachers in the science, methods and techniques of trusted computing development. Our ulterior goal is to increase the security of the national Information Infrastructure. The primary determinant of this security is to have trusted architectures, systems and components deployed in the infrastructure, as required for security. The "first derivative" or limiting factor of this determinant is the ***availability of developers*** who have the requisite skills to create trusted computing systems and components; the "second derivative" is the ***availability of teachers*** to create new trusted computing developers. Our Education Initiative addresses these limiting factors (first and second derivatives) to increasing the security of the national Information Infrastructure.

The trusted computing education initiative will be based on existing Information Assurance education programs and the previously defined Trusted Computing Exemplar (see above). The National Security Agency has recognized several Centers of Academic Excellence in Information Assurance Education (CAEIAEs). For the most part, these centers are focused on providing education and training regarding issues for information assurance at the user, certifier and administrator levels, but they are not generally oriented to teaching the art and science of trusted computing system ***development.*** We propose establishment of a ***trusted computing-development curricula and teacher education program*** directed to CAEIAEs and other educational institutions with computer security interests. This program will provide educational materials for enhancing curricula with trusted computing development topics, as well as courses for teachers in how to teach trusted computing development. All of these materials and courses will utilize materials produced by the Trusted Computing Exemplar, for example, formal and detailed design specifications, code modules, and project management documentation.

To educate new trusted computing developers directly will require the creation of new undergraduate, masters, and Ph.D. courses specifically focused on the art and science of producing high assurance trusted computing systems. Some materials and content for these classes can be adapted from existing computer security courses that are taught at the Naval Postgraduate School, such as "Secure Systems" and "Formal Methods." Other materials will be based on documentation, code and other artifacts from the Trusted Computing Exemplar.

**Summary**

In this section, we have described a holistic approach to reversing the national decline in the area of trusted computing. It includes our trusted computing exemplar project, a national research initiative, and an educational initiative that will build upon the exemplar project and research results.

## INITIAL DISTRIBUTION LIST

| | | |
|---|---|---|
| 1. | Defense Technical Information Center<br>8725 John J. Kingman Rd., STE 0944<br>Ft. Belvoir, VA 22060-6218 | 2 |
| 2. | Dudley Knox Library, Code 013<br>Naval Postgraduate School<br>Monterey, CA 93943-5100 | 2 |
| 3. | Research Office, Code 09<br>Naval Postgraduate School<br>Monterey, CA 93943-5138 | 1 |
| 4. | Dr. Cynthia E. Irvine<br>Code CS/Ic<br>Department of Computer Science<br>Naval Postgraduate School<br>Monterey, CA 93943-5118 | 10 |
| 5. | Mr. Timothy E. Levin<br>Code CS/Lt<br>Department of Computer Science<br>Naval Postgraduate School<br>Monterey, CA 93943-5118 | 2 |
| 6. | Dr. George W. Dinolt<br>Code CS/Dg<br>Department of Computer Science<br>Naval Postgraduate School<br>Monterey, CA 93943-5118 | 2 |