COMPOSING A HIGH-ASSURANCE INFRASTRUCTURE OUT OF TCB COMPONENTS

Mark R. Heckman
AESEC Global Services, Inc.
mark.heckman@aesec.com

Roger R. Schell
AESEC Global Services, Inc.
roger.schell@aesec.com

Edwards E. Reed
AESEC Global Services, Inc.
ed.reed@aesec.com

ABSTRACT

U.S. Government agencies and major vendors are actively attempting to secure critical infrastructure networks, but those efforts depend on patching unsecure, commodity systems, installing add-on security appliances, and applying other industry "best practices" that are ineffective against new attacks and software subversion. This has unfortunately led to the conclusion that it is impossible to secure critical infrastructure networks and even that a completely new, "alternative" Internet is needed. These conclusions disregard known and proven techniques for building secure, high-assurance, trusted systems – techniques developed as a result of years of research and engineering experience and systematically codified in the Trusted Computer System Evaluation Criteria (TCSEC) and related documents. Those techniques have not since been improved upon or adequately replaced, not even by the more recent Common Criteria for Information Technology Security Evaluation. In this paper, we sketch how the trusted systems technology codified in the TCSEC can be applied today to create a secure infrastructure network.

Keywords

Infrastructure Protection, Trusted Computing Base, TCB, GEMSOS, GemSeal, Security Kernel, High-assurance, Class A1, TCSEC, TNI, Crypto-sealing. Common Criteria, Software subversion

1. INTRODUCTION

Growing awareness that terrorists or other adversaries could harm the U.S. through cyber attacks on national infrastructure systems has resulted in increased attention paid to protecting those systems. The recent and widely reported Stuxnet worm demonstrated that those concerns are well-founded [13].

Suggested approaches for securing critical infrastructure from cyber attack often focus on applying information technology (IT) industry "best practices" [14]. Traditional IT security, however, may not always be a good fit for infrastructure security. Patching and frequent updates, for example, a staple of IT security, is difficult and risky in infrastructure systems [6].

Moreover, traditional IT security practices are basically an arms race that can't be won, because attackers can find unknown flaws in low-assurance systems and develop attacks for them faster than defenders can find and patch them [21]. Poor software development and distribution practices, furthermore, create the opportunity to exploit artifices previously put in place through software subversion [3]. Stuxnet, for example, took advantage of known but unpatched flaws and "zero-day", previously unknown flaws in Windows for which no patches yet existed in order to subvert the operating system [12].

Traditional IT security best practices alone are insufficiently effective for protecting critical systems when the underlying systems themselves were neither designed, engineered, nor evaluated to be secure. The Stuxnet attack is illustrative in that it happened years after other successful attacks on Windows systems used in infrastructure [18] [16], demonstrating the enduring vulnerability of the Windows platform. Yet, Windows and other low-assurance alternatives, like Linux, continue to be used in critical infrastructure systems.

Recognizing this problem, the U.S. Department of Homeland Security has created a "Software Assurance Program" to promote the development of high-assurance software for infrastructure [11]. The standards and practices described, however, are neither mandated nor part of a formal development and evaluation process. Without a formal development and evaluation process, there can be little assurance about the correctness of a system and its software. Furthermore, if software developed using such a "software assurance program" is then run on a low-assurance platform like Windows or Linux, the software provides no meaningful assurance whatsoever, particularly in the face of (even moderately determined) adversaries likely to employ subversion as their mode of attack.

At the same time, the U.S. Department of Energy has created the Open PCS (process control system) Security Architecture for Interoperable Design (OPSAID) and the Lemnos Interoperable Security programs. The goal of OPSAID is to help vendors build add-on security devices for existing infrastructure [9]. The Lemnos program is intended to create standard metrics for describing the functions of network security devices and for evaluating their performance [10]. Both programs, however, represent an ad hoc, piecemeal approach to improving the security of critical infrastructure rather than a well

thought-out, high-assurance solution. And the tools developed under these programs are themselves almost certain to be built on low-assurance platforms, thereby increasing rather than decreasing the domain of vulnerabilities within critical infrastructure networks.

At the other end of the protection spectrum is a recent proposal by an assistant director of the FBI to develop an entirely new, separate, "secure alternative" Internet [17]. The expressed justification is that no system will ever be secure enough to defend against new attacks. The proposed solution is to set up another Internet in which access controls and monitoring would be strict, making it the analog of a "gated community". Leaving aside the likelihood that the "secure alternative" Internet would inherit many of the flaws and vulnerabilities of the existing Internet, the justification for this solution shares with the OPSAID and Lemnos programs the assumption that component systems are and always will be unsecurable, and that they must be tightly wrapped with layers of compensating controls to protect them.

Many years of science and engineering experience, however, have shown that we can build highly secure systems [4]. The techniques developed were systematically codified in the U.S. National Security Agency's "Trusted Computer System Evaluation Criteria" (TCSEC, also known as the "Orange Book") [8], and the potential to apply these techniques was largely carried forward in the more recent "Common Criteria for Information Technology Security Evaluation" (CC) [7]. A rigorous method of composing high-assurance networks out of high-assurance components was presented in the "Trusted Network Interpretation" (TNI, also known as the "Red Book") of the TCSEC [22].

An infrastructure composed of verifiable, high-assurance system components to enforce critical policy components, instead of low-assurance Windows and Linux systems, would be much less vulnerable to attacks – known or unknown. The composition of high-assurance components would provide the necessary assurance for a critical infrastructure network as a whole. This approach offers other advantages, as well:

- Unlike the ad hoc OPSAID/Lemnos "add-on" approach, a composed, high-assurance network offers a well thought-out and systematically applicable approach for securing infrastructure.
- Unlike the alternative Internet approach, which requires all parts to be working before the whole can work (a.k.a. the "Big Bang"), this compositional approach can be incrementally added into existing infrastructure networks and provide a high-assurance layer on which lower-assurance components could be used.

In this paper we present an approach to creating high-assurance critical infrastructure networks through applying the science of knowing how to build high-assurance components and how to compose them, building on the verifiable trusted systems technology that was originally codified in the TCSEC and TNI.

2. TCSEC/TNI Verifiable Protection

Our approach here is based on applying the verifiable protection technology codified in the TCSEC and TNI, not the CC. The CC together with an appropriate protection profile could potentially provide the necessary criteria and evaluation framework. Currently, however, the CC has neither an analog to the TNI to provide systematic guidance for composing a secure network of high-assurance systems, nor a published protection profile equivalent to the TCSEC's Class A1 level that would permit application of the TNI to compose high-assurance components evaluated under the CC.[1]

The TNI interprets the TCSEC in several ways. For one thing, while the TCSEC emphasizes secrecy policies and controlling the ability of users to read information, the TNI points out that the TCSEC definition of policy also encompasses integrity policies and controlling the ability of users to modify information. We presume in this paper that the chief concern when securing critical infrastructure is protecting it from tampering – i.e., an integrity policy.

The TNI's main focus is to interpret the TCSEC, without adding any new requirements or criteria, to explain how the TCSEC's requirements and criteria are directly applicable to trusted networks, using the concept of a partitioned trusted computing base (TCB). A TCB is "the totality of protection mechanisms within a computer system – including hardware, firmware, and software – the combination of which is responsible for enforcing a security policy" [8]. The TNI interprets the TCB concept for a Network TCB (NTCB) that is composed of TCB components [22].

A key element of the TNI (in particular for its "Class A1") is that an NTCB can be shown to have high-assurance with respect to a network security policy if it can be shown to be a sound composition of trusted elements. Thus, the network architecture must provide "a clean decomposition of an overall network security policy into policies for the individual components" [22]. The individual components can be separately evaluated and their composition shown to satisfy the NTCB policy.

[1] Readers interested in learning about other ways that the CC does not carry forward some of the lessons of the TCSEC and its "rainbow series" of guidelines and interpretations are directed to [5], [20].

Because the NTCB is a network of secure components, it is axiomatic that communications channels between the components must implement a trusted network service that preserves the security of the information they carry, including maintaining the integrity of sensitivity labels, user identifiers and clearances, and referenced object identifiers. The formal top level specification of an NTCB must include representations of the trusted network service specifications [22].

3. Example Applications

We present here two examples of how TCSEC/TNI concepts (whether articulated by the TSCEC per se, by an equivalent CC protection profile, or by some other criteria) can be applied to create a secure critical infrastructure network:

1. To create a secure infrastructure communications system that provides high-assurance, high-integrity communication.
2. To create secure behavior for applications through partitioning functions and constraining them using the TCB's mandatory controls.

For specificity and concreteness, we use as our base system in both examples the commercial product known as the Gemini Secure Operating System (GEMSOS). The U.S. National Security Agency (NSA) previously evaluated the GEMSOS security kernel and ratings maintenance phase (RAMP) at Class A1 as part of the Gemini Trusted Network Processor (GTNP) [15]. Sensitivity labels in GEMSOS include both secrecy and integrity components. GEMSOS was developed as a high-assurance, real-time operating system and is commercially available today as an OEM product.

3.1 Secure Infrastructure Communications

In the first example, subversion-resistant guards built using GEMSOS (called "GemSeal" guards [1]) sit on the network in front of each existing component (controllers and edge clients). The guards cryptographically seal packets sent between controllers and edge clients with a high-integrity label for their source. The guards forward each labeled packet across an untrusted network to a guard at the destination. Destination guards validate the data and label of each packet against the destination label before releasing it. Unlabeled or altered packets cannot enter the destination because they will not have a crypto seal that binds a label to a matching destination label.

This architecture is shown in figure 1, where "high-integrity packets" are packets that are part of legitimate communication between "high-integrity" infrastructure components, while "low-integrity" packets are injected packets that are not legitimate communication.

The TCSEC requires that "Sensitivity labels shall accurately represent security levels of the specific … objects with which they are associated. When exported by the TCB, sensitivity labels shall accurately and unambiguously represent the internal labels and shall be associated with the information being exported [8]." GEMSOS uses crypto seals internal to its TCB to protect the label and data integrity of non-volatile storage. GemSeal applies this same crypto seal concept to network packets forwarded by guards to ensure that packet data is not altered and that the source sensitivity label is authentic.

The seal is a Message Authentication Code (MAC) created by using the Cipher-Block-Chaining (CBC) mode of a symmetric encryption operation. Packet contents and

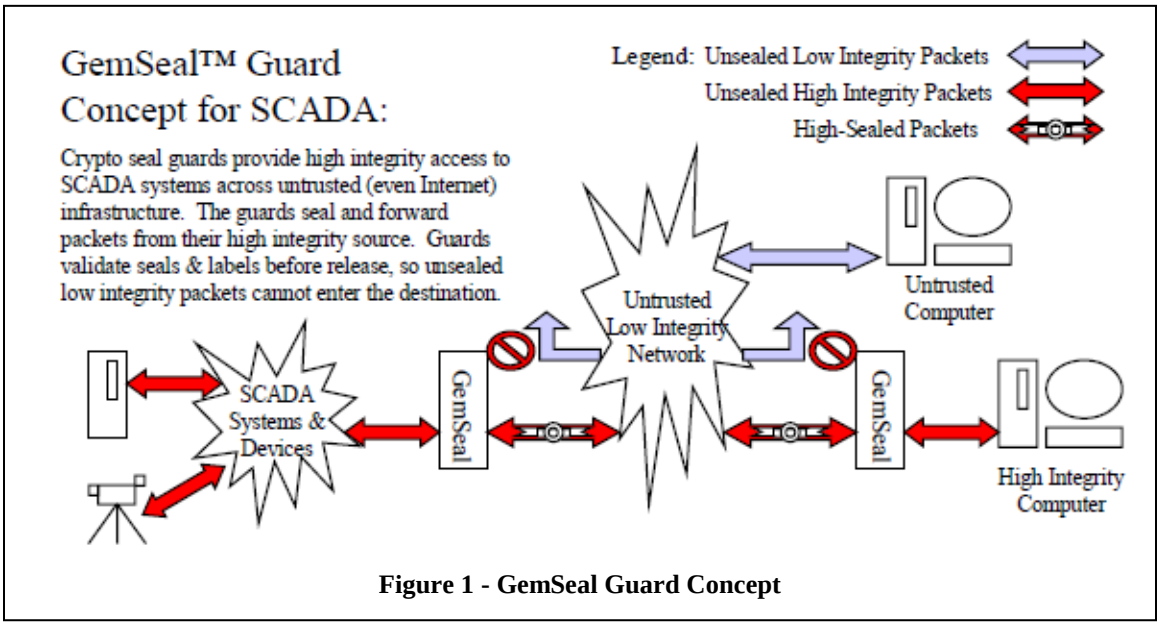

Figure 1 - GemSeal Guard Concept

the canonical representation of the source network sensitivity level are included in the CBC computation of the seal. The seal is the final encryption block of the CBC-mode encryption of the packet source-network sensitivity label (canonical representation) and contents of the packet, using a packet-specific initialization vector (IV) and the configured sensitivity level secret key.

The transmitted packet includes the forwarded packet as well as the seal. The label need not be transmitted as part of the packet, but is established for each security association (network-to-network connection) between GemSeal guards.

The GemSeal design makes substantial use of previously evaluated security services provided by the GEMSOS security kernel to minimize the amount of new trusted code (to several hundred lines). GemSeal accesses previously evaluated GEMSOS security services by way of published and stable APIs. The vast majority of GemSeal application code (including the network protocol stack) is untrusted; only two new security services need be trusted – "Seal Packet" and "Release Seal-Validated Packet". The previously evaluated GEMSOS protection ring mechanism protects these trusted functions from applications.

NSA deployed the GEMSOS kernel for key management and distribution in their Class A1 BLACKER project to implement host-to-host secure communications across the Defense Data Network [23], an application with significant similarities to the GemSeal guard concept. Like critical infrastructure networks, the operationally deployed BLACKER system required protection from particularly determined adversaries, so a major focus of the design was to address the threat of software subversion. That requirement necessitated the verifiable protection of the TCSEC's Class A1, which substantially deals with the threat of subversion of a system's security mechanisms [19].

Aesec has developed a proof of concept application of GEMSOS to SCADA systems using GemSeal guards to connect devices across an Internet-technology network. The proof of concept uses a pre-production update of the GEMSOS security kernel derived from the Class A1 GTNP [2].

The Department of Energy recognizes that a secure communications system is essential for securing critical infrastructure systems and, to address this need, has specified a VPN tunnel as part of the Lemnos Interoperable Security program [10], but the VPN appliances are not necessarily high-assurance, nor can they be used as the basis for verifiable protection for a high-assurance network as defined in the TNI.

GemSeal guards can be built to satisfy the interoperability requirements of the Lemnos program, but an important difference is that GemSeal guards are built on a high-assurance TCB, so the guards themselves are high-assurance and implement a mandatory security policy. Moreover, by implementing a secure communications channel, the guards satisfy a requirement under the TNI for building an evaluable NTCB.

The OEM nature of GEMSOS means that builders of diverse infrastructure components can maintain Class A1 security while porting GEMSOS to other, unique IA-32 hardware devices. The TCSEC (but not the CC [5]) supports the ratings maintenance phase (RAMP) process to support the reevaluation of evaluated systems when they upgrade to new hardware or when selected internal modules are changed. It is expensive and time-consuming to evaluate a high-assurance system. The RAMP process can dramatically reduce the time for a reevaluation to months or weeks [20]. GEMSOS could potentially be ported, for example, to a newer Intel processor without changing the TCB's formal top level specification or changing its modularity definition, which gives a high degree of confidence that it would still satisfy the Class A1 requirements when undergoing a RAMP.

3.2 Partitioned, Constrained Applications

In the second example, shown in figure 2, an infrastructure application is built on the GEMSOS TCB using a POSIX-compatible API. Instead of a monolithic application that mixes low-integrity functions with high-integrity functions, in this concept, the application is partitioned into several parts to take advantage of the TCB's mandatory security controls:

1. A "Controller" application manages critical system functions. Data sent by the controller to (and received from) edge components must be protected from accidental or deliberate contamination by other applications.
2. A Human-Machine Interface, "HMI" application is responsible for sending complete, detailed system status data to, and receiving operator commands from, a workstation (also running on a TCB), possibly over a local area network. This is obviously also a high-integrity application, yet it must have a different sensitivity label from the Controller application. The Controller is focused on managing the controlled process (e.g., a nuclear power station); presenting data to operators and receiving commands is only one part of its job. Moreover, because it is responding to rapidly changing conditions and real-time events, it must mediate, interpret, and apply commands sent by human – i.e., slow – operators, based on the current situation. The HMI may itself be partitioned into two parts: input and output.

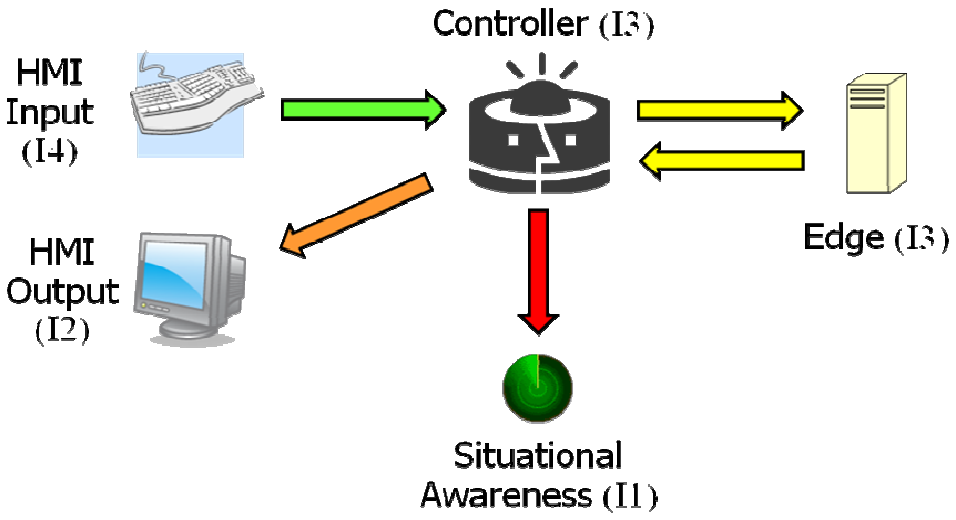

Figure 2 - Partitioned Application

3. A "Status" application collects and distributes low-integrity situational awareness information, such as system reliability statistics, possibly over the Internet, to a central headquarters. The TCB securely isolates each application and connection so that the untrusted, "summary" data cannot contaminate the higher-integrity HMI and Controller data.

In the figure, integrity levels are labeled I1 through I4, where I4>I3>I2>I1. Human input (for example, to shutdown the system in an emergency situation) has the highest integrity level. Communication between the controller and the edge components has the next highest integrity level. Human-readable output, based on information from the controller, has a lower level integrity, while status information has the lowest integrity level.

The implementation of the partitioned application on the GEMSOS TCB is depicted in figure 3. The different layers shown in the figure represent the GEMSOS protection ring mechanism. Code in lower, more trusted rings, cannot be bypassed by, and are protected from, untrusted but more feature-rich applications. The vertical "silos" denoted by the dotted lines represent security "domains" differentiated by mandatory security secrecy and integrity labels.

Each silo in figure 3 represents a different sensitivity level. The applications in each silo communicate with applications in other silos and with the outside world through GemSeal. This architecture is a fundamentally new approach that is not found in any of the widely-discussed proposed or deployed "best practice" SCADA implementations, none of which have the high-assurance enforcement of separation and sharing policies for both confidentiality and integrity afforded by technology that satisfies Class A1 requirements.

The mandatory policies implemented by the GEMSOS TCB support the TNI requirement that every component contains a component reference monitor that enforces part of the network access control policy. Combined with the secure communications implemented by GemSeal, an infrastructure network built using these components could satisfy a Class A1 evaluation under the TCSEC/TNI (or equivalent criteria).

4. CONCLUSION

U.S. Government agencies and their vendors are actively attempting to secure critical infrastructure networks. Surveying the futility of current efforts to secure networks using unsecure, commodity operating systems, add-on security appliances, and other industry "best practices", they have seemingly despaired of ever being able to adequately secure those networks, leading to the conclusion that threats will always outpace the ability of defenders to secure the networks.

But this conclusion reflects a needless rush to judgment. It ignores known techniques for developing secure, high-assurance systems – techniques created after years of research and engineering experience and codified in the TCSEC and related documents – that demonstrate that it is possible to build systems that verifiably protect against unknown attacks, including subversion.

In this paper, we've sketched how the TCSEC and TNI can be applied to create a secure infrastructure network.

We are not advocating for the TCSEC and TNI, per se, but for applying the science and technology of knowing how to build secure components and how to compose them that the TCSEC/TNI encapsulate. In this effort, for example, one could potentially also use the CC with a protection profile equivalent to the TCSEC/TNI.

Above all, we are not proposing in this paper a solution to the political problem of who will actually run and manage the evaluation process.

5. REFERENCES

[1] Aesec Global Services. "GemSeal Guard: High Assurance MLS." Unpublished white paper, 2007.

[2] Aesec Global Services. "GemSeal Guard: High Assurance Integrity for SCADA." White paper, 2007. http://aesec.com/guards/Aesec-GemSeal-SCADA-Concept-070220.pdf (accessed August 21, 2011)

[3] Anderson, E. A., Irvine, C. E., and Schell, R. R., "Subversion as a threat in information warfare," in Journal of Information Warfare, Volume 3, No.2, June 2004, pp. 52-65.

[4] Bell, D. E. “Looking back at the Bell-LaPadula model”. Proceedings of the 21st Annual Computer Security Applications Conference, December 2005.

[5] Bell, D.E., “Looking Back: Addendum,” Invited paper at the 22nd Annual Computer Security Applications Conference, December 2006. http://selfless-security.offthisweek.com/presentations/Bell_LBA.pdf (accessed November 20, 2011)

[6] Cárdenas, A. A., Amin, S., and Shankar, S. "Research challenges for the security of control systems". Proceedings of the 3rd Conference on Hot Topics in Security (HOTSEC'08). Berkeley, CA, USA: USENIX Association, 2008.

[7] Common Criteria for Information Technology Security Evaluation, Version 3.1, CCMB-2009-07-001, July 2009. http://www.commoncriteriaportal.org/cc/

[8] "Department of Defense Trusted Computer System Evaluation Criteria”. (Orange Book) 5200.28-STD, United States National Computer Security Center, December 1985.

[9] Department of Energy, “Open PCS Security Architecture for Interoperable Design (OPSAID)”. http://energy.gov/oe/downloads/open-pcs-security-architecture-interoperable-design-opsaid (accessed November 12, 2011).

[10] Department of Energy, “Lemnos Interoperable Security”. http://energy.gov/sites/prod/files/oeprod/DocumentsandMedia/Lemnos_Interoperable_Security.pdf (accessed October 5, 2011).

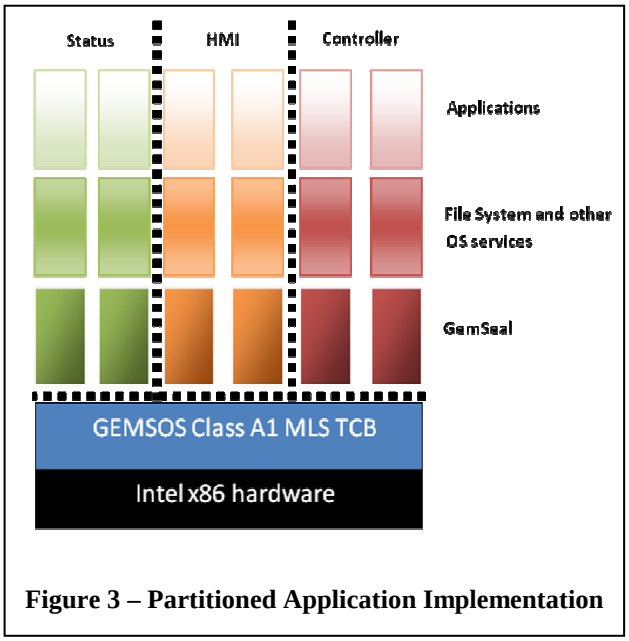

Figure 3 – Partitioned Application Implementation

[11] Department of Homeland Security, “Build Security In: Setting a Higher Standard for Software Assurance”. https://buildsecurityin.us-cert.gov/bsi/home.html (accessed September 6, 2011).

[12] Falliere, N., O Murchu, L., and Chien, E. "W32.Stuxnet Dossier." symantec.com. February 2011. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf (accessed August 21, 2011).

[13] McMillan, R. Siemens: Stuxnet worm hit industrial systems. September 14, 2010. https://www.computerworld.com/s/article/9185419/Siemens_Stuxnet_worm_hit_industrial_systems?taxonomyName=Network+Security&taxonomyId=142 (accessed August 30, 2011).

[14] Naedele, M. "Addressing IT Security for Critical Control Systems." Proceedings, 40th Hawaii International Conference on Systems Science (HICSS-40 2007). Waikoloa, Big Island, HI, USA: IEEE Computer Society, 2007.

[15] National Computer Security Center. "Final Evaluation Report for the Gemini Trusted Network Processor." 1995. http://aesec.com/eval/NCSC-FER-94-008.pdf (accessed August 27, 2011).

[16] Niland, M. Computer Virus Brings Down Train Signals. August 20, 2003.

http://www.informationweek.com/news/13100807 (accessed August 28, 2011).

[17] Rashid, F. Y. “FBI Official Backs Alternative Internet to Secure Critical Systems”. eWeek, October 23, 2011. http://www.eweek.com/c/a/Security/FBI-Official-Backs-Alternative-Internet-to-Secure-Critical-Systems-620446/ (accessed November 12, 2011)

[18] Roberts, P. F. Zotob, PnP Worms Slam 13 DaimlerChrysler Plants. August 18, 2005. http://www.eweek.com/c/a/Security/Zotob-PnP-Worms-Slam-13-DaimlerChrysler-Plants/ (accessed August 23, 2011).

[19] Schell, R. R., Brinkley, D. L., “Evaluation criteria for trusted systems”, in Information Security: An Integrated Collection of Essays, ed. Abrams and Jajodia and Podell, IEEE Computer Society Press, Los Alamitos, CA, pp. 137-159, 1995.

[20] Schell, R. R., Reed, E. E. “Computer Security: A Historical Perspective”. In Encyclopedia of Quantitative Risk Analysis and Assessment. Melnick, E. L., Everitt, B. eds. John Wiley, 2008.

[21] Somayaji, Anil. "How to Win an Evolutionary Arms Race." IEEE Security & Privacy, November-December 2004: 70-72.

[22] “Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria”, DoD 5200.28–STD, 31 July 1987, NCSC–TG–005.

[23] Weissman, C. "BLACKER: security for the DDN examples of A1 security engineering trades." Proceedings, 1992 IEEE Computer Society Symposium on Research in Security and Privacy. Oakland, CA: IEEE, 1992. 286-292.