An Open-Source High-Robustness Virtual Machine Monitor

J. McDermott and M. Kang
Naval Research Laboratory

1 Introduction

The NRL high-robustness virtual machine monitor (VMM) project is based on the Xen open-source hypervisor [1]. (The term *robustness* [2] relates the concepts of *strength of mechanism* and *level of assurance*; high-robustness denotes a strong mechanism with high assurance.) Sailer et al. at IBM [3] have developed a mandatory access control security architecture for the Xen hypervisor, called *sHype*. The current Xen implementation of sHype is targeted at Common Criteria EAL 4 assurance. The NRL project has two goals: 1) demonstrate the usefulness of virtual machine monitors for realistic security applications, and 2) using Xen, demonstrate how the U.S. Navy, the Department of Defense, and other enterprises with high-robustness security requirements can take advantage of open source technology. NRL has a five-part strategy for accomplishing these goals. Each element of the strategy serves as both a set of work products to build and a research topic to investigate:

Policy-To-Code Modeling Policy-to-code modeling produces a high-fidelity collection of models of the high-robustness virtual machine monitor. For our work, these models support high-robustness vulnerability analysis and code base improvement. They are also used as evidence for third party security evaluation. Our policy-to-code modeling is focused on four work products: an *informal security problem model* that defines the security problems to be solved by the high-robustness VMM, a *formal security policy model* that unambiguously defines the security policies that our high-robustness VMM can enforce, a *semiformal interface model* that describes the interface of the VMM, and a *semiformal design model* that describes the internal architecture of the VMM. An important criterion for the interface and design modeling is to describe behavior rather than class or data structure organization. Policy-to-code modeling not only helps demonstrate how open source technology can be adapted to high-robustness, it also helps define realistic security applications of virtual machine monitors. We must be able to define realistic security problems and policies, while staying within the constraints of open source technology.

High-Robustness Code Base A high-robustness code base supports a high-robustness vulnerability analysis. The current Xen code base is already relatively small and a lot of it is high quality (e.g. cyclomatic complexity less than 10). A high-robustness code base will use a more uniform coding style than the current Xen code (e.g. all code follows all of the Linux kernel guidelines), it will be even smaller, its modularity and simplicity will be greater, and it will be refactored into layers (for EAL 6). Our demonstrations of realistic security applications need to show that a smaller, simpler, refactored Xen code base can enforce realistic policies for practical missions. It is also essential to

produce the actual code base, as part of demonstrating how DoD can take advantage of open source technology.

High-Robustness Vulnerability Analysis The goal of a high-robustness vulnerability analysis is a product that has no policy, algorithmic, or design flaws, no developer-inserted malware, and fewer coding flaws than conventional software. Security flaws are not limited to familiar *coding flaws* such as buffer overruns, but also include *policy flaws* such as enforcing digital rights management when user data protection is the security problem, *design flaws* such as using one-way authentication when two-way authentication is needed, and *algorithmic flaws* such as a weak initialization step in a cryptographic protocol. Source code analysis (with or without tools) can discover coding flaws but it cannot, by definition, discover policy, design, or algorithmic flaws without reference to some kind of (possibly implied) model. High-robustness vulnerability analysis uses explicit models. The scope and nature of the threat model used for our vulnerability analysis helps us demonstrate the usefulness of virtual machine monitors for solving our realistic security applications. It also clarifies what we mean by realistic security applications. A complete high-robustness vulnerability analysis is essential to demonstrating how open source code can be adapted to a high-robustness form.

Agile Development Process An agile development process is needed to reap the benefits of interaction with an open source community. NRL's agile process for high-robustness open source has two significant differences: third-party evaluators are viewed as additional customers to be satisfied and third-party evidence is treated as part of the "working software." The resulting approach emphasizes self-organizing teams, daily interaction with short face-to-face meetings, frequently incremental delivery, and product-focused work. Product-focused work allows choice of tools and workflows as long as the result is a conforming product. Communication and organization are accomplished through a shared repository view of work products. Our agile development process is not only essential to demonstrating use of open source for high-robustness but also helps us refine our concept of realistic application of virtual machine monitors. As we become aware of new or changing high-robustness security problems, we are able to adapt the other four parts of our strategy.

Security Feature Enhancements A VMM used for military applications will need some security feature enhancements, to meet military requirements. NRL's strategy is to identify the minimal enhancements that would be needed to support realistic military applications. Security feature enhancements are needed to demonstrate virtual machine monitor solutions to high-robustness problems. By keeping these enhancements to a minimum, we demonstrate how the solution can be based on open-source technology.

2 Accomplishments

NRL's progress toward its goals is reflected in the following accomplishments:

- completion of an informal security problem model, organized as a Common Criteria v 2.3 Security Target,
- reorganization of the Xen code base into a more modular top-level structure,
- completion of a CSP-based formal security policy model,
- identification of some flaws in the Xen source,

- minimization analysis of the Xen code base,
- layering analysis of the *acm* component of Xen,
- development of tools for construction of XML-based semiformal models, and
- construction of box-structure based semiformal models [4–6] of the *acm*, *event channel*, and *scheduler* components of Xen.

3 Future Plans

NRL's future plans are to complete the policy-to-code modeling, refactor an EAL 5 code base for a subset of the Xen VMM, design an EAL 6 code base, and release selected results to the open source community.

References

[1] P. Barham, B. Dragovic, K. Fraiser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *Proc. 19th ACM Symposium on Operating Systems Principles (SOSP-19)*, Bolton Landing, New York, USA, October 2003.

[2] ASD(C3I). *DoD Directive 8500.1 Information Assurance (IA)*, October 24 2002.

[3] R. Sailer, T. Jaeger, E. Valdez, R. Cáceres, R. Perez, S. Berger, J. Griffin, and L. van Doorn. Building a MAC-Based security architecture for the Xen open-source hypervisor. In *Proc. 21st Annual Computer Security Applications Conference*, Tucson, Arizona, US, December 2005.

[4] H. Mills, R. Linger, and A. Hevner. Box-structured information systems. *IBM Systems Journal*, 26:395–413, 1987.

[5] H. Mills. Stepwise refinement and verification in box-structured systems. *IEEE Computer*, 21:23–36, 1988.

[6] S. Prowell, C. Trammell, R. Linger, and J. Poore. *Cleanroom Software Engineering: Technology and Process*. SEI Series in Software Engineering. Addison-Wesley, 1999.