In Proceedings of the (First) USENIX UNIX Security Workshop, August 1988, p. 50

StrongBox: Support for Self-Securing Programs

B. S. Yee, J. D. Tygar, A. Z. Spector

Computer Science Dept.
Carnegie-Mellon University

ABSTRACT

Security is a pressing problem for distributed systems. Distributed systems exchange data between a variety of users over a variety of sites which may be geographically separate. A user who stores important data on processor A must trust not just processor A but also the processors B, C, D, ... with which A communicates, and the various media M_{ab}, M_{bc}, ... by which these processors communicate. The distributed security problem is difficult, and few major distributed systems attempt to address it. In fact, conventional approaches to computer security are so complex that they actually discourage designers from trying to build a secure distributed system. A software engineer who wishes to build a secure distributed data application finds that he must depend on the security of a distributed database which depends on the security of a distributed file system which depends on the security of a distributed system kernel, etc. It is hard just to make a distributed system work efficiently without considering security issues.

We have constructed and are using trusted application systems that run efficiently on machines having only minimal security facilities. Rather than depending on a tight kernel of security code, our applications perform operations that allow us to guarantee security. We call these trusted programs *self-securing*. By limiting dependence on underlying system components, we separate the security problem from the rest of the system, simplify the task of the engineer who must build such a system, and allow existing distributed systems to be retro-fitted with security. Our concern here is with security issues arising from protecting the privacy of data and the integrity of data from alteration. We do not presently consider issues of denial of service, covert channel analysis, or traffic analysis of message patterns, although we are extending our work in these directions. An important assumption in this work is the integrity and privacy of address spaces.

We have developed a family of algorithms that support self-securing programs. To show the effectiveness and efficiency of our methods, we have implemented them in a package called *Strongbox* and measured their performance in our computational environment. We have successfully built on two ongoing systems research projects at Carnegie Mellon University: Mach, a distributed operating system which is upward compatible with 4.3 BSD UNIX; and Camelot, a distributed transaction facility which runs on Mach. We do not assume that our base operating system provides full security. Our tools allow users to run application programs securely on Mach and Camelot. To demonstrate our tools, we have built a full protection system for Camelot which runs with negligible overhead. This protection system has been distributed with Camelot.