The Poly^2 Project

Center for Education and Research in
Information Assurance and Security (CERIAS)
Purdue University
http://www.cerias.purdue.edu/homes/poly2/
cerias-proj-poly2@cerias.purdue.edu

The Poly^2 project is a research project in security architecture. The goal of the project is to provide secure, highly reliable network services through the use of multiple, independent systems and multiple networks. The implementation of the platform is based on sound design principles. Poly^2 addresses security issues, high availability, and scalability for critical network services.

What is Poly^2?

Poly^2 (short for poly-computer, poly-network) is a hardened server architecture in which server applications of an organization can operate. This architecture is intended to provide robust protection against attacks to the applications running within its domain, guided by the design principles at the beginning of this section. Poly^2 is based on an approach of *physical isolation*, separating server applications onto individual computers, each of which is running an application-specific (minimized) operating system. These computers are then connected using multiple, secured networks, thus isolating different types of traffic. Trust in the entire architecture comes from the isolation of untrusted server applications running on separate computers. This hardened isolation helps contain successful attacks against individual applications. Therefore no single compromised server can bring down the entire architecture.

The minimized operating systems only provide the services required by a specific server application. Removal of all other services reduces the functionality of each Poly^2 node to a bare minimum. Specific types of network traffic such as administrative, security-specific, and application-specific traffic are also isolated onto special sub-networks. Because the nature of the traffic on each sub-network is specific and known in advance, deviations from normal traffic patterns can be detected more easily.

One Machine / One Application

Running a single application on a node (computer) achieves strong isolation, thus providing immunity from flaws in other applications. Additionally, because the underlying operating system only needs to support one specific application, the OS can be tuned to best support that application, both in terms of performance and security. Examples of performance characteristics that can be tuned include scheduling algorithms and file systems. From a security standpoint, behavior of such a specialized system is simpler to specify and deviations from normal behavior are easier to detect.

Specialized Operating System Through Minimization

The ubiquitous use of computing systems in a variety of environments has given rise to general-purpose operating systems that typically combine support for all the target environments into one assorted collection of software artifacts. Such general-purpose operating systems provide more functionality than needed, posing a security risk. The research goal is to minimize a general-purpose operating system such that it supports only those specific services that are supposed to run in a Poly^2 node, thus eliminating the threat from vulnerabilities in unnecessary subsystems of the operating system.

Specialized Networks

Poly^2 uses specialized networks for carrying different types of traffic with no traffic routed between the networks. Types of information can be categorized based on various attributes. One such categorization is based on the intended purpose of the information: application, maintenance/ administrative, and security. These three types of traffic are defined in Poly^2 to be at different sensitivity levels with the security and administrative traffic being more critical than the

application traffic. Each Poly^2 node uses a private IP address so that it is not reachable from outside the perimeter of the poly server environment.

High-Assurance Start State

The start state is the state that a Poly node boots into. Subsequently, the node may be instantiated to support a particular server application. Our start state will use invariants to argue the level of its assurance. The invariants are verified to hold by auditing the source code, and by a suite of testing methods. Invariants on the start state are stated for the interrupt controllers, protocol handlers, and a file system that is appropriate for the start state. Before the transition from the start state to another trusted state when the Poly^2 node is instantiated, cryptographic checks are needed; and the security of the transition will be proved using an enhanced version of the GNY logic, which was originally proposed for authentication protocols.

Design Principles

The Poly^2 network was designed with the following ideas in mind:

Modularity

Systems inside Poly^2 are interchangeable and independent. Any system in the infrastructure can be removed, replaced, or provisioned as needed.

Fault Tolerance

A failure in one component will not cause the rest of the system to fail. Multiple instances of a service can run on different hardware nodes.

Scalability

Adding capacity to the system is addressed with the addition of relatively inexpensive hardware.

Service Isolation

Each service resides on a singular, independent hardware node. Information flow is restricted to reduce the impact of attacks from compromised nodes.

Least Privilege

Systems and services have only the privilege needed for their function, and no more.

Economy of Mechanism

Since hardware nodes do not host multiple, unnecessary services, the supporting O/S and network implementations can be simplified.

Defense in Depth

The protection mechanisms are layered to prevent a single successful attack leading to compromise of the entire system.

Summary

For the continuation of the project, the following research tasks are proposed: (1) develop and evaluate the methodology for operating system minimization; (2) evaluate the poly-network design and define security policies for it (namely, how information can and cannot move about in a Poly^2 environment); and (3) formalize the start state and state transitions for Poly^2 nodes.

We created a preliminary implementation in 2003. Through operating system configuration, customization, and the judicious selection of networking protocols, we demonstrated that server applications operating in the Poly^2 architecture presented markedly fewer vulnerabilities than the same applications running on a general-purpose OS. Furthermore, we showed that the use of hardware and network separation eliminated the compounding effect of having multiple network services, each with their own set of vulnerabilities, running on the same hardware platform.

Selected References

- E. Bryant, J. Early, R. Gopalakrishna, G. Roth, E.H. Spafford, K. Watson, P. Williams, and S. Yost. Poly^2 Paradigm: A Secure Network Service Architecture. In *Annual Computer Security Applications Conference Proceedings - 2003*, 2003.
- P.G. Neumann. Architectural Frameworks for Composable Survivability and Security. http://www.csl.sri.com/users/neumann/chats.html.