COSEDA
Comprehensive Security for Distributed Architectures

Stephen Wolthusen

German Abstract

Das COSEDA-Programm stellt eine Information Assurance-Infrastruktur für vernetzte, heterogene Systeme bereit, mit dem beliebige Sicherheitspolitiken auch parallel automatisch durchgesetzt werden können. Sicherheitspolitiken werden auf bestimmten Knoten festgelegt, die Durchsetzung jedoch erfolgt auf jedem einzelnen Rechner innerhalb einer Organisation. Hierfür wurde ein formales mathematisches Modell entwickelt, welches die Spezifikation von Sicherheitspolitiken in einer abstrakten Darstellung ermöglicht und die jederzeit korrekte Umsetzung dieser Darstellung in einem konkreten System ermöglicht, ungeachtet der Unterschiede zwischen einzelnen Systemen. Die Realisierung einer Referenz-Implementierung der Durchsetzungsmechanismen erfolgt auf Grundlage der Microsoft Windows NT(2000/XP) Betriebssystemfamilie, ohne daß Kenntnis oder Anpassungen der Quellen des Systems erforderlich sind und können auch auf eine Vielzahl weiterer Betriebssysteme, Middleware-Plattformen und Anwendungsprogramme angewandt werden. Bei Bereitstellung innerhalb des Betriebssystems sind die Durchsetzung der Sicherheitspolitiken sowohl für Anwendungsprogramme als auch für die Nutzer selbst vollständig unsichtbar, sodaß weder Anpassungen von Anwendungen noch Schulungen von Mitarbeitern Zusatzkosten verursachen.

Introduction

The COSEDA research program's goal is to provide an information assurance infrastructure for internetworked and heterogeneous systems. Such systems face a number of challenges; some deriving from the fact that the individual operating systems, middleware platforms, and applications are frequently limited in the security mechanisms they provide or inadequate when exposed to a large, potentially hostile, internetworked environment and others from a lack of efficient enforcement mechanisms for security policies. One recent example for the inadequacies of existing mechanisms is the widespread and systematic use of Trojan horses, worms, and other malicious code for subverting system security. Besides vulnerabilities in the implementation, the commonly deployed discretionary access control mechanisms cannot deal with such threats. Similarly, while an organization's security policy or set of policies may require certain risks to be mitigated, the technical means available do not match the requirements. This is undesirable both in case the technical means are inadequate here, non-complicance will counteract risk management and mitigation intended by the policy – and in case they are overmatched and too strict – in which case needless productivity losses will result.

It is therefore necessary to be able to define and enforce sufficiently expressive policies which can be enforced in operating systems, middleware platforms, and applications, even if these did not (e.g. as with commercially available operating systems and applications) anticipate such mechanisms. At the same time policies must be defined in such a way that they can be enforced through technical means yet be sufficiently abstract so as not to place an undue burden on the security administrators and managers defining the policy. For an overall security policy to be enforceable through technical means, the ability to define an unambiguous syntactical and semantical model is critical. Moreover, for policies to be applied consistently across a distributed, heterogeneous system, a number of ontological and epistemological questions must be resolved to model the environment in which the policies are to be enforced correctly.

Architecture

The COSEDA architecture considers two types of entities, one defining policies and collecting information within their domain of control (Policy Controller Nodes, PCN), and the other enforcing the policies defined by the PCN (Policy Enforcing Nodes, PEN). A PEN can be an arbitrary entity (e.g. an embedded device, a PC running a commercial operating system and applications, or a mainframe system). Each relevant operation within a PEN is controlled by an Enforcement Module (EM), which can intercede if an operation is not conforming to the set of policies applicable to a given situation. Similarly, if an event or operation of the PEN is considered relevant for further analysis or even just auditing, the EM can note the event at any level of detail appropriate. The EM are, in case of an operating system, inserted into the kernel of the operating system in such a way that they cannot be bypassed by either the remainder of the system kernel or application programs. If placed within middleware components or application programs, they are similarly placed in non-bypassable positions. Enforcement of policies can, for example, include the automatic and mandatory encryption and integrity protection of file systems, or collecting audit information that is

used in intrusion detection and defensive reactions. However, EM themselves do not reach the requisite decisions. Rather, this is the domain of the Externally Controlled Reference Monitor (ECRM), which can (based on applicable policies cached) either decide on e.g. permitting an operation, or forward a request for a policy decision to a PCN. Once a PCN has answered, the result can be both cached for future reference and is more general than the individual instance of the request (e.g. permission to access a certain file in a networked file system).

The PCN therefore provide a mechanism to define policies in such a way that the policies are enforced in a uniform way across a distributed heterogeneous system while the actual enforcement is performed at each individual system. The distribution of enforcement is critical in an environment where centralized security mechanisms such as firewalls or mail guards are becoming increasingly ineffective. It also provides the ability to mount a nuanced defense in depth by providing the means to dynamically react to events such as elevating the auditing level and network defenses in case of a likely intrusion in an individual node of a protected network.

As part of the research program, policy-enforcing nodes (PEN) are implemented based on the Microsoft Windows NT (2000/XP) family of operating systems as a reference platform. This demonstrates that the architecture can be implemented even in case of a highly complex system to be instrumented with enforcement modules without requiring access to or modifications of source code of the operating system. Instead, arbitrary policies can be enforced without changing application programs or altering legitimate behavior of users; the security mechanisms are fully transparent to both. Policy enforcement elements have also been implemented for Unix-based operating systems and can be realized on a large number of platforms, achieving full coverage of complex heterogeneous environments. Embedding policy enforcement in operating systems, particularly systems for which no immediate access exists, can be employed in a number of security-related areas, but also extends to the policy-based management of heterogeneous systems. The sensors and actuators embedded in PCN provide a detailed view of operational characteristics of all nodes in an organization's network and can also be used to effect required changes. Other benefits provided by the enforcement mechanisms can include a rapid reaction to attacks such as constraining the behavior of a node's network stack and applications in such a way that a vulnerability for which no direct patch exists is eliminated or at least mitigated. This can significantly limit the exposure to vulnerabilities since any patches or ECOs must typically be qualified by IT staff while policy-based preventive measures can be implemented immediately, frequently even automatically. To achieve adequate assurance in the COSEDA architecture itself, formal methods are employed in the design, specification, and implementation of the architecture.

Mathematical Framework

The core of the COSEDA program is a mathematical framework facilitating a number of advanced security mechanisms. The relevant aspects of the systems to be secured are captured in a formal mathematical model based on first order logic and derived from formal concept analysis. This provides a common basis for formulating the configuration, behavior, and operation of heterogeneous components such as operating systems, while an interpretation maps this formal model onto a specific instance of e.g. an operating system. By employing the same mathematical model in formulating security policies, it thus becomes possible to describe permitted or required behavior within the abstract model once while the interpretation or translation into a given system occurs consistently and, more importantly, automatically. A security administrator can therefore be sure that a policy formulated once is always enforced consistently, even in a heterogeneous distributed system. Given the multitude of possible technical instantiations of entities and operations relevant to the enforcement of a policy and the limited ability of humans to handle inordinate amounts of data in defining such technical detail, an additional element of the mathematical framework in the form of infinite lattice structures provides an abstraction mechanism which permits the identification of derivative instances (e.g. a block in a file system can be considered a derivative instance of a file, which in turn can be derived from a document object) of entity types and identities. This not only provides an abstraction which can be used to simplify and generalize the definition of policies, it also provides a means for identifying entities and entity types consistently across distributed networks, considerably adding to the means available for defending internetworked heterogeneous systems.

Moreover, the formal model also permits real-time automated reasoning over security policies, automatically deriving permitted or required operations based on abstract or more general policy formulations. This considerably reduces the complexity of policy definitions a security administrator is faced with, since the semantic distance between traditional natural language policies and the technical means for enforcing these policies is reduced. Since at the same time the derivation of technical enforcement measures is performed automatically, human error and other inconsistencies (e.g. due to parts of a technical implementation not being updated manually when the underlying security policy changes) are eliminated from the process and the auditability of the security policy mechanism is enhanced.

Point of contact

Stephen D. Wolthusen
Fraunhofer IGD, Darmstadt,
Germany
Email: wolt@igd.fraunhofer.de