Environment-Aware Security Enforcement (EASE) for Cooperative Design and Engineering

Korba, Larry; Xu, Y.; Song, Ronggong; Yee, George

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de technologie de l'information

NRC · CNRC

*Environment-Aware Security Enforcement (EASE) for Cooperative Design and Engineering**

Korba, L., Xu, Y., Song, R., and Yee, G.
September 2005

* published in Proceedings of the Second International Conference on Cooperative Design, Visuallization and Engineering (CDVE 2005). Palma de Mallorca, Spain. September 17-20, 2005. pp. 140-148. NRC 48225.

Canada

Environmentally-Aware Security Enforcement (EASE) for Cooperative Design and Engineering*

Larry Korba, Yuefei Xu, Ronggong Song, and George Yee

Institute for Information Technology, National Research Council of Canada,
Building M-50, Montreal Road, Ottawa, Ontario
{Larry.Korba, Yuefei.Xu, Ronggong.Song,
George.Yee}@nrc-cnrc.gc.ca
http://www.iit-iti.nrc-cnrc.gc.ca

Abstract. In cooperative design and engineering, work context has dramatic implications when it comes to building understanding and teams, and getting the work done. From an information security perspective, the context of operation for software or hardware can drive the expectations for security, and may indeed determine the levels of security policy that would be required for collaborative operation. There is a place in this domain for proactive, context-based security implementation. This paper describes a context-centric security enforcement system intended for cooperative design and engineering environments that we call: Environment-Aware Security Enforcement (EASE). We describe the application of this approach to an existing client-server, e-manufacturing application.

1 Introduction

In computer supported cooperative work environments work context can have a dramatic bearing upon the effectiveness of a collaborating team. Yet building effective cross-organizational collaborative environments is a big challenge, especially from the security perspective [1]. Context may be used to tailor information delivery and sharing based upon location, available resources, perceived activity needs, and expertise requirements. To support this idea, there have been a number of recent developments in assessing and using context in collaborative work environments.

When considering it from a security standpoint, context may be used to determine where, when, how and for how long individuals and organizations may share information with each other. The greatest benefit of establishing context-aware security mechanisms is to enable enforcement of security for mission-critical distributed applications in conformance with the security expectations of all collaborators in all contexts of their work. Ideally the collaborative environment would take into account the networks, computer operating environment, the tasks at hand, and other factors for all collaborators when determining if and how they may be allowed to work together.

In this work we present a model, design and prototype implementation for a system that uses a variety of contextual information to enforce whether or not collaborative environments may be usable. This builds upon research we have presented else-

* National Research Council Paper Number 48225.

Y. Luo (Ed.): CDVE 2005, LNCS 3675, pp. 140 – 148, 2005.

where [2]. The type of contextual information we use is typical of the information used when building security policies for organizations. The design provides a means for setting and enforcing security policy for the operating contexts of collaborators. The system assures that security requirements for all clients are properly maintained for secure collaborative operations. Within the system, policies may be set for different aspects of a user's context, including: computer or software platform, network connections, locations (physical context), social and work behaviors (cultural and social contexts), and the nature of the information being shared or built (information context). As extensions to this work, historical contexts relating the nature of activities over a period of time may also be used as a trigger for precautions or allowances in collaborative environments. In the current implementation, a variety of software-based sensors are used to determine the context of the user. Security agents running on client computers are responsible for controlling and monitoring these sensors. The security agents also communicate with a policy agent during collaborative interactions to enforce the security policies. Depending on the policies and the client operating contexts, the security agent may prevent or proactively enable computer activities locally, or deploy services in support of secure collaborative operation. An additional challenge associated with this work is that we want to secure operations in a pre-existing collaborative software (an e-manufacturing application). Our approach is to provide security enforcement while minimizing the impact upon the existing legacy e-manufacturing application.

This paper is organized in the following way. Section 2 describes the problem we are addressing with this work. With a description of the target domain in place, we describe our approach in general, detailing the architecture, and implementation in Section 3. In Section 4 we describe relevant previous research related to this area. Our Discussion and Conclusions section follows in Section 5

2 Problem Statement

There have been a great many technologies developed to help groups of people collaborate more effectively. Networked computers form the basic substrate upon which different collaborative technologies have been built. Computer Supported Collaborative Work (CSCW) research involves investigation and development of approaches that make collaboration between users using this substrate more effective. Advances of CSCW have been applied to cooperative design and engineering.

Indeed, computer networks have become prevalent in all organizations. While organizations have been able to gain advantages in efficiencies and their work through their use, inter-networked computer systems also present a risk to the operation of organizations. In terms of cooperative design and engineering, a key concern is the assurance that proprietary information about the intellectual property owned by the organization or information about the company operations is available only to authorized individuals. Within an intranet environment, access privileges may be adequately controlled. Interconnecting intranets over the Internet to allow different organizations in different locations to collaborate, as would be the case for cross-organization collaboration, and/or design/production outsourcing, creates a liability in terms of the potential for unauthorized access to information, computers or devices on the company intranet.

Internet-based manufacturing involves sharing intellectual property in the form of detailed engineering and manufacturing information as well as competitive information in the form of order and costing details. The bottom-line here is that for general acceptance of an Internet-based cooperative design and engineering approach, the secrecy of the proprietary or competitive information must be maintained.

In addition to maintaining secrecy, Internet-based manufacturing must accommodate confidentiality of the organizations involved in the manufacturing process. Gathering and processing information about the activities of individuals or groups while managing or operating processes or machinery via computer networks can provide considerable detail concerning the ways in which the individuals interact as well as process-related information. In a highly competitive manufacturing environment, information about internal organizational operations must only be shared on a "need–to-know" basis. This work addresses the following questions:

- While the organizations involved in the collaborative work may have written policies describing how all participants are expected to behave, how will those policies be enforced?
- In a client-server context, can security policies apply security constraints on an end user's operating environment and based on the context of the end user's operations?
- Is it possible to add security enforcement to existing or legacy applications easily?

The next section describes our approach for answering the above questions.

3 Our Approach

Our approach is to apply policy-based security enforcement for a client-server application. Policies are created and managed centrally at the server. There are sensors at the client that measure whether or not each central security policy is maintained in compliance on the client platform. Enforcement and feedback to the user through the client application is done via the server. We detail first the design for EASE as applied to a client-server application, followed by a description of the implementation.

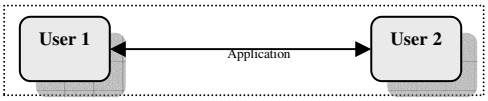
3.1 Design

In order to understand the design approach we have taken for EASE, we first describe the overall concept, then provide an example in the form of a specific implementation targeting an e-manufacturing application.

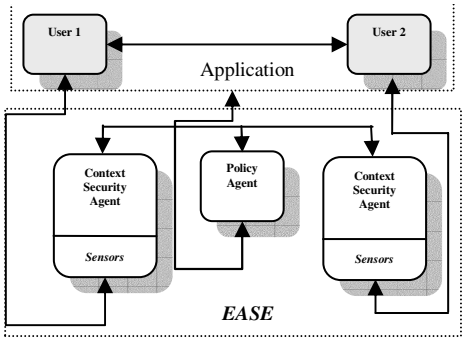
Figure 1 illustrates a system before and after being deployed with EASE. Illustrated in Figure 1(a) is the system to be protected. It may be a client-server or distributed application. The application may be designed for any particular purpose and may or may not have security functions built into its design. Indeed, while the system may include, for instance, methods to authenticate users, traditionally there are few methods available to enforce security compliance at the client end.

Figure 1(b) illustrates the application in operation with EASE in place. A policy agent, located at the server site, interacts with a security agent located on each platform where the application may be operated remotely. The policy server interprets an electronic version of the security policy for the applications that may run on the computer platform. The security agent and policy agent communicate with each other via a dedicated, secure communication channel. The policy agent communicates to the application in order to share information through messages and to enable or disable access to the application for any user. In order for the application to operate from any computer platform, the security agent must be present and must be in communication with the policy agent. Each security agent has a set of security (software) "sensors" that monitor different aspects of operation context for the application on its computer platform. Examples of the security policies that sensors and controlling software may monitor and enforce include the following:

- Ensure the operation environment is appropriate for the application. For instance for a Java application, it is important to ensure that the Java runtime version installed is adequate for the application to run correctly. In addition, to prevent security flaws and exploits, the system must ensure that the version of Java installed has not been tampered with.
- In order to protect distribution of copyrighted material or trade secrets, the system must ensure that when the application is running there are no other applications operating that may be used to garner inappropriate access to intellectual property or the application itself. These applications may include disassemblers, reverse compilers, screen snapshot software, file or system activity monitoring software, etc.
- Ensure that the computer platform running the application remotely is safe from viruses and Trojans. As with the above, this would amount to an enforcement of the security policy before the application is allowed to run.
- Ensure that only certain computers run the application. There may be other measures that can restrict access to the application. The environment's security policy may restrict computers based upon IP address, MAC address, or computer hardware signatures.
- Ensure that the computer upon which the application runs remotely always has a security device installed. (The device may be a smart card, USB security device, other security dongle (wired or wireless). Security enforcement runs in a separate execution thread from the application, regularly.
- The policy agent may add further functions for improving authentication and authorization for the application. For instance, it may enforce password changes, or role-based authentication and authorization. The advantage of adding this function would centralize security administration for the application.
- Another item that may be monitored and reacted upon is an analysis of the situational behavior of all participants. If a user is considered to be behaving in an anomalous way, the application may disconnect the user. In other words, users will remain connected as long as they behave in the fashion expected by the administrators of the collaborative software system.



a) The application environment with two users, 1 and 2 connected through the application



b) The application environment with two users employing EASE.

Fig. 1. This diagram illustrates how EASE may be applied to an existing application

3.2 Implementation

To demonstrate EASE, we have created a software prototype. This section describes the implementation of the prototype.

This work builds upon the research and prototype development described in [3]. The web-based shop-floor monitoring and control program is called Wise-ShopFloor and is further described in [4]. A much simplified block diagram for the Wise-ShopFloor is shown in Figure 2. Web clients may access a variety of shop floor

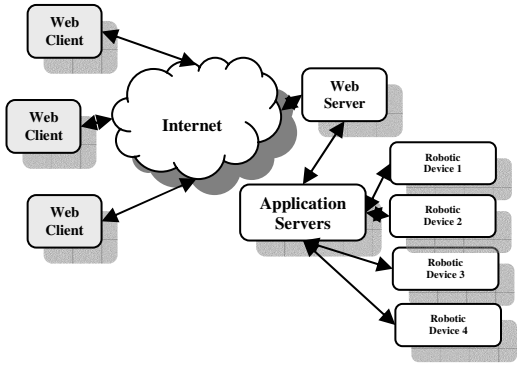


Fig. 2. Three-tiered architecture for e-manufacturing (Wise-Shopfloor) before the addition of EASE

equipment (e.g. robots, milling machines) via a web interface. Users are provided with three dimensional visualizations of the operations of machinery for control or operation purposes. The system uses a client-server architecture. The clients use a web browser. Java 3D provides the 3 dimensional visualizations of all machine operations. Java servlets form an application server used to access the various different machine tools. Before EASE was applied to the prototype, the primary security feature of the Wise-ShopFloor application was based on simple password authentication access control. The objective of our work was to add monitoring of the contexts of application execution and using the monitoring results to control context-aware access to the application.

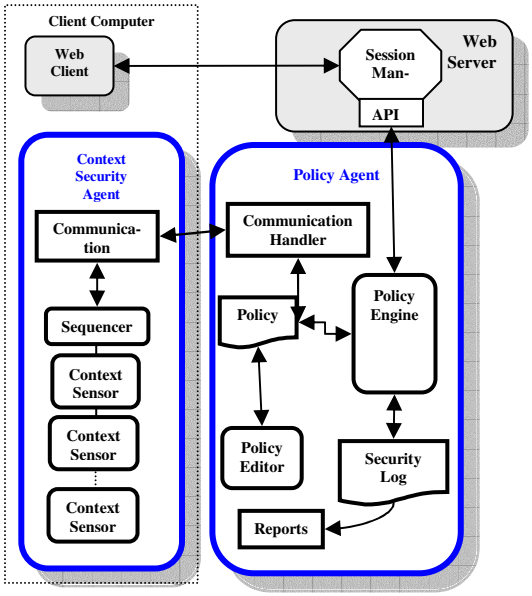


Fig. 3. This diagram illustrates the components comprising the system we build to provide Environment-Aware Security Enforcement (EASE)

Figure 3 illustrates our approach for adding EASE to the Wise-ShopFloor application. The Session Manager is the key place connecting EASE with the application. The interface consists of a simple application programmer interface (API). Figure 4 illustrates some of the components of the Policy Agent (PA) and Context Security Agents (CSA). The PA interprets policy, communicates with all context CSAs (one for every computer connected remotely to the application). Communication handlers at both the PA and CSA exchange commands, alarms and policy information. Depending on the nature of the policy, it is either interpreted at the PA where upon, the PA sends a message to the CSA to perform a policy compliance check, or it is sent to the CSA if a policy is to be scheduled regularly. In the latter case, the CSA schedules the appropriate sensor to perform the test periodically, signaling the PA when the test fails. When a sensor test fails, the PA sends a message to the session manager to deal

with the consequence of the failure and ultimately provide feedback to the user. A policy statement in its simplest form contains the name of the sensor test, the frequency of the test, the consequence of its failure and the message that is sent to the Web Client for display as a pop-up window indicating a security failure. The failure may result in a warning to the user, or a message indicating that the application has disconnected.

The PA also houses a policy editor for creating and changing policies. Each rule is named and applied to operate with a single machine connected to the Wise-ShopFloor Application. Rules may also be applied to users. The rule is applied to one of the two tasks associated with most machines: Monitor or Control. The action associated with the rule may be either to permit or deny the selected task on the machine. As many sensors as required may be added to the rule. For each sensor, the administrator selects the sensor test to be applied, the expected response for the test, and the message to be sent to the application user if the test fails.

4 Previous Work

Two general research areas attributable to EASE are policy-based management (EASE uses policy rules to manage its operation) and pervasive/ubiquitous computing security (EASE enforces security based upon user and computer platform contexts).

McDaniel and Prakash have described an architecture for security policy enforcement [5]. Named "Antigone", the architecture offers a modular approach for adding security event detection modules. The system uses a transport layer mechanism and security-related events for handling by the detection modules. The authors describe optimization methods to reduce overheads in the architecture. Antigone is intended to be built around applications. With Antigone, the intended target for the security enforcement is appropriate behavior of the application as opposed to EASE where the target for enforcement is beyond the application extending to the context of work station operation and user behavior.

An example of an attempt to provide security enforcement for a computer platform at least at the level of the file system is given by Wolthusen [6]. The system described in this paper provides mandatory access control, encryption and auditing of file activity on an individual file basis for a distributed system. The system that the author has developed provides a holistic approach for handling file activity for the Windows NT file system. Ostensibly, the technology could be applied as a file context sensor for EASE, taking advantage of our effective management interface. While this approach would provide little or no advantage for the Wise-ShopFloor application, it would be advantages when EASE is adapted for distributed applications that have local file management requirements.

Schneider describes a practical way to enforce security policies by monitoring system and application processes by automata for safety-critical systems [7]. Each automaton is intended to deal with a security policy. The author describes and defines security policies as being "specified by giving a predicate on sets of executions. A target *S satisfies* security policy *P* if and only if $P(\sum_S)$ equals *true*." This definition applies well in EASE as well, since the decomposition of what might be complex

policies into the conjunction of separate mechanisms used to enforce each of the component parts holds for our work. In addition, EASE targets enforcing security policies for distributed applications, whether they are safety critical or not. Moreover, EASE is not concerned with the policies for the application, per se, but rather those security policies applied to the execution environments for applications. The context sensors in EASE are automata-like in operation.

Covington et al. describe a context-aware security architecture (CASA) for emerging applications [8]. Similar to our work, CASA employs different sensors that monitor resources, systems and physical sensors to measure different contexts for participants in the application. CASA also uses context and object management layers as well as services for object and environment roles activation that influence an authorization service. The overall objective of this work is to provide more adaptive security services. With EAVE, we present and demonstrate a straight-forward approach for adding and managing security for a distributed, shared workspace using context and an agent-based methodology.

5 Discussion and Conclusions

There has been considerable research and development of policy management systems for network management and security management. However, there has been little work in the area of enforcement of security policies. The work that has been done in this area often involves incorporating security enforcement into the systems early on; at the beginning of the design process. In this work we take a new approach we call: Environment-Aware Security Enforcement (EASE): a fusion of work in policy-based management and enforcement and the context-based security research in pervasive computing. EASE offers a means of providing security policy enforcement for legacy applications. As was shown with our target Wise-ShopFloor application, the interface with the existing application was a simple API added to the legacy system to provide control over a user's authorization to use the application and messaging through the application for user feedback. The result nicely integrates the security enforcement functionality of EASE with the original application functionality.

EASE extends the security functionality of the target application to which it is applied by providing context-based security enforcement features. A variety of logical sensors may be applied to different aspects of a user's operating environment. Tests and enforcement measures may be taken to assure that the user is maintaining certain security requirements for operating an application even if the location of the user is outside the organizational physical and virtual boundaries.

The current proof-of-concept prototype demonstrates a limited number of sensors in operation. These sensors currently include: Java virtual machine integrity, operating system version, MAC and IP addresses. Future work will involve developing a more comprehensive set of sensors, including ones to determine whether or not an individual user is doing inappropriate or unexpected activities based upon system call information and social network analysis [9].

Acknowledgements

The authors acknowledge the National Research Council of Canada for its support of this work through their collaborative grant program with the National Science Council of Taiwan (2002-2005). We gratefully acknowledge the contribution of our collaborators in the NRC Integrated Manufacturing Technologies Institute for constructing the hooks in their prototype software required for our implementation. We also acknowledge the assistance of Myroslav Palenychka in the development of the prototype during his Jan.-Apr., 2005 work term within the Information Security Group.

References

1. Fuchs, L, Geyr, W., Richter, H., Poltrock, S. Frauenhofer, T., Daijavad, S. Enabling inter-company team Collaboration. Enabling Technologies: Infrastructure for Collaborative Enterprises, 2001. Proceeding tenth IEEE International Workshop on WET ICE 2001, Cambridge, June 20-22, 2001, pp. 374-379.
2. Xu, Y., Song, R., Korba, L., Wang, L., Shen, W., Lang, S., Distributed Device Networks with Security Constraints, IEEE Transactions on Industrial Informatics, 2005 (Accepted).
3. Shen, W., Lang, S., Korba, L., Wang, L., and Wong, B., 2000, "Reference Architecture for Internet Based Intelligent Shop Floors," Proceedings of the SPIE International Conference on Network Intelligence: Internet-Based Manufacturing, Vol. 4208, pp. 63-72.
4. Wang, L., Song, H. Cunningham, A. Development of Wise-ShopFloor for Web-based Monitoring and Control, Technical Report from the Integrated Manufacturing Technology Institute, National Research Council of Canada, IMTI-TR-023 (2004/02) January, 2004 (41 pages).
5. McDaniel, P., Prakash, A. A flexible architecture for security policy enforcement, Proceedings of the DARPA Information Survivability Conference and Exposition, Washington, DC, April, 2003, pp. 234-239, vol. 2.
6. Wolthusen, S.D. Security Policy Enforcement at the file system level in the Windows NT operating system family, Proceedings of the 17th Annual Computer Security Applications Conference, New Orleans, Dec 10-14, 2001.
7. Schneider, F.B. Enforceable Security Policies, ACM Transactions on Information and System Security (TISSEC), Vol. 3, Iss. 1, February, 2000, pp. 30-50.
8. Covington, M.J., Fogla, P., Zhan, Z., Ahamad, M. A context-aware security architecture for emerging applications, Proc. of the 18th annual Computer Security Applications Conference (ACSAC'02), Las Vegas, Dec. 9-13, 2003, pp. 249-258.
9. Temdee, P., Korba, L. Of Networks, Interactions and Agents: An Approach for Social Network Analysis, Proc. 6th International Conference on CSCW in Design, London, Ontario, July 12-14, 2001.