Coordination of Security Levels for Internet Architectures

Eduardo B. Fernandez
Dept. of Computer Science and Eng.
Florida Atlantic University
Boca Raton, FL 33431, USA
ed@cse.fau.edu

Abstract

Internet systems provide a variety of ways to exchange information, contain large amounts and variety of data, and have become quite complex, making them vulnerable to attacks from determined hackers. There are many products used currently to stop these attacks but they suffer from lack of completeness, they only apply to one type of attack. Several mechanisms are needed for a comprehensive defense but this incurs in the problem of their lack of coordination, which can be exploited for attacks. We propose a way to coordinate different mechanisms based on a unified object-oriented modeling approach and a hierarchical architecture whose layers define the scope of each security mechanism.

1. Introduction

Internet systems are very complex, they involve a variety of machines, operating systems, languages, and applications exchanging information using a rather simple protocol. They provide a variety of functions and every day new ways of use are being found. The complexity of the systems involved and the limitations of the typical protocols used provide ample opportunities for hackers to attack these systems [Bou98].

Current systems incorporate a variety of mechanisms to thwart attackers, e.g., cryptographic protocols, intrusion detection methods, authorization systems, etc. [Gar97] Each mechanism is geared to stop some specific type of attacks, is typically produced by a different vendor, and there is no coordination between different mechanisms. This lack of coordination can be exploited to attack the system, a knowledgeable hacker can exploit interactions between systems which are not well checked. To make things worse, some surveys [Atk97], give the misleading impression that Internet security is only a problem of network security and ignore the rest of the system. In fact, the Internet has brought upon a fusion of internal and external systems, and its security problems cannot be considered in isolation of the involved internal systems. From current statistics one can see that more than a half of the attacks come from internal users [NYT98].

Object-oriented design has introduced a systematic way of developing software [Rum91]. Less known, is its ability to model existing systems.

We propose here a methodology to coordinate security mechanisms that should increase total system security. The idea is to define abstract hierarchical architectural levels and to model all mechanisms in a uniform way using an object-oriented approach. The modeling includes the definition of mappings between the elements of the models at each level. Authorization restrictions are defined at the application level and enforced by the lower levels, that implement the appropriate mappings.

Section 2 describes a hierarchical architecture and indicates where the current security mechanisms fit. Section 3 shows the types of object-oriented models that we need at each level, while Section 4 describes mappings between levels and an ideal embodiment of these ideas. We end with some conclusions and ideas for future work.

2.Architectural levels and security mechanisms

It is possible to visualize the structure of a computer system as a hierarchic set of layers or levels. Many interpretations of the layers are possible, e.g., the layers correspond to the levels of the structure of the software/hardware architecture. In particular, in earlier work we defined a security layer [Lan77], a fault tolerance layer [Anc88] and a refined set of security layers [Fer96]; that is, we have concentrated on layers that correspond to nonfunctional aspects of an application. A detailed set of nonfunctional layers was shown in [Fer95]. We consider here further aspects of this multilevel hierarchy that can be exploited for Internet security.

In the case of the Internet we can interpret these layers as shown in Figure 1. We have left out real-time and fault tolerance aspects and we concentrate on those aspects relevant to security.

Security mechanisms normally are applied to one or more of these layers:

- At the physical or network layer, cryptographic protocols may be applied [Opp97]. Several sublayers may be involved, using different cryptographic protocols, e.g., SSL.

- At the OS layer we have memory protection and file rights [Tan96]. The complete OS can be protected from external access using firewalls [Opp97]. Internal firewalls can control departments or other company divisions. Languages such as Java let users download programs from other nodes in the network; these programs may become security threats, for example compromising files [Kov97].

- At the DMBS layer, many authorization models have been proposed [Ber94, Fer93]. These protect the data at a granularity that can cover specific data values.

- At the application layer we can define authorizations using the conceptual model of the system and Role-Based Access Control [San96].

In fact, authorization constraints should be defined at the application layer and mapped downward to the lower layers, which enforce them [Fer93]. Under the application layer, a variety of software packages or DBMSs apply the security constraints [Rie98]. The application authorization constraints are the basis of the coordination of all the lower-level mechanisms. It makes no sense to define rights directly at the low levels because these cover only specific layers and can be easily bypassed.

3. OO models at each level.

It is possible to model the mechanisms at each level using an object-oriented notation such as UML [Rum98]. We have shown in earlier work models of this type for authorization rules and for a complete DBMS [Fer93]. We also developed models for more specific software systems, e.g., hypertext document security [Fer98]. It is also possible to model in this way hardware configurations or even the structure of processors and processes.

We start by defining authorization rules from use cases [Fer97]. This is an application of Role-Based Access Control, where the actors are given the rights they need to perform their duties. These rights are then applied to the OOA class and state models, as shown in Figure 2. In this case they define that a manufacturing employee (MfgEmp), can cut (reserve components) and pick components for a shop order, while an OrderEntryEmployee can create shop orders.

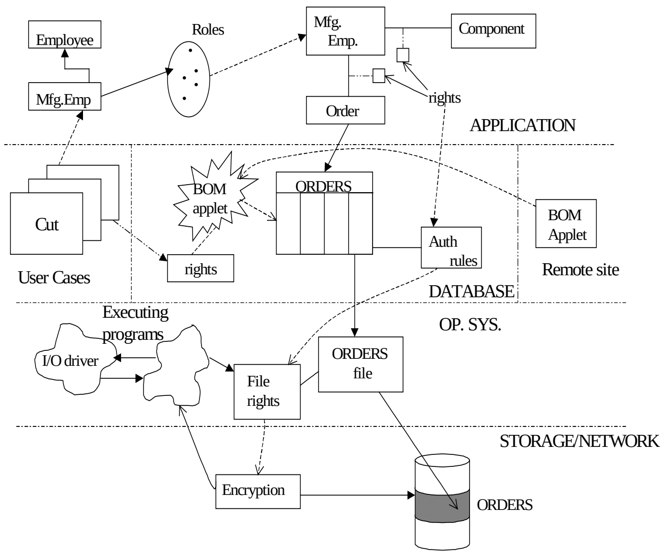

Figure 1. Security levels

Figure 2 shows an authorization rule at the application level. In particular, a MfgEmp role is showed having rights to access components and shop orders. The application authorizations map into some authorizations for a table in the DBMS. In turn, these DBMS authorizations define some file and memory rights for executing processes at the OS/concurrency level. The figure also shows how a downloaded applet that determines Bill of Materials for orders receives rights to access table ORDERS and other resources from the Use Cases. At the OS level, the rights of executing programs, including applets, should be controlled with respect to use of resources, e.g., memory or files. Cryptographic controls at the hardware (storage and network) level protect the information in transit.

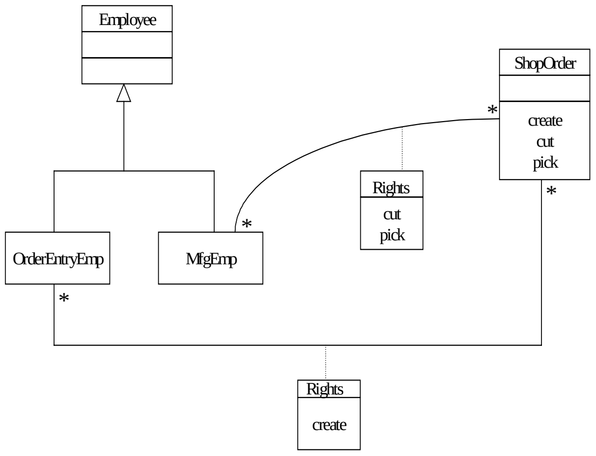

Figure 2. Authorizations

Figure 3 shows the generalized model, where object-oriented diagrams, both static and dynamic, describe specific security mechanisms at each level. The addition of constraints can make the models more precise. Either OCL [War98], Z [Coo94], or a similar formal language can be used. The dynamic model uses statecharts but Petri nets can be used for more detail. Use of the dynamic model allows the definition of authorization rules with timing restrictions and guards.

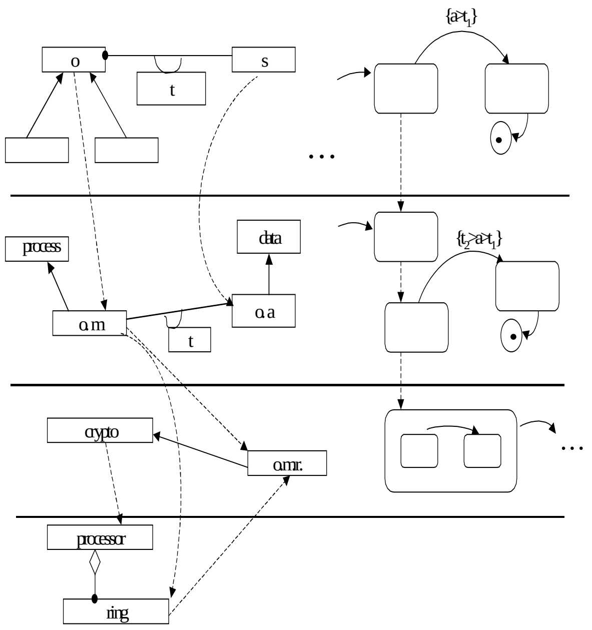

Figure 3. Object-oriented models of security mechanisms

The use of purely formal methods at each layer, as discussed in [Fer95], doesn't seem a good approach, because of their lack of intuitive interpretation and their difficulty in describing complex systems. As indicated above, a combination of object-oriented diagrams with OCL or Z constraints appears precise enough for most restrictions as well as being intuitive.

Z or OCL also appear convenient to define precisely the mappings from one level to another. A class can be mapped to a whole table or its attributes can be distributed in several tables [Rum91]. Authorizations are defined as associations in the conceptual model and are stored in relational databases as special tables [Cas94]. Note that there may be application authorization rules that do not map to a DBMS authorization, but map to a concept in a lower level. Part of the Z mapping for the rules of Figure 2 is as follows:

```
------------------- Rel_mapping --------------------
   Order :  CLASS
   MfgEmp :   CLASS
   ORDERS :   table
-------------------------------------------------------
   order.number  →  ORDERS.number
   order.date      →   ORDERS.issue_date
   MfgEmp  →  order.cut  →  MfgEmp
                →   ORDERS.number(select),
                →    ORDERS.issue_date(select)
-------------------------------------------------------
```

This mapping uses an extended Z, where the symbol -> indicates correspondence across levels and the dot notation indicates either an attribute of an object or a column in a table. Specifically, the mapping shows here that order.number in the application level maps to the column number in table ORDERS, etc. The last line defines the rights of manufacturing employees at the DBMS level starting from their rights at the application level. Here ORDERS.number(select) indicates that the operation select can be applied to the column 'number' of table ORDERS. This example needs to be generalized in the style of [Woo79], where we mapped view authorizations to conceptual model authorizations.

Once we have defined the models at each level and their interlevel mappings we can look for patterns that correspond to secure system structures. Patterns have proved to be very useful in object-oriented design to help developers use well-tried designs. Figure 4 shows a pattern at the application level that makes use of an aggregation structure. It shows that the class CEO (a singleton) controls Dept. Heads, which, in turn, control Project Leaders. The CEO has global rights on the company, Dept. Heads have rights on departments and project leaders have rights on projects. Also, the CEO rights include the rights of Dept. Heads and so on down the aggregation hierarchy. This kind of patter simplifies the work of security administrators, who do not need to define each individual right.

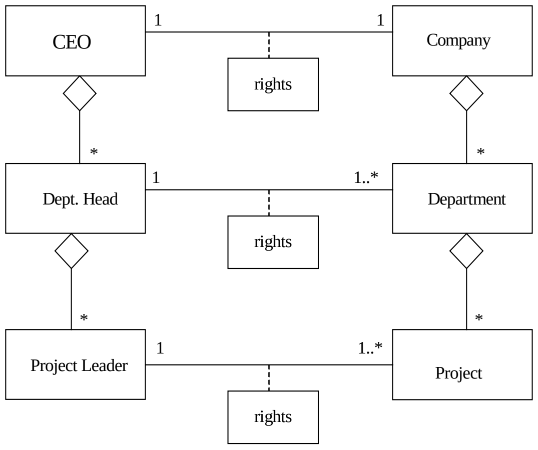

Figure 4. A security pattern

4. Conclusions

Internet systems are complex and their complexity is increasing through the use of use of components off the shelf (COTS) [Lin98], XML, and agents [Kar97]. Distributed systems, combining technologies such as CORBA and DCOM with the Internet are increasingly used. All this will bring new security problems. Without a unified view of security across all the system levels, security vulnerabilities will be exploited by hackers. Our unified model is an attempt to put some structure in what is now a chaotic combination of many diverse mechanisms.

These concepts can be applied in different ways:

- ❑ To define new secure architectures. For example, we proposed in [Fer98] a 3-layer approach using a hardware-based web server designed according to the following structure:

  - ▪ A view/presentation layer, distributed across user browsers.
  - ▪ A model layer, stored in the web server.
  - ▪ A storage layer.

This server would also require a strong operating system such as HP's Virtual Vault [Zho98], as well as cryptographic protocols. Its layers would be coordinated using the models described here.

- ❑ To complement a system-wide administration structure, such as the one described in [Ess98]. This system coordinates heterogeneous security systems at the user level, while our proposal takes the coordination to the lower levels.

The ability to model security mechanisms has already been demonstrated, what we need is a precise way to map mechanisms across levels in a generalized way. While it is very difficult to guarantee that the more detailed lower levels enforce the policies, we should at least guarantee that the concepts at each level are properly mapped. We are not dealing here with the design of new systems that have to be provable secure but we are considering existing products that need to work together. Another interesting direction is the use of reflection to provide security at specific layers [Anc98], a reified level could be controlled from a higher level.

5.Acknowlegments

The comments of the referees were very useful to improve this paper.

6. References

Anc88 M. Ancona, A. Clematis, G. Dodero, E. B. Fernandez, and V. Gianuzzi, "A system architecture for fault tolerance in concurrent systems", *Computer*, Vol. 23, No 10, October 1990, 23-32.

Anc98 M. Ancona, W. Cazzola, and E. B. Fernandez, "Reflective authorization systems: Possibilities, benefits, and drawbacks", in *Secure Internet programming: Security issues for distributed and mobile objects*, J. Vitek and C. Jensen (Eds.), Springer Verlag, 1999.

Atk97 R. Atkinson, "Toward a more secure Internet", *Computer*, January 1997, 57-61.

Ber94 E. Bertino and H. Weigand, "An approach to authorization modeling in object-oriented database systems", *Data and Knowledge Engineering*, 12, 1994, 1-29.

Bou98 A. Boulanger, "Catapults and grappling hooks: The tools and techniques of Information warfare", *IBM Sys. Journal*, vol. 37, No 1, 1998, 106-114.

Cas94 S. Castano, M. Fugini, G. Martella, and P. Samarati, *Database security*, Addison-Wesley 1994.

Coo94 S. Cook and J. Daniels, "Let's get formal", *JOOP*, July-August 1994, 22-24 and 64-66.

Ess98 W.Essmayr, E.Kapsammer, R.R.Wagner, G. Pernul, and A.M.Tjoa, "Enterprise-wide security administration", *Procs. 9th Intl. DEXA Wokshop*, 1998, 267-272.

Fer93 E. B. Fernandez, M. M. Larrondo-Petrie and E. Gudes, "A method-based authorization model for object-oriented databases", *Proc. of the OOPSLA 1993 Workshop on Security in Object-oriented Systems* , 70-79.

Fer95 E. B. Fernandez and R. B. France, "Formal specification of real-time dependable systems", *Procs. ICECCS'95*, 342-348.

Fer96 E. B. Fernandez et al., "High-level security issues in multimedia / hypertext systems", in *Communications and Multimedia Security II* , P. Horster (Ed.), Chapman & Hall, 1996, 13-24.

Fer97 E. B. Fernandez and J. C. Hawkins, "Determining role rights from use cases", *Procs. 2nd ACM Workshop on Role-Based Access Control*, November 1997, 121-125.

Fer98 E. B. Fernandez and K. R. Nair, "An abstract authorization system for the Internet", *Procs. 9th Int. Workshop on Database and Expert Systems Applics. (DEXA '98)*, 310-315.

Gar97 S. Garfinkel and G. Spafford, *Web security and commerce* , O'Reilly and Assocs., Inc., 1997.

Har97 B. Hartman, "DCOM and CORBA --Secure interoperability ? ", *Distributed Object Computing*, July 1997, 47-49.

Kar97 G. Karjoth et al. , "A security model for Aglets", *IEEE Internet Computing*, July-August 1997, 68-77. http://www.ibm.com/java/education/aglets

Kov98 L. Koved, A. J. Nadalin, D. Deal, and T. Lawson, "The evolution of Java security", *IBM Sys. Journal*, vol. 37, No 3, 1998, 349-364.

Lan77 T. Lang, E. B. Fernandez, and R. C. Summers, "A system architecture for compile-time actions in databases", *Procs. 1977 ACM Annual Conf.*, 11-15.

Lin98 U. Lindqvist and E. Jonsson, "A map of security risks associated with using COTS", *Computer*, June 1998, 60-66.

NYT98 *The New York Times*, March 2, 1998, "Threat to computers is often the enemy within", C1-C2.

Opp97 R. Opplinger, "Internet security : Firewalls and beyond", *Comm. of the ACM*, May 1997, 92-102.

Rie98 R. van der Riet, W. Janssen, and P. de Gruijter, "Security moving from database systems to ERP systems", *Procs. 9th Intl. DEXA Workshop*, 273-280.

Rum91 J. Rumbaugh et al., *Object-oriented modeling and design*, Addison-Wesley 1991.

Rum98 J. Rumbaugh, G. Booch, and I. Jacobson, *The Unified Modeling Language Reference Manual*, Addison-Wesley 1999.

San96 R. Sandhu et al., "Role-Based Access Control models", *Computer*, vol. 29 , No 2, February 1996, 38-47.

Tan96 A. S. Tanenbaum and A. S. Woodhull, *Operating systems: Design and implementation*, (2nd Ed.), Prentice-Hall 1996.

War98 J. Warner and A. Kloppe, *The Object Constraint Language: Precise modeling with UML*, Addison-Wesley 1998.

Woo79 C. Wood, R. C. Summers, and E. B. Fernandez, "Authorization in multilevel database models", *Information Systems*, vol. 4 No 2, 1979, 155-161.

Zho98 Q. Zhong and N. Edwards, "Security control of COTS components", *Computer*, June 1998, 67-73.