TOWARDS A MULTI-MODEL VIEWS SECURITY FRAMEWORK

Lei Xia, Hao Huang, Shuying Yu
State Key Laboratory for Novel Software Technology, Nanjing University,22 Hankou Road, Nanjing,China 210093
xiaxlei@gmail.com, hhuang@nju.edu.cn, yushuying_278@163.com

Keywords: multiple model views; access control model; security model; security framework

Abstract: With increasing diversity and complexity of the computing environments, various security needs in one system can no longer be met by single access control model at the same time. An operating system should be able to enforce multiple access control models. A Multi-Model Views Security Framework is proposed, which is able to enforce multiple access control model views in operating system flexibly.

1 INTRODUCTION

Various security requirements are coming up with the sharply increased diversity and complexity of the computing environments [Rushby,1992, Saltzer,1973]. To satisfy these security requirements, a variety of security models were proposed in last twenty years. Currently widely-used security models include multilevel security model (BLP [Bell,1975]) and its variants (Biba [Biba,1977], Dion model [Dion, 1981]), Domain and Type Enforcement (DTE) [Walker,1996, Badger,1995], RBAC [Sandhu,1996, Sandhu,1997], and etc. Each of these models aims mainly at one or few security require- ments, such as BLP aiming at the confidentiality assurance, Biba aiming at integrity assurance, DTE aiming at confining the information flow channels [Rushby, 1992].

Previous operating system usually enforced only one kind of access control model, for instance, Multics [Organick, 1972] implemented only BLP model in it. However, as mentioned above, the security goals in different applications are various. These different security requirements result in different security models needed for them. How operating system to support this kind of multiple security model views needs?

As a policy neutral security model, RBAC provides a valuable level of permission abstraction. However, using RBAC to simulate MLS or DAC models [Osborn, 2000] is over complex and therefore unpractical in real-world operating system.

The Multi-Model Views Security Framework (MMVSF) is proposed. Several access control models are embodied in MMVSF, including BLP, Biba, DTE and RBAC. These classical models can be easily enforced in MMVSF to implement multiple access control model views in system.

The remainder is organized as follows. Section 2 formally describes the MMVSF. Section 3 gives the examples of enforcing multiple access control model views. And section 4 is the conclusion.

2 MMVSF

2.1 The framework overview

The architecture of the MMVSF is shown in fig-

*This research is supported by National Natural Science Foundation of China under grant 60473093.

ure 2.1. MMVSF comprises of elements, relations and mappings. A *user* in the framework is a system user. A *role* is a job function or job title within some associated authority. Subjects are active entities. *Objects* are resource objects. *Domain* is a control access attribute associated with each subject. And *type* is the other control attribute associated with objects. *Permission* is an approval of a particular mode of access to object or interaction to subject. *Security label* contains a *confidentiality label* and an *integrity label*.

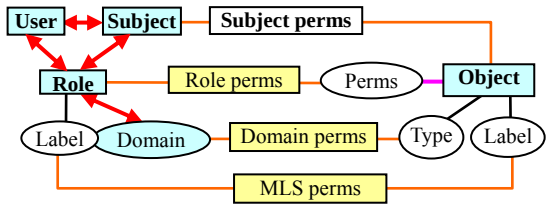

Figure 2.1 the MMVSF.

There are several relations and mappings between elements. *user-role* assignment relation, *user-subject* relation, and *subject-role* mapping figure out the relations between users, roles and subjects. Permissions in system can be authorized to roles, which are given in *role-permission authorization* relation. *Role-domain authorization* relation gives the authorized domains of each role. Each subject has only one running domain, which is given in *subject-domain* mapping. Besides, each role has a security label, and subject's security label is determined by its running role. Each object has a *type* and *security label*.

The *Final Permissions* the subject gets are based on three kinds of permissions corresponding to that subject: *MLS Permissions*, *Domain Permissions*, *Role Permissions.*

2.2 Formal definitions

Table 2.1 Symbols of Elements Sets in MMVSF.

Set Name	Symbol	Set Name	Symbol
Users	U	Subjects	S
Objects	O	Domains	D
Roles	R	Types	T
Confidentiality Labels	C	Integrity Labels	I

- Security Labels: $SL \subseteq C \times I$
- Access modes: $M=\{read, write, execute, ...\}$.
- Domain transfer operation: *transfer*, denotes subject transfer from one domain to another domain.
- Permissions: $CAP \subseteq O \times M$. $(o, m) \in AP$ denotes permission to access object o in mode m.

Definition 2.2 $US \subseteq U \times S$, *user-subject* relation. Many subjects can run on behalf of one user, but each subject can only have one running user.

- *user*: $S \rightarrow U$, mapping from subject to its running user. $user(s)=u$ if and only if $u \in U \wedge (u, s) \in US$.

Definition 2.3 $UA \subseteq U \times R$, *user-role assignment* relation. Each user can be assigned many roles and each role can be assigned to many users.

- UR: $U \rightarrow 2^R$, mapping from user to its assigned role set: $UR(u) = \{r \in R | (u, r) \in UA\}$.
- SR: $S \rightarrow R$, *subject-role* mapping, from the subject to its running role. Each subject's running role be assigned to its running user: $SR(s) \in UR(user(s))$.

Definition 2.4 RL: $R \rightarrow SL$, mapping from role to its security label.

- Ssl: $S \rightarrow SL$, mapping from subject to its security label. Subject's security label is equal to its running role's label: $Ssl(s)=RL(SR(s))$.

Definition 2.5 $RD \subseteq R \times D$, *role-domain authorization* relation, a many to many relation.

- $RDom$: $R \rightarrow 2^D$, mapping from role to its authorized domains set. $RDom(r)=\{d \in D | (r, d) \in RD\}$.
- $SDom$: $S \rightarrow D$, mapping from subject to its running domain. Subject's running domain must have been authorized to its running role, which means: $SDom(s) \in RDom(SR(s))$.

Definition 2.6 object's *security attribute*

- *OT*: $O \rightarrow T$, mapping from an object to its *type*.
- *OL*: $O \rightarrow SL$, from an object to its *security label*.

Definition 2.7 $RCAP \subseteq R \times CAP$, *role-permission authorization* relation. (r1, cap)$\in RCAP$ denotes role r1 has the *Role permission* cap.

- ◆ *Rolecap*: $R \rightarrow 2^{CAP}$, role's authorized permissions set. *Rolecap*(r)={cap| (r,cap) $\in RCAP$}.

Definition 2.8 Two control matrixes

- *DTM*: $D \times T \rightarrow 2^M$, *domain-type access control matrix*. $m \in DTM(d, t)$ denotes subjects in domain d can access objects with type t in mode p.
- *DDI*: $D \times D \rightarrow \{\Phi, \{transfer\}\}$, *domain interaction control matrix*. $transfer \in DDI(d_1, d_2)$ denotes subjects in domain d_1 can transfer into domain d_2.

Definition 2.9 Multilevel Security rule: *Mls_rule*: $SL \times SL \rightarrow 2^M$, $a \in Mls\_rule(l_1, l_2)$ implies subjects with security label l_1 can access target objects or subjects with security label l_2 in mode a. All of BLP and Biba security rules are implemented in this mapping. As a framework, the concrete implementing of this function is not given here.

2.3 Permissions

- *MLS Permission*: $mp(s,o)=\{(o,p)|p \in MLS\_rule(Ssl(s), OL(o))\}$.
- *Domain Permission*: $dp(s,o)=\{(o,p)|p \in DTM(SDom(s), OT(o))\}$.
- *Role Permission*: $rp(s,o)=\{(o,p)|(o,p) \in Rolecap(SR(s))\}$.

A subject's *Final Permissions* on an object is determined as: $fp(s,o)= rp(s,o) \cup (mp(s,o) \cap tp(s,o))$.

3 ENFORCE MULTIPLE MODELS

3.1 Enforcing Multilevel Security model

The way configuring MMVSF to enforce BLP model is described as following:

(1) $I=\{only\_I\}$, there is only one integrity label in system. $|R|=|SL|$, number of roles in the system is the same as the number of the security labels. Each role corresponds to one security label.

(2) $D=\{gen\_d\}$, $T=\{gen\_t\}$, only one domain and type in system. $RD=\{(r, gen\_d)|r \in R\}$, all roles' authorized domain is *gen_d*. all objects' type is *gen_t*: $OT=\{(o, gen\_t)|\ o \in O\}$.

(3) $DTM=\{(d, t, p)|d \in D, t \in T, p \in OM\}$, domain *gen_d* have all *Domain Permissions* to type *gen_t*.

(4) *Rolecap*(r:*R*)=Φ, each role has no *Role Permissions*.

We can use the similar way to enforce Biba model.

3.2 Enforcing DTE

(1) $R=\{gen\_r\}$, one role in system. $UA=\{(u, gen\_r)|u \in U\}$, *gen_r* is assigned to every user.

(2) $RD=\{(gen\_r, d)|d \in D\}$, all domains in system are authorized to the role *gen_r*.

(3) $SL=\{only\_sl)\}$, only one security label in system. $RL=\{(r,only\_sl)|r \in R\}$. $MLS\_rule(only\_sl, only\_sl)=M$, subjects' *MLS Permissions* contain all permitted modes in the set *M*.

(4) *Rolecap*(r:*R*)=Φ.

3.3 Enforcing RBAC

(1) $D=\{gen\_d\}$, $T=\{gen\_t\}$, one domain and one type in system. $RD=\{(r, gen\_d)|\ r \in R\}$, *gen_d* is authorized to every role and all objects' type is *gen_t*, $OT=\{(o, generic\_t)|\ o \in O\}$.

(2) $DTM(gen\_d,gen\_t)=\Phi$, subjects in domain *gen_d* have no *Domain Permissions* to all objects in type *gen_t*.

(3) $SL=\{only\_sl)\}$, only one security label in system. $RL=\{(r,only\_sl)|r \in R\}$. $MLS\_rule(only\_sl, only\_sl)=\Phi$.

3.4 Enforcing multi-model views

Assume all users in system can be divided into three groups: *Grpa*, *Grpb* and *Grpc*. Now we hope that

the model enforced on users in *Grpa* is MLS, on *Grpb* is RBAC and on *Grpc* is DTE. The configuration that enforces this multi-model views in one system is given below.

(1) $U=Grpa \cup Grpb \cup Grpc$, three disjointed subsets.

(2) $R=mls\_rs \cup rbac\_rs \cup \{dte\_r\}$. *mts_rs* is the roles set corresponding to MLS model. *rbac_rs* corresponding to RBAC and *dte_r* to DTE.

(3) $D= \{mls\_d\} \cup \{rbac\_d\} \cup dte\_ds$.

(4) $(u,r) \in UA \wedge (u,r') \notin UA$, where $u \in Grpa$, $r \in mls\_rs$, $r' \notin mls\_rs$, roles in *mls_rs* are only permitted to be assigned to users in *Grpa*. $(u, r) \in UA \wedge (u,r') \notin UA$, where $u \in Grpb$, $r \in rbac\_rs$, $r' \notin rbac\_rs$, roles in *rbac_rs* can only be assigned to users in *Grpb*. In the same way, $(u, dte\_r) \in UA \wedge (u, r) \notin UA$, where $u \in Grpc$, $r \neq dte\_r$.

(5) $|mls\_rs|=|SL|$, number of roles in set *mls_rs* is the same as number of security labels in system. Each role in *mls_rs* corresponds to one security label. $MLS\_rule(Ssl(r),tsl)=\Phi$, $r \in rbac\_rs$, $tsl \in SL$, roles in *rbac_rs* have no *MLS Permissions*. $MLS\_rule(Ssl(\text{dte\_r}),tsl)=M$, $tsl \in SL$, role *dte_r*'s has all of possible *MLS permissions*.

(6) $(r,mls\_d) \in RD \wedge (r,d) \notin RD$, where $r \in mls\_rs$, $d \neq mls\_d$, roles in *mls_rs* are only authorized domain *mls_d*. $(dte\_r,d) \in RD \wedge (r',d) \notin RD$, $r' \neq dte\_r$, $d \in dte\_ds$, all domains in *dte_ds* are authorized to role *dte_r*. Simliarly, $(r,rbac\_d) \in RD \wedge (r,d) \notin RD$, where $r \in rbac\_rs$, $d \neq rbac\_d$, roles in *rbac_rs* are only authorized domain *rbac_d*.

(7) $(mls\_d,t,m) \in DTM$, $t \in T$, $m \in M$. $DDI(mls\_d, d)=\Phi$, $d \in D$, subjects in domain *mls_d* can not *transfer* to any other domains. Similarly, $(rbac\_d, t,m) \notin DTM$, $t \in T$, $m \in M$. $DDI(rbac\_d,d)=\Phi$, $d \in D$.

(8) $Rolecap(\text{r})=\Phi$, $r \in mls\_rs \cup \{dte\_r\}$. *dte_r* and all roles in *mls_rs* have no *Role Permissions*.

4 CONCLUSION

The MMVSF security framework provides a way to easily enforce multiple access control models in an operating system to satisfy the diverse security requirements in one system.

REFERENCES

Badger, L., Sterne, D. F. and Sherman, D. L., et al, 1995. A Domain and Type Enforcement UNIX Prototype. In *Proceedings of the Fifth USENIX UNIX Security Symposium*.

Bell, D. and LaPadula, L., 1975. "Secure Computer Systems: Mathematical Foundations", Technical Report MTR-2547, MITRE Corporation, , Vol. I, MTR-2997 Rev.1.

Biba, K., 1977. Integrity Considerations for Secure Computer Systems. MITRE Corporation, Technical Report MTR-3153.

Dion, L. C., 1981. A complete protection model. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 49-55.

Organick, E., 1972. The MULTICS System: An Examination of Its Structure. The MIT Press.

Osborn, S., Sandhu, R. and Munawer, Q., 2000. Configuring Role-based Access Control to Enforce Mandatory and Discretionary Access Control Policies. *ACM Transactions on Information and System Security*, Vol.3, No.2, Pages 85-106.

Rushby, J., 1992. Noninterference, Transitivity, and Channel-Control Security Policies. Computer Science Lab, SRI International, Technical Report CSL-92-02.

Sandhu, R., Coyne, E., Feinstein, H. and Youman, C., 1996. Role-Based Access Control. *IEEE Computer*. Vol.29, No.2.

Sandhu, R., 1997. Rational for the RBAC96 Family of Access Control Models. In *Proceedings of 1st ACM Workshop on Role-based Access Control*.

Walker, K. M., Sterne, D. F. and Badger, M. L., et al, 1996. Confining Root Programs with Domain and Type Enforcement (DTE). In *Proceedings of the 6th USENIX UNIX Security Symposium*.