A novel approach for distributed updates of MAC policies using a meta-protection framework

M. Blanc +* P. Courtieu * G. Hains * L. Oudot + C. Toinard *
+CEA/DIF *Laboratoire d'Informatique Fondamentale d'Orléans
BP 12 10 boulevard Lahitolle
91680 Bruyères-le-Châtel 18020 BOURGES Cedex
FRANCE FRANCE

Abstract

This paper presents a novel approach where distributed nodes participating to a common infrastructure can modify in a distributed way a Mandatory Access Control policy without any central component. The local modification enables a node first to adapt its configuration to the local applications and second to react to specific attacks that are detected locally. Moreover, a local approach provides a better fault tolerance since the policy does not require any communication with an external node. The general idea is to have a common meta-policy including protection rules plus modification rules. A modification rule enables a node first to modify existing protection rules and second to add new types, roles and users in the system in order to define new rules. So, our approach is to have a local meta-control supporting distributed evolutions of local protection rules. This approach is developed as a joint research project with INRIA and FT R&D, called ACI SATIN, where verification techniques are proposed to verify that the distributed modifications cannot violate the required security properties.

1. Introduction

This paper presents a novel approach where distributed nodes can update locally their MAC policies, e.g. SELinux [1]. In contrast with the other solutions, it enables distributed nodes to update accordingly their local policy while satisfying a common meta-protection. This approach can be used for example in the context of a shared cluster or for distributed applications relying on a set of Internet nodes. The important point is that the framework provides a meta-protection guarantying that the distributed modifications satisfy a control policy telling who has the right to make local modifications and for which attributes and rules. Since the purpose is to be able to guaranty security properties for such distributed evolutions, we present a solution for verifying that the distributed modifications still satisfy some security properties. Typically, we propose to guaranty that there is no information flow for non permitted interactions.

In contrast, traditional MAC systems consider a model where the policy is global and can be distributed at the different nodes on a replication basis. Organization Based Access Control [2] provides a set of policies entities (role, activity, context, view, subject, object, action, organization) that enables to adapt a common policy to different organizations. It is not at all a solution to enable a distributed modification of the protection models since there is no capability to perform local modifications such as adding new entities, removing or modifying existing ones.

2. The meta-protection framework

Our meta-protection framework distinguishes a modification policy from the reusable protection policy. A modification policy defines modification rules using a meta-protection language. These modification rules can easily be projected onto a meta-protection matrix. In turn, the reusable protection policy includes reusable protection rules. Those rules can be expressed with a protection language that can be projected into a protection matrix. The different nodes, participating to the same distributed system, share the modification policy. They share also the reusable protection policy that composes the initial node policy. Since those policies are modified locally without need of any external exchange, two distant policies can have completely different values while satisfying the global security properties that are defined by the modification policy.

2.1. Reusable protection policy

A protection policy defines a permission between a couple of security context. Each security context has three attributes role, type and user. A protection rule gives the

authorization for a specific interaction between two security contexts. For example, an interaction IT for writing on rough device is allowed between a subject security context (R1,T1,U1) and an object security context (R2,T2,U2). So, a protection rule is expressed with a language such as enable(Interaction Type, Subject Security Context , Object Security Context). That type of language can be easily projected onto a protection matrix with SSC as raw, OSC as colon and IT as matrix element. For compact coding, each element of that matrix contains a list of permitted interactions. At the beginning of its lifetime, a node recovers a copy of the reusable protection rules. Then that copy of the protection rules can evolve locally.

2.2. Modification policy

a) Meta-protection for policy modification: a rule enableAddIT(SC, [SSC,OSC]T, PVA) allows a given entity SC to add an authorization for a new interaction IT between the two target security contexts SSC and OSC, in the scope of PVA which is a Permitted Value Array. For example, consider SSC1 (a log process executed by a user_daemon with a specific logging role) where the only permitted interactions are ITa (read authorization) and ITb (link permission) onto OSC1 (a log file belonging to log_admin user with the role admin). There are also primitives such as for permitting the removal or the update of a given interaction. b) Meta-protection for type, role and user: three rules enableCreateT(SC), enableCreateR(SC) and enableCreateU(SC) allow a given entity SC to create either a new type, role or MAC user. For example, the addition of a new application requires the creation of a new type. A user with a security context SSCU installing this new application can be granted the right to add new types so that it can be integrated in the protection policy, by the way of the installer. A rule such as enableCreateT(SSCU) would be added to the meta-protection. Rules are available to cancel the capability of creation. The six rules defined previously apply to three functions: createT(T, attributes), createR(R, attributes), createU(U, attributes). According to the examples given previously, a user installing a new application would call createT(Ta) with Ta being the type of the new application. There are also functions to remove and change, with specific enable/disable rules.

2.3. Policy Adapter

The protection policy, computed from the reusable protection policy using the modification policy rules, is expressed in a generic protection language. A policy adapter takes the rules expressed in that generic protection language as input. It projects the current protection policy onto a target system such as SELinux and GR. Thus, a generic protection language is distributed efficient onto heterogeneous MAC systems.

3. Formal model and verification

Dynamic protection rules raise a much more difficult verification problem than static systems. We now outline our formal model and verification methodology for this general framework. A basic problem to be addressed is the compositionality of sequences of interactions. In the context of dynamically changing protection rules, it is not possible to compute information flow from a set of fixed protection matrices, even if the infinite set of possible protection matrices was taken into account. As a result we model the general information flow problem by considering protection matrix changes at the same level as interaction transitions. This is done by defining the graph nodes as pairs of a security context x with the current protection matrix R. The backward information flow graph is symmetric. The formulation expresses the compositionality of sequences by allowing edges of the second sort to occur at any time. In general this graph is infinite because of the possibility of adding new contexts dynamically. We therefore plan to solve the reachability problem either by theorem proving or application of reachability algorithms for weak infinite-state process algebras.

4. Conclusion

Currently, the language to express the meta-protection rules as been designed but it is still under development. The Policy Adapter has been implemented and validated onto SELinux and grsecurity. The formal model of verification is defined and different methods, theorem proving and process algebras, are evaluated to get the best results. Future work consist in finishing the implementation our meta-protection framework and show the efficiency of the chosen verification systems.

References

[1] P. Loscocco and S. Smalley, “Integrating flexible support for security policies into the linux operating system,” in *Proceedings of the FREENIX Track: 2001 USENIX Annual Technical Conference (FREENIX ’01)*, USENIX, June 2001.

[2] A. A. E. KALAM, R. EL-BAIDA, P. BALBIANI, S. BENFERHAT, F. CUPPENS, Y. DESWARTE, A. MIEGE, C. SAUREL, and G. TROUESSIN, “Organization based access control,” in *4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY’2003)*, (Lake Como, Italie), pp. 120–131, june 2003.