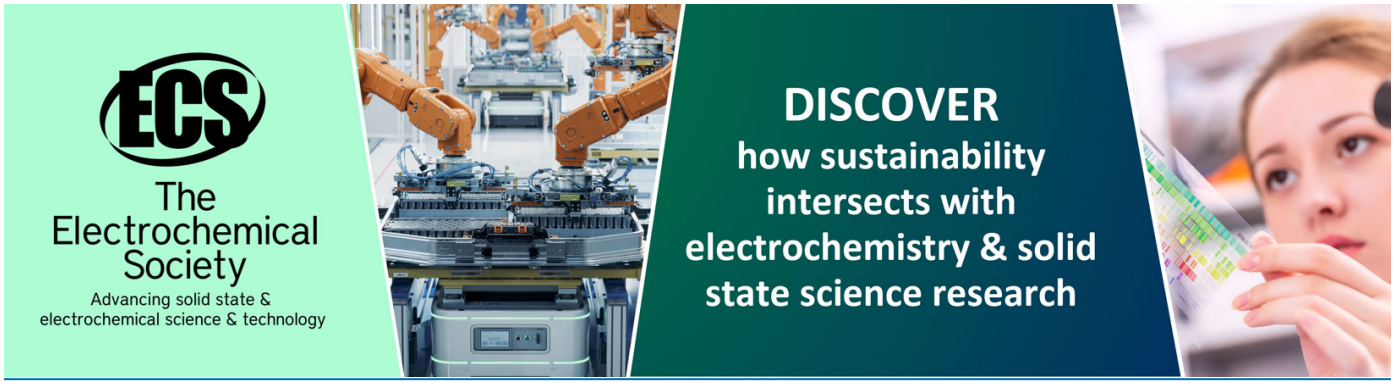
A discerning algorithm for authorization



IOP Conf. Series: Journal of Physics: Conf. Series **1352** (2019) 012040 doi:10.1088/1742-6596/1352/1/012040

A discerning algorithm for authorization

S Y Petrova and S I Eminov

Yaroslav-the-Wise Novgorod State University, ul. B. St. Petersburgskaya 41 173003 Veliky Novgorod, Russia

E-mail: svetayp@list.ru

Abstract. In the article, we consider a discerning authorization algorithm that eliminates the contradictions in the assignment of access rights to security principals in complex infrastructures, which are using concepts such as group and inheritance. It also can use for the identification of security risks and it ensures that the security risk assessments produce consistent, valid and comparable results.

1. Introduction

Every organization faces daily security threats such as: unauthorized use of infrastructure resources. Therefore, when you plan an information security management system, to prevent or reduce undesirable effects, you have to take cognizance of this problem, identify risks and ensure coherent and noncontradictory access rights of one security object to another security object, in other words, make an authorized access system security principals in the infrastructure. A security principal can be any subject that can be authenticated in an organization's infrastructure, such as a user account, a computer or other device account, and a thread or process that runs in the security context of a user account or device. The main problem areas of the authorized access system are the following:

- the process of assigning, enabling, and revoking a security principals account;
- the process of granting and revoking privileges associated with the securable object;
- the process of controlled distribution and using of privileged access rights;
- the process of ranking the inherited access rights of the security principals (if the securable object inherits the privileges from the parent securable object);
- the process of checking the access rights of the security principals at regular intervals;
- the process of removing and correction the security permissions of a securable object when its state or status changes.

In this article we consider the problem of contradictory access rights as effect of ranking the inherited access rights of the securable object and security principal's permissions rank. In [1] described access control model based on discretionary access control list (DACL). When a securable object is created, the system assigns it a security descriptor that contains security information. Normally, when a security principals tries to access a securable object, the authorized system steps through the access control entries (ACEs) in the object's discretionary access control list until it finds ACEs that allow or deny the requested access. The access rights that a DACL allows a user could vary depending on the order of ACEs in the DACL. Consequently, the operating system defines a preferred order for ACEs in the DACL of a securable object. The following steps describe the preferred order [1]:

1. All explicit ACEs are placed in a group before any inherited ACEs.
2. Within the group of explicit ACEs, access-denied ACEs are placed before access-allowed ACEs.

3. Inherited ACEs are placed in the order in which they are inherited. ACEs inherited from the child object's parent come first, then ACEs inherited from the grandparent, and so on up the tree of objects.
4. For each level of inherited ACEs, access-denied ACEs are placed before access-allowed ACEs.

Unfortunately, the proper order of ACEs is complicated by the introduction of object-specific ACEs and automatic inheritance. The Fig.1 demonstrates us how the system of granting access rights based on the inheritance and ranking process of the inherited access rights of the securable object often creates security tokens with conflicting permissions. The following illustration shows how file Will_GUEST_read_it.txt has an object's DACL which can allow and deny FullControll access simultaneously.

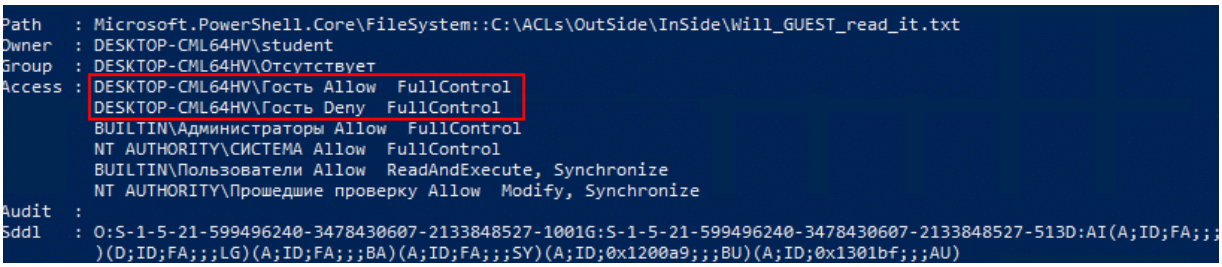


Figure 1. Security token with conflicting permissions.

Generally, you can control access to a securable object by using Allow access ACEs and don't use a Deny access to an object. The shown problem is not exception, because if an ACE allows access to a group and you want to deny access to a member of the group, and place a Deny access after the Allow access for the group, then the system reads the group's access allowed ACE first, and it will grant access to the restricted user. The order of the ACEs is important because the system reads the ACEs in sequence.

To eliminate described problem, it is proposed to create a fundamentally different system of authorization of security subjects based on the logical recognition algorithm. The using of logical methods to the formalization of the problem of recognition, to the choice of the decision rule, to the construction of computational algorithms and to the estimation of the probabilities of correct and erroneous decisions is rational only when there is no a priori information about the quantitative distribution of subjects by spatial, time, weight, energy or any other intervals in the feature space, but there is only data on deterministic logical connections between objects and their attributes. Logical methods allow us to present this information in the form of Boolean relations, reflecting the causal relationships between the classes of objects under consideration and their attributes, to apply well-known efficient algorithms for solving Boolean equations to determine whether a recognized object belongs to one of the classes. This method involves the selection of atomic elements of recognizable objects and the definition of relations between them, reflecting their structure. The mathematical apparatus of the logical recognition algorithm is the theory of formal grammars, described in [2, 3, 4].

2. Results and Discussion

All securable objects arrange their access rights by using the model of basic permissions are assigned on permissions in each case, shown in the following table 1.

Specify $Y_\alpha(\alpha = 1,2,\dots)$ as the type of access right forming the series of values of access rights for the i-th access level $(i = 1 \div n)$, for example, Y_1 - Execute File, Y_2 - Read Data, , Y_3 - Read Attributes, Y_4 - Read Extended Attributes, Y_5 - Write Data, Y_6 - Append Data, Y_7 - Write Attributes, Y_8 - Write Extended Attributes, Y_9 - Delete Subfolders and Files, Y_{10} – Delete, Y_{11} - Read Permissions, Y_{12} - Change Permissions, Y_{13} - Take Ownership, Y_{14} - Synchronize.

Then applying the symbolism of the propositional calculus, for example, we write the rule for granting Basic Read rights to the securable object:

$$\mathcal{R}_1 = (\bar{Y}_1 \cdot Y_2 \cdot Y_3 \cdot Y_4 \cdot \bar{Y}_5 \cdot \bar{Y}_6 \cdot \bar{Y}_7 \cdot \bar{Y}_8 \cdot \bar{Y}_9 \cdot \bar{Y}_{10} \cdot Y_{11} \cdot \bar{Y}_{12} \cdot \bar{Y}_{13} \cdot Y_{14}) \quad (1)$$

where we defined the first level of granting access rights $\mathcal{R}_1$, which can consist only from reading data, attributes, permissions and synchronization ($Y_2, Y_3, Y_4, Y_{11}, Y_{14}$) and no other right (the bar above the letter denotes negation, the point between the statements denotes a logical multiplication operation corresponding to the join rules by the logical AND).

Table 1. Permissions relationship.

Permissions	Basic Full Control	Basic Modify	Basic Read & Execute	Basic List Folder Contents	Basic Read	Basic Write
Execute File	X	X	X	X		
Read Data	X	X	X	X	X	
Read Attributes	X	X	X	X	X	
Read Extended Attributes	X	X	X	X	X	
Write Data	X	X				X
Append Data	X	X				X
Write Attributes	X	X				X
Write Extended Attributes	X	X				X
Delete Subfolders and Files	X					
Delete	X	X				
Read Permissions	X	X	X	X	X	X
Change Permissions	X					
Take Ownership	X					
Synchronize	X	X	X	X	X	X

As a result of combining all the access rights granted to a securable object, we have a complex rule as Boolean equation, which was created from elementary rules using multiplication, addition or negation operations. When a securable object has multiple access rules, the rules are combined using the logical OR (the sign "+" denotes logical addition), which means: "either only one of the combined rules, or two of them, or together".

Based on Table 1, we can get the following implications, denoting the construction of the form IF THEN:

$$(\mathcal{B}_1 = Y_1) \to (\bar{\mathcal{R}}_5, \bar{\mathcal{R}}_6) \tag{2}$$

Let the rules $\mathcal{R}_1, \dots, \mathcal{R}_l$ mean permissions or rights to actions for securable objects.

Let the rules $\mathcal{B}_1, \dots, \mathcal{B}_n$ registered during authorization permission or access rights are features, for certain sets of which (meaning the implicant of the functions $\mathcal{R}_1, \dots, \mathcal{R}_l$ or $\bar{\mathcal{R}}_1, \dots, \bar{\mathcal{R}}_l$), you can found some and the absence of other rights for actions analyzed securable object. A priori information about them, expressing, on the one hand, the relationship between the rules $\mathcal{R}_1, \dots, \mathcal{R}_l$ and $\mathcal{B}_1, \dots, \mathcal{B}_n$, on the other – dependencies only between the elements $\mathcal{R}_1, \dots, \mathcal{R}_l$ or only between the elements $\mathcal{B}_1, \dots, \mathcal{B}_n$ in the general case can be represented in the form of a single equivalence relation of the following form:

$$\mathcal{E}(\mathcal{B}_1, \dots, \mathcal{B}_n; \mathcal{R}_1, \dots, \mathcal{R}_l) = \mathcal{J}, \tag{3}$$

where $\mathcal{E}$ is a Boolean function.

Recognizing the security level of the securable object means to define, based on a priori dependencies (3) and experimental data on the features $\mathcal{B}_1, \dots, \mathcal{B}_n$, which rights from among $\mathcal{R}_1, \dots, \mathcal{R}_l$ this object has,

and which does not. In accordance with the fact that the properties of objects are characterized by $\mathcal{J}$ elements $\mathcal{R}_l$, there can be at most 2^l object types of different access rights.

Suppose that as a result of the authorization process, some data were obtained regarding the truth values of the $\mathcal{B}_1, \dots, \mathcal{B}_n$ features, that characterize the object being recognized, and that this data is represented as a Boolean function

$$\mathcal{G}(\mathcal{B}_1, \dots, \mathcal{B}_n) = \mathcal{J} \quad (4)$$

where $\mathcal{J} = 1$ if the security principals is considered authorized.

Methodically, solving the problem in the above formulation reduces to finding the unknown function $\mathcal{F}(\mathcal{R}_1, \dots, \mathcal{R}_l)$, associated with the function $\mathcal{G}(\mathcal{B}_1, \dots, \mathcal{B}_n)$ by implicant

$$\mathcal{G}(\mathcal{B}_1, \dots, \mathcal{B}_n) \to \mathcal{F}(\mathcal{R}_1, \dots, \mathcal{R}_l) \quad (5)$$

given that the rules $\mathcal{B}_1, \dots, \mathcal{B}_n$ and $\mathcal{R}_1, \dots, \mathcal{R}_l$ satisfy the constraints (3), (4).

The function $\mathcal{F}(\mathcal{R}_1, \dots, \mathcal{R}_l)$ represents the conclusions that can be drawn regarding the properties of the $\mathcal{R}_1, \dots, \mathcal{R}_l$ securable object based on a priori information (3) and additional information (4).

If the conclusions about the properties of the securable object contained in the statement $\mathcal{F}(\mathcal{R}_1, \dots, \mathcal{R}_l) = \mathcal{J}$, are incomplete, for example, $\mathcal{F}_1(\mathcal{R}_1 \cdot \dots \cdot \bar{\mathcal{R}}_{i-1} \cdot \mathcal{R}_i \cdot \dots \cdot \mathcal{R}_l)$ (i.e. there is no information about the rule $\bar{\mathcal{R}}_{i-1}$), then you need to set the well-becoming function, which is a possible comprehensive conclusion about the access rules of the object and find the unknown function $\mathcal{G}_1(\mathcal{B}_1, \dots, \mathcal{B}_n)$, which is associated with the given function $\mathcal{F}_1(\mathcal{R}_1, \dots, \mathcal{R}_l)$ the following dependency

$$\mathcal{G}_1(\mathcal{B}_1, \dots, \mathcal{B}_n) \to \mathcal{F}_1(\mathcal{R}_1, \dots, \mathcal{R}_l) \quad (6)$$

and represents all the first implicants of the function $\mathcal{F}_1$, provided that the elements $\mathcal{B}_1, \dots, \mathcal{B}_n$ and $\mathcal{R}_1, \dots, \mathcal{R}_l$ are superimposed by the constraints (3).

After the function $\mathcal{G}_1(\mathcal{B}_1, \dots, \mathcal{B}_n)$ is defined, it is necessary to compare the functions $\mathcal{G}(\mathcal{B}_1, \dots, \mathcal{B}_n)$ (4) and $\mathcal{G}_1(\mathcal{B}_1, \dots, \mathcal{B}_n)$ with each other and determine which features need to be identified in addition to the existing ones , in order to obtain the full characteristic of a recognizable securable object as a function $\mathcal{F}_1(\mathcal{R}_1, \dots, \mathcal{R}_l) = \mathcal{J}$. After that, the missing features can be searched for purposefully.

When the security principal tries to gain access to the securable object, the system scans the access control entries in the access control list at the object level until it finds access control entries that allow or deny the requested access. If, during the authorization, data is received not for all the missing attributes of the subject, but only for some of these features, then the first task needs to be solved again and, in accordance with (5), narrow down our conclusions about the access rights of the authorized subject, etc. After finding all the missing features, the subject is considered authorized

3. Conclusion

The logical recognition algorithm enables you to control the ability of a process to access securable objects or to perform various system administration tasks.

Firstly, the logical recognition algorithm for any presented set of rules of access permissions to the securable object allows to determine whether this set is correct or not, and in the case of a positive answer, gives instructions on the structure of this set.

Secondly, the logical recognition algorithm allows you to build any regular set of rules, while giving instructions on its structure, and does not construct any irregular set of rules.

In the first case, the formal logical recognition algorithm we called discerning authorization algorithm, in the second, the generating one.

References

[1] Parts of the Access Control Model on https://docs.microsoft.com/en-us/windows/desktop/secauthz/access-control-components. (accessed 26.06.2019)
[2] Gladkii A Vand Dikovskii A Ya 1974 Theory of formal grammars *J. Soviet Mathematics* **2 (5)** 542–563
[3] Gross M and Lentin A 1971 *The Formal grammar theory* Publisher: Mir Moscow
[4] Bresnan J 1982 *The mental representation of grammatical relations* Cambridge Massachusetts: The MIT Press