https://doi.org/10.52326/jes.utm.2021.28(2).05
UDC 004.056

TRUST- BASED MODELING MAC-TYPE ACCESS CONTROL THROUGH ACCESS AND ACTIONS CONTROL POLICIES

Marcel Danilescu, ORCID: 0000-0002-6561-7955,
Victor Beşliu*, ORCID: 0000-0001-7265-903X

Universitatea Tehnică a Moldovei, Bd. Ştefan cel Mare, 168, Chisinău, Republica Moldova
*Corresponding author: Victor, Beşliu, *victor.besliu@ati.utm.md*

Received: 03.07.2021
Accepted: 05.06.2021

Abstract. In recent decades, the number of researches on access control and user actions in computer systems has increased. Over time, there have been two models of implementing Mandatory Access Control (MAC) policies for government institutions and Discretionary Access Control (DAC) for the business environment, policies that various access control modeling solutions seek to implement. Among the access control modeling solutions developed are Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC), presented in the U.S.A. by the National Institute of Standard and Technology (NIST). In Romania, in 2010, the access control solution based on trust was presented. This paper presents Mandatory Access Control policy modeling using the trust-based access and actions control modeling solution.

Keywords: *domain, objects, organizations, relation, rules, security, users.*

Introduction

For starters, we will do a trust-based presentation of access and action control. Starting from the requirements of implementing an access control policy for virtual organizations, ad-hoc, following the studies, a new approach was proposed to restrict users' access to data and information within a computing system. To present this approach to creating trust-based access control policies we need to define the following:

- the domain of activity,
- the object,
- group of objects,
- the life cycle or lifespan of an object,
- users,
- user groups,
- delegation,
- restriction,
- the level of trust given to action,
- necessary criteria for establishing the value of trust.

The trust value is given to a user, or group of users, for a particular field of activity, or one or more objects in the field of activity.

The domain of activity is a part of the performed activities, grouped according to common characteristics, such as common technical, economic, or scientific knowledge, common interest, scope, etc.

The object represents a homogeneous and unitary entity of information on electronic support, on which the actions are carried out to achieve the purpose for which it was created.

The group of objects is a collection of objects that have common characteristics or complement each other to create a description as accurately as possible on reality and belonging to a field of activity.

One can encounter the situation when an object or group of objects could belong to several fields of activity, which are modeled within a computer system.

To establish the affiliation of these groups to one domain or another, we will consider that the object or group of objects belongs to the domain that has the most interactions with it and possibly ends their life cycle.

The life cycle (the lifespan of an object) is the totality of the steps an object goes through, from creation to archiving or deletion.

The user is the person who interacts with objects during their existence and performs various actions.

The user group consists of people who interact with a set of objects within a field of activity.

Delegation is the transfer of rights and obligations of one user to another user.

The restriction is the revocation of some user's rights.

Definition: We call a trust value, a value assigned to a user or application that corresponds to an action taken on an object.

The criteria necessary to establish the trust value are an arbitrary set of conditions that a user must meet to be granted a certain trust value to perform certain actions [1].

The access of trust level is a value assigned to an action that can be applied to an object or group of objects and for which a user or group of users receives permission to execute the action.

To create a logical mechanism for controlling access to objects, we will formalize the principles set out above. For this, we will make the following considerations on the elements we will work with.

We define a hierarchy as a finite set of finite values ($H1 \leqslant H2$) [2 - 8].

We define a sub-hierarchy ($I1 \leqslant I2$) as a subset of a hierarchy ($H1 \leqslant H2$) if ($I1 \leqslant I2$) $\subseteq$ (H1 $\leqslant H2$) [2 - 8].

There is the possibility of interaction between objects and users, i.e., a user can perform certain operations on an object like Read, Create, Write (update), Append , Copy, Rename, Delete, Archiving, Approval, Etc.

We call the interaction between the object and the user action and we write it down with a_i. The totality of the actions creates the set of actions.

We define a relation as a connection that exists between two elements x and y that belong to disjoint sets and that can be expressed in the form (r, x, y).

A relationship of trust is a relationship that can be quantified by values between a minimum and a maximum value, values arbitrarily set, which correspond to the levels of trust used from "no trust" to "blind trust."

If $\boldsymbol{r} = 0$, there is no trust relationship between $\boldsymbol{x}$ and $\boldsymbol{y}$, and when $\mathbf{r}$ x $\boldsymbol{v}$= *max* (*value*), the trust is full. Between these values that represent the two extremes of the relationship, various actions can be defined that can be applied to the elements, depending on the value of the trust relationship applied to a user or group of users, for an element or category of elements.

Definition: An action "*a*" of "*x*" on "*y*" can only take place if the value of the relationship between "*x*" and "*y*" meets the minimum condition required to perform the action [2 - 8].

Thus: if $\boldsymbol{r} = 0 \vee rel(\boldsymbol{r}, \boldsymbol{v}) = 0$ ($\boldsymbol{v}$ = the minimum condition for which $\exists \boldsymbol{a}$) $\Rightarrow \neg \boldsymbol{a}$, otherwise $rel(\boldsymbol{r}, \boldsymbol{v}) \Rightarrow \boldsymbol{a}$.

Therefore, the control of actions "$\boldsymbol{a}$" can be based on the value assigned to "$\boldsymbol{r}$".

If "$\boldsymbol{r}$" has a value equal to or greater than "$\boldsymbol{v}$", and the condition is as "$\boldsymbol{r} \geq \boldsymbol{v}$" then "$\boldsymbol{v}$" = the minimum required to perform an action, therefore "$\boldsymbol{r}$" corresponds to all actions whose value is more than small as r or equal to "$\boldsymbol{v}$". If no value is set for "$\boldsymbol{r}$", then it is considered "$\boldsymbol{r} = 0$".

If the condition for performing an action is "$\boldsymbol{r} \leq \boldsymbol{v}$", then "$\boldsymbol{v}$" = the maximum required to perform an action.

Given that "$\boldsymbol{v}$" = the strict value for which $\exists\ \boldsymbol{a} \Rightarrow \neg\ \boldsymbol{a}$, otherwise $\boldsymbol{r} \equiv \boldsymbol{v} \Rightarrow \boldsymbol{a}$. So the action can be executed only if $\boldsymbol{r} \equiv \boldsymbol{v}$. We can therefore define the conditions necessary to apply a confidentiality policy.

Let $O_i \in GO$, $GO \in Di \wedge P_i \in P$ where $P_i=(p_1,p_2,\ldots p_k \ldots p_n)$, and $p_k=H_k(A_k) \cup H_k(E_k) \cup \sum F_k$ for $\forall A_k$, $\exists(U_k \in G_m \Rightarrow \exists R_u, R_u(U_k) = R_a(A_k) \wedge R_u(U_k) = \leq R_g) \oplus \exists(U_x \in G_m \Rightarrow \exists R_u, R_u(U_x)=R_u(U_k) \wedge \exists de_v(U_k)$ for $U_x) \oplus \exists(U_x \in G_m \Rightarrow \exists R_u, R_u(U_x)=R_u(U_k) \wedge \exists de_v(U_k)$ for $U_x \Rightarrow re_v(U_k) \in RE \wedge \neg\exists re_v(Ux) \in RE\ )$

Where:

A_k = an action applied to an object,

de_v = delegation received from any user U_k,

DE= the multitude of delegations,

D_i = The domain to which an object belongs,

F_k = flow sequences,

G_m = The user group to which any U_k user belongs,

GO = Group of objects,

$H_k\ (A_k)$ = the hierarchy of actions corresponding to the subprocess p_k,

$H_k\ (E_k)$ = hierarchy of events,

O_i = Object *I*,

P_i = The process suffered by O_i,

$p_1..p_n$ = the set of O_i threads,

R_u = confidence value for user U_k to perform an A_k action on the O_i object,

R_g = confidence value for the group G_m,

R_a = the confidence level required to act A_k on the object O_i,

re_v = restriction applied to a user U_k,

RE = set of restrictions,

U_k = the user designated to perform the A_k action,

U_x = any user belonging to the group G_m.

In the above, we presented the condition for creating a policy based on trust. Its purpose is to determine the conditions for making tuples of the form $\{D_i, O_i, Uk_x, (Ak_x, rev_x, dev_x), R_u\}$, which are trust-based access control policies applied to a user at a given time, to control its actions and which can be applied in shaping the various types of standardized policies.

For objects, the corresponding tuples are of the form $\{D_i, O_i, A_k, R_a\}$.

MAC policy modeling

Documents used in the areas of national, military, and government security are labeled according to their classification levels. In general, they range from Unclassified (anyone can see this) to Secret Service, Secret, Strict Secret, Secret of great importance. These levels correspond to the risk associated with the communication of information. Each level of classification can be associated with a scalar that represents a reliable value.

To ensure the information, not only the classification levels of the objects are used, but their compartmentalization by areas of interest is used. Each object is associated with a field, and if it is desired to respect the principle Need to know- access to information should be granted only if it is necessary to perform the duties. (The need to know is an instance of the least privileges.) Domains are used to manage this breakdown of information. For example, an object associated with the fields {biology, genetics} can only be accessed by subjects who need to know about both biology and genetics. An object associated with the empty set {} simply does not refer to any required domain.

A label is a tuple that describes a level of information classification and a set of domains. A document can be labeled (Secret of great importance, {biology, genetics}) if it contains extremely sensitive information about biology and genetics. In practice, each paragraph of a document is assigned a set of areas and a sensitivity. The classification of the whole document would then be the most restrictive classification given to a paragraph in that document.

Users are also labeled by the security clearance. User authorization, like the label of a document, is a tuple consisting of the classification level and a set of domains.

MAC policy models mainly follow the following rules:

1. no read up
2. no write down

Using trust-based access control and action policies to implement MAC policies, we need to add the following two rules:

for $\forall O_i \Rightarrow \exists V_o,\ \forall U_k$ *with* $C_u \geqslant C_o \Rightarrow A_k$ *where* A_k= *Read*

for $\forall O_i \Rightarrow \exists V_o,\ \forall U_k$ *with* $C_u < C_o \Rightarrow \neg A_k$ *where* A_k=*Write*

where:

- C_u = U_k classification label
- C_o = O_i classification label

Implementing MAC policies

In the following, we will present a model for implementing MAC policies for a computer system *Si*, which must serve several groups of users $G_m(x)$ where $x = (1, \ldots, n)$, who have access to objects from various areas of access with the help of trust-based access control and actions.

Access control policies and trust-based actions require the existence of $\{D_i, O_i, A_k, R_a\}$ tags attached to objects and $\{D_i, O_i, Uk_x, (Ak_x, rev_x, dev_x), R_u\}$ attached to users.

Under these policies, the C_u and C_o classification labels will be replaced by R_u (A_k) and $R_a(A_k)$, respectively.

An object in a domain to which a MAC policy applies, in addition to the privacy label, the domain to which it belongs, is attached for each action it can support, a trust value that a user must-have to be able to perform a certain action: Sd = the set of defined areas of interest; $Sd = \{D_i\,(i)\}$, $i = (1,\ldots, n)$ and $D_i\,(i)$ = a domain of interest

For $\forall O_i \subset\ D_i(i) \Rightarrow \exists\ R_a, R_a$ = the confidence level required to withstand the action of A_k.

Therefore, for the O_i element, we must have attached a tuple formed the level of sensitivity (domain confidentiality), the domain, and the level of trust attached to the action to be applied to the object.

Thus, for a user to be able to apply a certain action on an object in addition to the label with the security level he has for that domain, he also has attached the trust level for the action for which he is authorized, and his actions must take into account the rules presented above.

That is, for $A_k(O_i) \Rightarrow \exists GO(x)$ having attached $\{D_i\,(i), R_g\}$ where $A_k \subset H_k$, $H_k=\{\varnothing, A_k(n)\}$.

Therefore, we can say that:

For $\forall A_k\,(O_i),\ O_i\ \{D_i,R_a) \Rightarrow \exists\ (U_k, U_k\,\{\ D_i,R_k\} \in G_m) \vee \exists(U_x\{D_i,\ R_{k(x)}\} \in G_m \wedge \exists dev_v(U_k)\ for\ U_x) \vee \exists(U_x\{D_i,\ R_{k(x)}\}\ ,\ R_{k(x)}(U_x)=R_k(U_k) \wedge \exists dev_v(U_k)$ for $U_x \Rightarrow rev_v(U_k) \in RE \wedge \neg\exists rev_v(Ux) \in RE\ ) \oplus \exists\ (U_x\,\{D_i,\ R_{k(x)}\})$ for $R_k \geq A_k \wedge Rk_{(x)} \geq A_k$

- if A_k = "*read*" then $U_k \vee U_x \Rightarrow R_a \leq R_u$;
- if A_k = "*write*" then $U_k \vee U_x \Rightarrow R_a \geq R_u$.

Implementation of the invocation policy: Let S is a set of applications that have attached a confidence level and a domain of applicability i.e. $\forall s_i \in S \Rightarrow \exists\{D_i, R_s\}$, $\forall U_i\{D_i, R_u\}$ can also execute if $R_s \leq R_u$ and $D_i(s_i) = D_i(U_i)$. A s_i can interact with an O_i, if and only if $R_s \geq\ R_a$.

System architecture

To implement the system, we will create:

1. a Policy Creation Point
2. a Policy Storage Point
3. a Document Status Point
4. a Document Storage Point

where:

The policy creation point is where the person responsible for implementing access control policies and actions creates policies for each application that accesses documents from the document storage point.

The storage point for policies is where they are stored and are accessed by each application after the user has been logged in.

The status point of the document represents the point where each document to be processed by a user, has recorded the hierarchy of actions that can be applied to him, and the action to be applied to him, the active action at that time.

The document storage point is the location where the documents are located and will suffer from user actions.

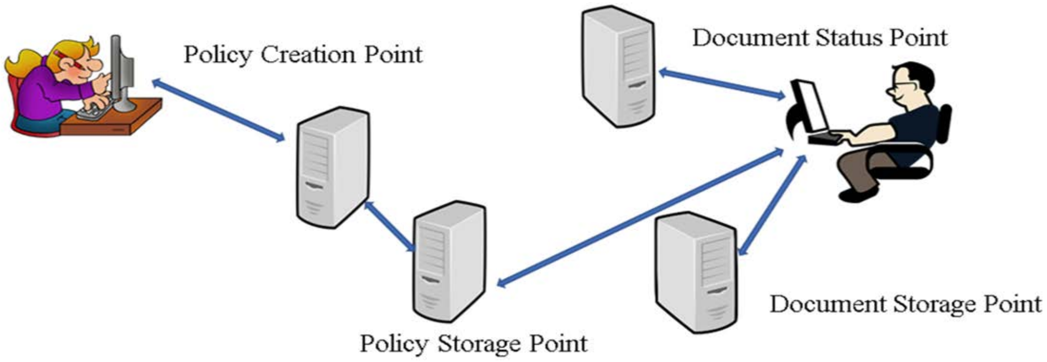

Figure 1. System architecture.

Thus, the person responsible for implementing access control policies and user actions (Figure 1), updates the policies, which are stored in the Policy Storage Point. The user launches the application that asks him to authenticate. It authenticates and the application downloads the policies that apply to the user. From the Document Storage Point download the documents to which it has access. After updating them after the application interacts with the user, they are stored in the Document Storage Point and the status of the document in the hierarchy of actions that it must support is updated in the Document Status Point.

By the status of the document, it is possible to determine the route on the book the document had and, what is its next stage.

Changing the status of the document is important, as it does not allow access by a user within the user hierarchy, which does not correspond to the hierarchy of actions that apply to the document. And the changes made to a user with higher credentials cannot be noticed by a previous user.

In this way, the two important conditions of the MAC model:

1. no read up
2. no write down,

are respected.

Creating the hierarchy of actions that can be applied to objects.

In [9] the importance of the workflow for designing access control policies and actions was presented. An example of workflow modeling was taken, starting from a request for rest leave where the importance of the workflow was shown. That example was for some organizations. Creating the action hierarchy allows for two things:

- creating access control policies and user actions,
- create the status vector for a document type.

Note that the status vector created is a template for that document type.

When creating a new document, a document status vector is created based on the projected model. If in paper [3] we presented how to build access control policies and actions, in this paper we will detail how to build the state vector and its importance in modeling MAC policies.

We will present a workflow model for approving a loan in a lending organization.

The applicable legislation in the field is European legislation on the protection of personal data (GDPR) [10], legislation on banking secrecy EMERGENCY ORDINANCE no. 99 of 6 December 2006 [11] and Moldavian Law 195 of 15-07-2010 [12].

Legislative conditions do not allow the data to be made public and the staff involved to ensure the confidentiality and integrity of customer data.

The table 1 shows the stages of the workflow and Figure 2 shows graphically the workflow.

Table 1

The stages of granting a loan

Step	Actor	Object	Action
1	Client	Request	Completion of credit application, submission of documents, and credit guarantees.
2	Credit officer	Request, file	Application registration, opening credit files for the client.
3	Credit officer	File	Verification of credit documents file/request for additional documents.
4	Head of credit bureau/ service	File analysis (grant / rejection report)	File analysis, proposal granting, or rejection of file.
5	Organization leadership	Decision to grant / reject	Approval/rejection of credit file.
6	Organization leadership	Credit account opening decision	In case of approval, a credit account is opened.
7	Organization leadership	Notification	Notify customer approval or file rejection

The following steps are required to approve a loan:

1. The client goes to the lending organization where he requests a loan that the lending agent registers.
2. Along with the request, the client brings certificates from the employer regarding the duration of the employment contract and the income he has.
3. The credit officer registers the application, takes over the attached documents, and opens (creates) the credit file.
4. Then analyze the documents and depending on the creditworthiness of the client and the amount requested, may require additional guarantees consisting of goods or guarantors. Also, check the client's history in the C.R.C. (Credit Risk Center) following the possible records about the requesting client.
1. After setting up the guarantee requested by the organization, the credit officer forwards the credit file to the head of the credit bureau/service with a proposal for approval or rejection and the history of credit risk incidents.
2. The head of the office/service in turn proposes the approval or rejection of the loan by the bank's management after analyzing the file.
3. The bank's management approves or disapproves the loan.
4. Under the conditions of credit approval, a credit account is opened for the client and he is replenished with the required amount.

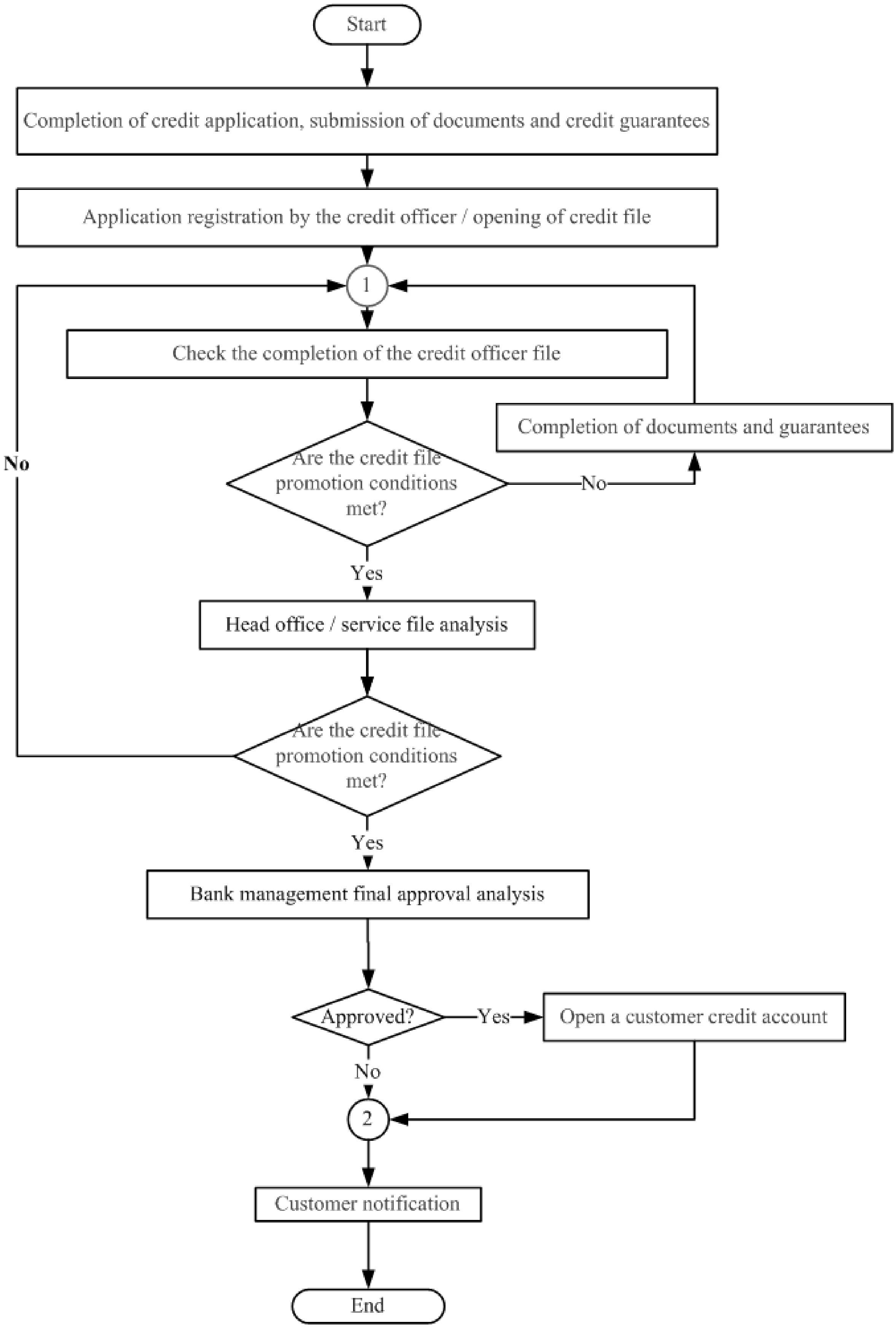
Start
Completion of credit application, submission of documents and credit guarantees
Application registration by the credit officer / opening of credit file
1
Check the completion of the credit officer file
Completion of documents and guarantees
No
Are the credit file promotion conditions met?
No
Yes
Head office / service file analysis
Are the credit file promotion conditions met?
Yes
Bank management final approval analysis
Approved?
Yes
Open a customer credit account
No
2
Customer notification
End

Figure 1. Workflow - graphical representation.

Analyzing the steps in Table 2, we can now create the status vector for the credit-granting computer application. For this, we will create a structure of the form, “stage, object, action, status”, where status can have one of the forms, active/inactive, true/false… etc.

Table 2

Object, action, status

Step	Object	Action	Status
1	Application	customer request posting	inactive
2	File	file record	inactive
3	File	check dates/complete dates	inactive
4	credit report	eligibility check	inactive
5	credit/credit decision ratio	credit validation	inactive
6	Account	open account	inactive
7	Decision	customer notification	inactive

The status vector model is in the form of:

Status(1)=”inactive”
Status(2)=” inactive”
Status(3)=” inactive”
Status(4)=” inactive”
Status(5)=” inactive”
Status(6)=” inactive”
Status(7)=” inactive”

The start of the lending process causes the status from inactive to active change and applying user actions to objects involves going through the steps within the workflow. An action cannot be initiated if the previous action has not been completed.

Creating the status vector of objects, and mentioning the existing active state at a given time, allows the implementation of a MAC control policy by imposing the following conditions:

1. at any time of the status vector scrolling can only be active one status value,
2. each stage is consecutive to the previous stage, which implies:
1. no object in the status vector can be accessed simultaneously with another object,
2. once the activity on an object is completed by a user, the object corresponding to the next step is accessed by the appropriate user,
3. the user who can access the object must be part of the user hierarchy corresponding to the hierarchy of objects and the trusted access value must be corresponding to the required confidence value for the object,
4. the current user has the confidence level equal to the trusted value of the object accessed,
5. an object after that has been accessed and various processes have been executed on it, receives the required higher trust value according to its processing flow.
6. Trusted policies for access control and applied actions are strict enforcement **policies** that mean each user can access an object only if it has a confidence level equal to the trusted value of the object accessed.

The objects involved are documents provided by the customer, internal analyses and reports on the customer's creditworthiness and credit history, other documents necessary for the granting of credit, applications built in the accounting of customers, credit applications, amounts credited, and records of payments and arrears.

For our example, the application for a customer record, the evolution of credit demand, and the management of credit documents are required.

The customer record application allows registration (temporary in the first phase, until the credit application is resolved) to the customers of the organization. This application is not accessible to users in different **"credit"** domains.

The application to record the evolution of the credit application allows tracking of the stages of the application, and updating its status vector, and is available to users in the field "credit".

The application for the management of the documents required for crediting shall contain in electronic form the documents submitted, created, and updated during the information flow of their processing.

If the first two applications can be implemented through basic bases where the implementation of user control, domain and sensitivity level (trusted value) is easier to implement through access and action control, the third application can be implemented through messaging systems, and/or shared folders.

In this case, to ensure data protection and control of the user's access and actions, documents must be placed in a labeled container. The label must contain the sender, the scope, the description of the contents of the container, and the level of trust required to access the container. The recipient may or may not be mentioned. The existence of the necessary level of trust restriction prevents documents from being accessed. A possible deployment format in xml is presented below:

```
<?xml version="1.0" encoding="utf-8"?>
    <document_container>
        <etiquette>
            <sender>xxxxxx</sender>
            <description>a.....z</description>
            <domain>yyyyyy</domain>
            <trust_level>n</trust_level>
            <global_action>RW</global_action>
        </etiquette>
        <documents>
            <doc1>
                <name />
                <action />
                <content />
            </doc1>
            <docn>
                <name/>
                <action />
                <content />
            </docn>
        </documents>
    </document_container>
```

The use of containerization allows documents to be kept in a homogeneous space that allows the simultaneous handling and management of all categories of documents belonging to a credit application.

Switching from a lower processing point to a higher one by going through the processing flow hierarchically is done after updating the required level of trust and simultaneously updating the status vector.

This way you can find information about the status of documents and ensure the confidentiality and integrity of documents and the processing process.

Conclusions

This paper outlines how to implement MAC policies in general and BIBA in secondary through access control policies and trust-based actions.

The use of access control policies allows both application modeling, by controlling user access and actions, and information systems in their complexity, allowing a high granularity control over them.

Controlling user access and actions by applying a strict policy (R_u=Ra), allows easy design and implementation of the above-mentioned policies (MAC and BIBA), making it easier for both the analysis and programming part of the information system to be.

This paper was presented to present the advantages of using access control solutions and trust-based actions, not only at the application level, but also at the design level of a computer/ information system and, shaping various types of policies and ensuring the integrity of documents.

References

Journal published papers:

1. *Control Access To Information By Applying Trust Policies.* Danilescu, Marcel și Danilescu, Laura. [ed.] Universitatea "Titu Maiorescu" București. Bucuresti: Universitatea "Titu Maiorescu" București, 2010. Educație și creativitate pentru o societate bazată pe cunoaștere" ediția a IV-a. pg. 49-54. ISBN 978-606-8002-47-7.
2. *Modeling access control and user actions using Trust - Based Access Control Policies*. Danilescu, Marcel. 2, Chisinau: Universitatea Tehnică a Moldovei, 2020, Journal of Social Sciences, Vol. III, pg. 72-84. ISSN 2587-3490, eISSN 2587-3504.
3. Creating Trust Based Access Policies to control users actions on documents. Danilescu Marcel, Beşliu Victor. [ed.] Veaceslav Perju. Chisinau: s.n., 2013. Information Technologies and security 2012. p. 388. ISBN 978-9975-4172-3-5.
4. Assurance model behavior in social networks based on trust. Adomnicăi, Cosmin și Danilescu, Marcel. Chengdu: s.n., 2011. 3rd International Conference on Computer Technology and Development ICCTD.
5. XML based techniques for data privacy in e-business. Danilescu Laura, Danilescu Marcel. Bucuresti : Titu Maiorescu University Publisching House, 2009. Education and creativity for a knowledge society. ISBN 978-606-8002-36-1.
7. Data security management applying trust policies for small organizations, ad hoc organizations and virtual organizations. Danilescu, Marcel. [ed.] Danubius Jurnals. 3, Galați: Danubius Publishing House, 2012, The Journal of Accounting and Management, Vol. 2, pg. 47-64. ISSN: 2392 – 8778 ; ISSN: 2284 – 9459.
8. Control Access To Information By Applying Policies Based On Trust Hierarchies. Danilescu, Laura și Danilescu M. Manila, Philippine: Institute of Electrical and Electronics Engineers, Inc, 2010. International Conference on Computer and Software Modeling, ICCSM 2010. pg. 285-290. IEEE Catalog Number: CFP1093L-PRT ISBN: 978-1-4244-9095-0, IEEE Catalog Number: CFP1093L-ART ISBN: 978-1-4244-9097-4.
9. Control access to information by applying policies based on trust hierarchies. Danilescu, Marcel și Danilescu, Laura. Bucureşti: Universitatea Titu Maiorescu, 2010. Conferinţa internaţională "Educaţie şi Creativitate pentru o societate bazată pe cunoaştere". 2010. pg. 49-54. ISBN 978-606-8002-47-7.

Legal regulations and laws, organizations:

10. Parlamentul European. Regulamentul (ue) 2016/679 al Parlamentului European și al Consiliului din 27 aprilie 2016 privind protecția persoanelor fizice în ceea ce privește prelucrarea datelor cu caracter personal. eur-lex.europa.eu. [Interactiv] accesat: 27.12.2020. eur-lex.europa.eu/legal-content/RO/TXT/?uri=CELEX32016 R0679&qid=1613764859607.
11. Guvernul Romaniei . Ordonanță de urgență nr. 99 din 6 decembrie 2006 privind instituțiile de credit și adecvarea capitalului. Portal Legislativ. [Interactiv] accesat: 27.12.2020. http://legislatie.just.ro/Public/DetaliiDocum ent/78313.
12. Parlamentul Republicii Moldova. LEGE Nr. 195 din 15-07-2010 pentru modificarea şi completarea unor acte legislative. Legis.md. [Interactiv] 15 07 2010. [accesat: 22.12.2020.] https://www.legis.md/cautare/getResults ?doc_id=4584&lang=ro.