Access Control for Newly Created Objects: Principles, Abstract Model and Implementation

Andrey Sheglov, Konstantin Sheglov, Alexander Ogolyuk
Saint Petersburg National Research University of Information Technologies, Mechanics and Optics
St. Petersburg, Russia
xms2007, npp-itb@yandex.ru

***Abstract*—we describe principles and implementation details of access control for newly created objects – like files and clipboard data. This access control mode can be used to enhance the well known access control methods based on access subjects and static access objects. It allows to defend against many actual threats based attacks. The principles of such control mode are based on excluding access object entity from access control policy and inheritance of creator credentials.**

I. INTRODUCTION

The basic contradiction in existing access control and access rights allocation methods lies in access policy implementation for the newly created objects – those objects which do not exist while system administrator creates access rights for static objects. In first place newly created objects are file objects (which are used to store processed data) and temporary data stored in system clipboard.

In this work we introduce completely new principles of access control for newly created objects, which were implemented and approved in working technical solutions. By using this principles currently used system information security approaches could be reviewed again and potentially transformed.

II. NEWLY CREATED OBJECTS ACCESS CONTROL PRINCIPLES

Under "newly created" objects inside working system we understand objects which are absent in access policy configured by administrator. I.e. those objects which are created by user after access policy was configured. Under newly created objects access control we understand the control mode which is based on excluding "access object" entity from access policy. I.e. access object will not be used while creating access policy rules. This can be done through automatic marking of newly created objects with inheritance of the creator credentials.

The base principles of such access control are:

- "Access subject" entity is excluded. Access policy is built basing on just two entities: creator subject identifier (credentials) and access subject identifier (credentials) – i.e. subject (person) who requests access to the object.
- Access control rules are established between two entities: "access subject who requests access to the object" and "access subject who did create the object" (creator owner).
- When access subject creates new object – this object is automatically marked and inherits creator credentials.
- When access to the object is requested – access manager analyzes the presence of credentials (inherited from creator subject). If such credentials are present inside object access manager analyzes access control rules between creator and requestor subjects. If credentials are not present – the "unmarked object" rule is used. I.e. the rule for the object which has no credentials inherited from creator.

The base usage scenario of access control method for newly created objects is isolated (between different access subjects) information processing implementation.

III. ABSTRACT ACCESS CONTROL MODEL

Abstract access control model in current case is completely different than known "Harrison-Ruzzo-Ulman" [1] model. Our model looks like (1).

If we assume that the set C = {C1, …, Cl} is linearly ordered set of access subjects and R = {R1,…,Rm} be a finite set of access rules (read (r), write (w), delete (d), execute (e), etc., access control absence(0)) of subject Ci to the object created by Cj where i = 1…l and j = 1..l then the access control matrix M looks like following one. We agree to indicate requestor subject credentials in the matrix rows and creator or object inherited credentials (identifiers) in matrix columns.

At any given time the system is described by its current state Q = (C,C,M). M[C,C] is the matrix cell containing the access control rules set for requestor access subject to creator access subject. With Ci(R)Cj we define access rights of Ci subject to the object created by Cj subject where i = 1…l and j = 1..l and R = {x,w,r,d}: read (r), write (w), delete (d), execute (e).

ISSN 2305-7254

$$M = \begin{array}{c} \\ C1 \\ C2 \\ . \\ . \\ . \\ C1\text{-}1 \\ Cal \end{array} \begin{array}{c} \begin{array}{ccc} O1 & O2 \ldots\ldots\ldots & Ok \end{array} \\ \begin{bmatrix} r,w,d & w & 0 \\ r & r,w,d & 0 \\ & \ldots\ldots\ldots\ldots\ldots & \\ & \ldots\ldots\ldots\ldots\ldots & \\ 0 & 0 & r \\ 0 & w & r,w,d \end{bmatrix} \end{array}$$

Fig. 1. The access control matrix M

Abstract access control model is used to form requirements for its implementation correctness. The state Q0 = (C0, C0, M0) should be considered safe towards access right R if there is no sequence of actions resulting subject C0 to receive the access right R to the object created by another subject which (right) is not present in the cell of matrix M0[C0,C0]. If the right R, absent in the M0[C0,C0] cell is received by subject C0 then we should consider it the right R leakage and the system is not safe on the right R.

IV. FILE ACCESS CONTROL IMPLEMENTATION

Let's review the formulated principles implementation based on developed security system example. This system works with Windows Operating System [2].

The access subject is defined by three entities – primary user identifier (user account under which the process was started), process name (executable full path), effective user identifier (user account under which the object access is requested). These entities are defined in security system graphical interface (2) and shown inside access subject list in the same system (3).

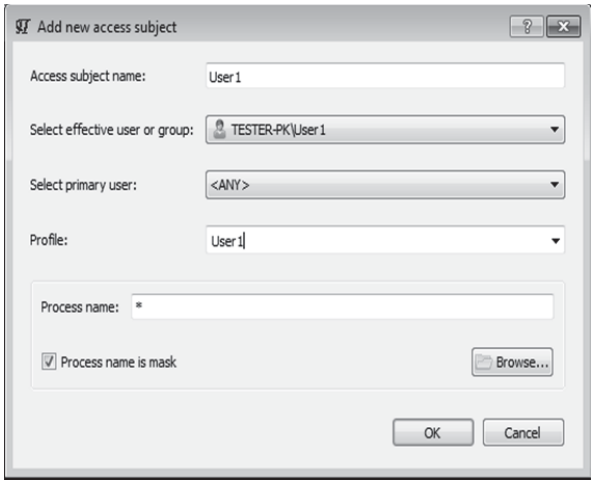
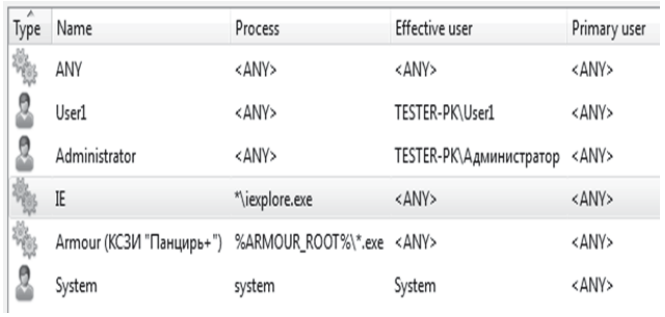


Fig. 2. Creating access subject graphical interface



Type	Name	Process	Effective user	Primary user
	ANY	<ANY>	<ANY>	<ANY>
	User1	<ANY>	TESTER-PK\User1	<ANY>
	Administrator	<ANY>	TESTER-PK\Администратор	<ANY>
	IE	*\iexplore.exe	<ANY>	<ANY>
	Armour (КСЗИ "Панцирь+")	%ARMOUR_ROOT%*.exe	<ANY>	<ANY>
	System	system	System	<ANY>

Fig. 3. Access subjects list graphical interface

While creating access subjects security system administrator can use masks and environment variables (which makes rules more flexible). The base example of created subjects to prevent intrusions is shown below (4). Access control policy in this case is based on process access control rules.

система	system
Local Service	<Любой>
Network Service	<Любой>
Администраторы	<Любой>
службы	<Любой>
Панцирь+	%ProgramFiles%\SPC ITB\Armour\bin*
Windows\System32	C:\Windows\System32*
Explorer.EXE	C:\Windows\explorer.exe
SearchIndexer.exe	C:\Windows\System32\SearchIndexer.exe
wbem\wmiprvse.exe	C:\Windows\system32\wbem\wmiprvse.exe
svchost.exe	C:\Windows\System32\svchost.exe
Local Service_M3	C:\Windows\System32\svchost.exe
Network Service_M3	C:\Windows\System32\svchost.exe
defrag.exe	C:\Windows\system32\defrag.exe
..\Program Files\Windows Sidebar\sidebar.exe	%ProgramFiles%\Windows Sidebar\sidebar.exe
RdrCEF	%ProgramFiles%\Adobe\Acrobat Reader DC\Reader\AcroCEF
Acrobat Reader	%ProgramFiles%\Adobe\Acrobat Reader DC\Reader\AcroRd3
AdobeARM	%ProgramFiles%\Common Files\Adobe\ARM*\AdobeARM.
OUTLOOK.EXE	%ProgramFiles%\Microsoft Office\Office14\OUTLOOK.EXE
Microsoft Office	%ProgramFiles%\Microsoft Office\Office14*
Internet Explorer	%ProgramFiles%\Internet Explorer\iexplore.exe
The Bat!	%ProgramFiles%\The Bat!\thebat.exe
Windows*	%SystemRoot%*
Program Files	%ProgramFiles%*.exe
Users	%SystemDrive%\Users*
ProgramData	%PROGRAMDATA%*
любой	<Любой>

Fig. 4 Access subject list example with using of masks and environment variables

The usage of primary and effective identifiers while creating access subject allows to implement user authentication when making object access requests. Also it allows to control and delimit impersonation (with other account credentials including system one) rights of process [1].

Access control rights are defined in following graphical interface examples (5, 6).

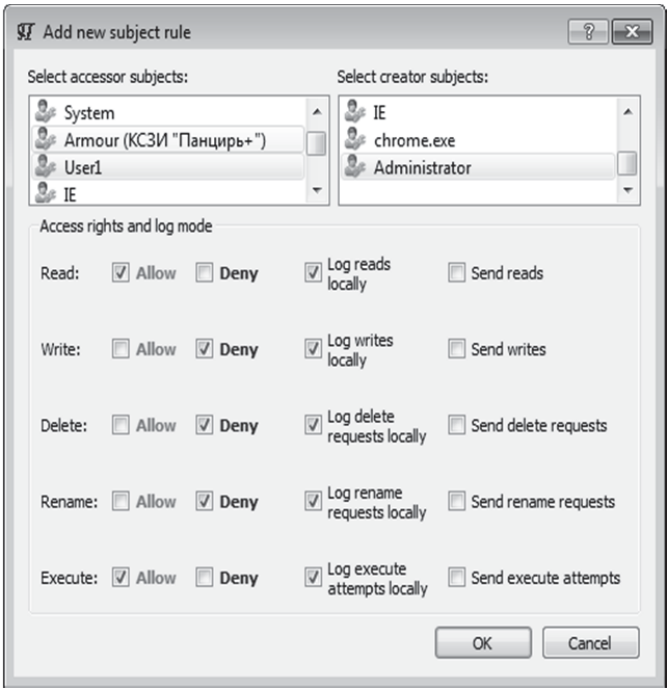


Fig. 5. Access control rights definition graphical interface

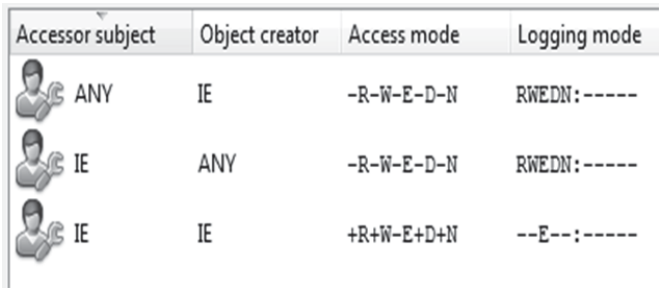


Fig. 6. Access control rights list graphical interface

The definition of access control rights is formed from a few actions. We choose the creator subject from list (5) so the access to objects created by chosen subject will be controlled and limited. After selecting creator subject we choose (5) the subjects which will be limited in accessing objects created by creator subject. For all of these subjects we define access rights to newly created objects.

Access control is implemented in the following approved way [3-5]. While controlled subject creates new file (or modifies existing file which has no marking) the file is automatically marked – file inherits creator credentials (this information could be stored in file alternate stream).

When another subject requests access to the file, access manager analyzes the file marking (inherited credentials). In case there is no marking file access is not controlled at all. If marking exists then access request is analyzed against access rules contradiction. As a result of analysis the request is denied or allowed.

Fig. 5 illustrates the ease of solving complex security tasks with just a few access rules if using this elaborated method. It illustrates rules which allow isolating Internet browser process. We allow browser to access only files created by itself, also denying to execute self created files. To other subjects we deny all access to files created by browser process. “Browser” subject is identified by process full pathname inside the access control policy.

We can list many practical tasks which can be solved using this method (with approved results received with developed technical solution). This method involves minimal security administrator input comparing with other methods. We could deny executing any newly created file by marking it – solving the malicious file execution problem. In other hand denying to execute files created by interactive user to System processes (System user account) – we could solve the system privileges escalation problem. Another example is denying other’s user data (i.e. information) access to Administrators and System user – solving unauthorized information access from privileged processes. Similar cases are multiply.

V. CLIPBOARD DATA ACCESS CONTROL IMPLEMENTATION

All temporary data pated to system clipboard also can be treated as “newly created” data and the described above access control method can be applied in similar manner. Access control rules can be similar to those shown in Fig. 5. While rules list representation can be like this one represented below. In this example first column represents the access subject requesting the data (from clipboard) while second column represents creator subject. Green highlighted strings mean allowing rules and red ones deny rules.

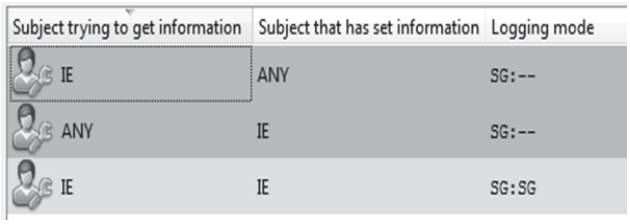


Fig. 7. Clipboard data access control rules

Clipboard access control method works in similar manner as file access control. When creator subject (including system user account and processes) writes data to clipboard access manager saves his credentials. When other subject requests access to clipboard data access manager verifies if this subject is represented inside the access control policy lists. If not so the access control is skipped. Else the request is analyzed against access rules contradiction. As a result of analysis the request is denied or allowed. Fig.7 illustrates example of

access control rules which isolate Internet browser process. These rules allow browser to read clipboard data only in case this data was written by the browser itself. All other access subjects also can't read data which was placed into clipboard by the browser.

VI. MANDATORY ACCESS CONTROL IMPLEMENTATION

Mandatory access control is mostly used to prevent unauthorized data access (stored in file objects). This allows saying that reviewed above principles of access control for newly created objects could be used as the base for mandatory access control implementation.

When implementing classical mandatory access control for static objects many realization problems do appear. This problems are related to implementation correctness and implemented security system administration complexity.

Mandatory access control method is based on security markers or "mandates" usage to specify access rights of subject to the object. The basic concept of mandatory method is the possibility of categorizing subjects and objects on any ground (like information access level, information privacy level, software trust level, object security level, etc.). practical implementations do mostly use mandatory access control basing on categorization of processed data.

The idea of suggested method usage inside the mandatory access control is based on understanding that if we can categorize access subjects and objects by some characteristic, then for every category we can define own security mark (digital mandate) and to define access control "default" rule basing on arithmetic comparing of mandates (numerical). Let's illustrate this approach implementation for mandatory control basing on Bell-LaPadula abstract model (which describes mandatory access control method [6]). Object security mark (mandate) describes the information privacy category which can be saved inside this object. Subject security mark (mandate) describe subject privileges (access level) to access different categories of information. We say that higher privileges access subject and higher object privacy are mapped to lower number in the linearly ordered sets of subjects (C) and objects (O) - C = {C1,…,Cl} and O = {O1,…,Ol} and lower mandate number Mi, i = 1,…,l is assigned to them, i.e. M1 < M2 < M3 < … < Ml. In common case the mandate is assigned to the group of equal (same access level) access subjects and objects of same privacy category. We use the following notations:

- Mc – subject (or group) mandate
- Mo – object (or group) mandate

Bell-LaPadula model implementation is used to prevent information privacy violations. It is provided by implementing of following formal mandatory access control rules:

- Subject C can read object O if Mc <= Mo
- Subject C can write object O if Mc = Mo

Access manager analyzes the request and basing on arithmetic mandate corporation allows or denies access. The basic contradiction of this method consists of necessary mandate assigning to all existing objects including system ones (which are not designed to be categorized by information processing privacy level). Plus correct mandatory access control method implementation is complex due to public access objects presence inside operating system (like temporary storage folders). If such temporary object gets security mark (mandate) then only user with the same mandate can write data to this object.

We can eliminate all these drawbacks while implementing mandatory access control using principles of access control for newly created objects (described above). In this case the security mark (mandate) must be assigned only to access subjects (users) and there is no need to assign mandates to objects.

Name	Domain	Mandate Level
система	NT AUTHORITY	
Гость	TESTER-PK	Open
Администратор	TESTER-PK	
User1	TESTER-PK	Secret
Tester	TESTER-PK	
NETWORK SERVICE	NT AUTHORITY	
LOCAL SERVICE	NT AUTHORITY	
Igor	TESTER-PK	Confidential
Administrator	TESTER-PK	Open

Fig. 8. Mandatory access control configuration based on access control for newly created objects

Access manager works using the following algorithm [3-5]. Mandatory levels (mandates) are assigned only for subjects (users) which must be controlled. On file creation (or modification) by the access subject this file is automatically marked – it inherits security mark (mandate) of the creator (mandate can be stored in alternate stream of the file). On file access this inherited mandate presence is analyzed. If the is no mark (mandate) then access to this object is not controlled. If mandate exists then the request is analyzed against access rules contradiction (formal mandatory access control rules). Then access is granted or denied.

This method makes administration easier and evident. Implementation correctness can be explain due to the fact that every created file without depending of place (this includes temporary files) will have the security mark (mandate).

The very important mandate method practical usage scenario is access control for newly created files implementation which allows secure information management with different confidential levels in the single informational system. In this case the access policy is based on defining

security marks (mandates) for both access subjects and access objects which are compared (greater, lower, equal) while access request analyzing.

We shall use following definitions:

- Mc is an access subject mandate
- Mo is an access object mandate

Then the mandate access control abstract model will look like this:

- Subject C can access object O in read mode if the following condition is true: Mc ≤ Mo
- Subject C can access object O in write mode if the following condition is true: Mc = Mo

The base mandate access control implementation problem is the need to define mandates for objects because not all objects could be classified by confidential levels.

If we will use the suggested method we need only to define mandates for access subjects (users). Meanwhile every created file receive mandate automatically by inheriting its creator (owner) mandate. I.e. object mandates are defined in auto mode by the security system itself when object (file) is created, eliminating the need of manual administration.

VII. ACCESS CONTROL FOR NEWLY CREATED OBJECTS ENHANCEMENT

The enhancing of access control for newly created files could be based on extension of automatic marking to the static files too (in addition to newly created files). It can be illustrated on security system which defends against malicious applications. The method particularity (access control for newly created objects method) in such illustrated case will be following. Any created on the computer file will be marked (by access manager) as “created” (same marking is applied for existing files on modification access). All executable files are marked as legal initially.

On every request to the file access manager analyzes file marking presence. If there is no marking – the access is allowed (the mark is automatically applied if modification requested). If mark is present the request is analyzed against access rules contradiction. If subject requests file execution which is marked (previously) then access is denied. If the subject requests modification or deletion or renaming of executable file access is also denied.

Thus in this case we do not only prevent malicious application execution (independently from infection method) but also deny modifying legal application. At the same time there is no need to configure the system – all marking will be done automatically during a short timeline (while system is in work). The only needed action will be access control system deactivation when installing new legal applications (but this case is not too often or even is rare in many systems).

VIII. CONCLUSION: ROADMAP FOR METHOD IMPLEMENTATIONS

We must to admit that suggested access control method and its implementation must not be used as alternative for standard access control mechanism (which targets static objects). They must be used together. In this case suggested method makes security configuration wider and easier. The standard approaches could be implemented with operating system security mechanisms (like File System ACLs - Access Control Lists in Windows and Linux operating systems), while the access control for newly created objects method needs additional tools (or additional security subsystem). This additional subsystem can base on the very same ACLs or use modern integrity levels and alternative file streams APIs (Application Program Interface) of Windows operating system (or some similar API on other operating systems). In most cases the implementation will need low level system FS driver to analyze file object requests and grant or deny access after all access subject and object verifications. Such security subsystem was implemented for the reviewed method and was practically approved in modern operating system environments.

We need to say that similar approaches could be applied to implement this access control method for other operating system and user owned objects (including critical for system security and operating system objects). The ease of administration when using this method could enhance the overall systems security cause one of the big problems in system security is the problem of complex administration. Complexity of system security administration is well known human factor based source of end system breaches. Also additional security mechanisms like mandatory access control described above also add complexity to security administration and introduce compatibility problems with operating system modules which were not design to be used in privacy categorized environment.

Suggested method can improve the situation helping system administrators keeping systems in secure state.

We plan also research on these approaches in future works.

REFERENCES

[1] A.U. Sheglov, K.A. Sheglov «Informational systems security analysis and design», St.Petersburg: Professionalnaya literatura, 2017.
[2] K.A. Shcheglov, A.U. Shcheglov, T.A. Markina «Formulation And Solution the Tasks of Protection the Information from Unauthorised Access» // Proceedings of the 18th Conference of Open Innovations Association FRUCT - 2016, pp. 599-605
[3] A.U. Sheglov, I.P. Pavlichenko, S.V. Kornetov, K.A. Sheglov Complex information security system “Armour+”, Software registration certificate #2014660889
[4] A.U. Sheglov, K.A. Sheglov «File access control system based on automatic marking», patent for innovation #2524566
[5] A.U. Sheglov, K.A. Sheglov «File access control system based on manual marking», patent for innovation #2543556
[6] A.U. Sheglov, K.A. Sheglov «Access control system for newly created encrypted files», patent for innovation #2533061K.G. Walter, W.F. Ogden, W.C. Rounds, F.T. Bradshaw,
[7] S.R. Ames, D.G. Shumway «Models for Secure Computer Systems». Tech. Rep. 1137, Case Western Reserve University, 1973.