Tightening Computer Security by Automating OS Attribute Configuration

Dimitri Podkorytov,
Computer Science Department
Kurgan State University, Russia,
podkorytov@mail.ru

Dennis Guster and Paul Safonov,
BCIS Department
St. Cloud State University,
St. Cloud, MN 56301
dcguster@stcloudstate.edu and safonov@stcloudstate.edu

Renat Sultanov,
Chemistry Department
University of Nevada, Las Vegas,
Las Vegas, NV
sultano2@unlv.nevada.edu

Abstract

This paper will examine applied security policies that are often employed both in government and business. A review of the literature has revealed that the common methods of assigning security attributes to system objects and granting user access rights does not provide the required level of information security in today's hostile environment.. Therefore, a method of creating a security subsystem within an operating system is suggested which is designed to significantly reduce the risk from human error in controlling access rights of system objects and administering the operating system. Specifically, the paper recommends dividing security policy into the following three subsystems: the policy for managing security labels, the policy for granting access rights in accordance with such security labels and the policy for verifying the rights of the system objects to exist in the system.

1 Introduction

The article provides an overview of applied security policies that are oriented towards the use not only in government information exchange, but also in business. The present article shows that traditional methods of assigning security attributes to system objects and granting user access rights does not provide the guaranteed level of information security, thus putting the information at risk. In the present article, the principle of creating a security subsystem of an operating system is suggested, which helps solve the abovementioned problems, as well as significantly reduces the risk from human error while controlling access rights of system objects. In addition, the article recommends dividing security policy into the following three subsystems:

1) The policy for managing security labels.
2) The policy for granting access rights in accordance with such security labels.
3) The policy for verifying the rights of the system objects to exist in the system.

2 Overview of Applied Security Policies

When defining access rights for a modern multi-user operating system, the principle of assigning security attributes to each object of the system has been widely used. Furthermore, the users of a system are assigned to specific groups in accordance with the nature of tasks they solve and the types of privileges they need. Such access groups make it considerably easier to administer the system .Each user or group (i.e. subject), as well as each resource (i.e. object), is assigned a specific security label—an identifier which allows controlling the access rights of subjects and system objects in accordance with the existing security policy. Currently, managing security labels implies manual assignment of such labels to every object of the system on the basis of the requirements set forth by the organization administering the system. The policy of managing security labels has not been given adequate attention in modern research, since managing security labels is a manual, not automated procedure, which makes it an organizational issue rather than a technical one.

Nevertheless, it is also possible to implement an automated or semi-automated strategy of assigning security labels based, for example, on the content of documents. Therefore, it is suggested that overall security policy should be divided into policies for managing security labels, granting access rights, and verifying the right of an object to exist in the system. Specifically, *the policy for managing security labels* is concerned with a set of rules according to which security labels are assigned to the objects and subjects of the system. Whereas, *the policy for granting access rights* involves a set of rules, according to which subjects are granted access rights to the system objects based on their corresponding security labels. Lastly, *the policy for verifying the right to exist* involves a set of rules to which the subjects and objects of the system shall comply in order to have the right to exist in the system. Thus, the expanded definition of the System Security Policy is as follows: it is a combination of the policies for managing security labels, granting access rights, and verifying the right of subjects and objects to exist in the system. At present, several security policies have been described in the literature. According to conventional terminology, these policies pertain, for the most part, to the policies for granting access rights. While only some of them are concerned with the issue of managing security labels in terms of granting subjects the rights to control security labels for subjects and objects within their authority.

2.1 The Harrison-Ruzzo-Ullman (HRU) Model [8]

This model is based on the administration of the rights of subjects to access objects and their subsequent control over the distribution of access rights. The components of the HRU model include objects *O* (system resources), subjects S (active subjects accessing the information), and a set of access rights *R*. The information base of a system is an access matrix *O*S*, described r[j,i], where the intersection specifies the rights subject *i* has on object *j*.

The policy provides granting subjects the right to control security labels for subjects and objects within their authority. Harrison, Ruzzo and Ullman proved, that in general, there is no algorithm that can determine if a given system in its initial state possess secure access rights. Since it does not take into account the security of transitions from one state to another, the discretionary access control model of Harrison-Ruzzo-Ullman neither guarantees security of the system, nor provides any method to control the information flow of Trojan horses [6].

2.2 The Typed Access Matrix Model (TAM & MTAM) [15]

This policy expands the discretionary model of Harrisona-Ruzzo-Ullman by introducing the concept of classifying conditions into the HRU model which provides more flexibility in decision making. The formal description of the TAM model includes:

1. a finite set of access rights $R=\{r_1,\ldots r_j\}$
2. a finite set of types $T=\{t_1,\ldots t_g\}$
3. a finite set of subjects and objects $S=\{s_1,\ldots t_n\}$, $O=\{o_1,\ldots o_m\}$
4. matrix M containing the access rights of subjects to objects and its initial state M_0

5. a finite set of operating commands over the access matrix C={$\alpha_1(\ldots),\ldots\alpha_k(\ldots)$}, including the conditions for command execution and their interpretation.

MTAM does not contain the non-monotonic operations such as *delete, destroy subject, destroy object* and therefore simplifies the process of proof of safety for the access matrix, but safety in MTAM's decidable case is, however, NP-hard problem**.** [15].

2.3 The MAC (Bell-LaPadula Mandatory Access Control Model) [1]

This security policy is concerned primarily with maintaining confidentiality. It is based on principles of maintaining security in classified government documentations. The principle underlying the MAC model involves a hierarchy of several security levels (top secret, confidential, office use only, etc). Objects are assigned security labels indicating the appropriate level of security. Subjects are also assigned security labels indicating their rights level. The two rules that ensure confidentiality in the MAC model are: a subject can read an object only if the security level of the subject is higher or equal to the security of the object .and a subject cannot transfer data to users with a lower security clearance.

This makes it impossible for a subject to write to an object if the security level of the object is lower than the security level of the subject. It prevents information flow from 'high' security levels to 'low' security levels. Unfortunately, it sometimes happens that personnel with high security clearance have to resolve problems on lower security levels. Also, there are situations when it is necessary to arrange a bilateral information exchange between two subjects with higher security clearance. Therefore, the MAC security policy is often used in combination with the discretionary access model. The combination of two models, however, may involve logical gaps and vulnerabilities.

2.4 The Chinese Wall (CW) Security Policy [4]

This security model was proposed by Brewer and Nash and is oriented towards commercial security. It takes into account possible conflicts of interest between rival companies using services of the same business consultants. For example, a market analyst can get information on the company's critical interests in a certain sphere Q_i. Upon obtaining such information from this company, the analyst cannot obtain information on similar business interests in the sphere Q.

2.5 The Clark-Wilson Security Model [3]

The main goal of this policy is to prevent unauthorized modification of information by authorized subjects. Clark and Wilson introduced the notion of two mechanisms for maintaining integrity: well-formed transactions (WFT) and separation of duties. The model is concerned with ensuring integrity of processed information/data. As its goals suggest, the policy involves the following: subjects can access objects only through authorized programs, duties are separated, one operation can be performed by several different people, one person can perform several operations and the events can be audited. Further, the model describes two security systems for military and commercial

purposes. According to this policy, each object and subject is assigned a hierarchical level of security. Also, like in the BLP model, objects of lower security levels are available to subjects in read-only mode. This helps maintain the integrity of the information.

2.6 The Biba Integrity Model [11]

The Biba's integrity model primarily addresses information integrity policies. It may be in fact viewed as the opposite of mandatory access control policies. The model classifies the data into integrity levels. According to this model, there are two key rules ensuring integrity: subjects cannot write objects with a higher integrity level (No write-up) and a subject cannot read an object with a lower integrity level (No read down). Within the framework of this policy all read-write operations are possible only within the limits of a single security level. To enable information transfer between objects at different levels, authorization at several levels of security is required. This policy is the easiest to implement, and probably therefore has served as a prototype for many security systems in modern operational systems.

2.7 McLean Model [13]

The McLean model is an expanded modification of the mandatory access control model BLP. The primary focus of the McLean model is not on the system level, but rather on secure transitions through which the secure state can be achieved. McLean's premise is that it is not possible to define a secure system solely in terms of the notion of a secure state because there is no global definition of security. According to McLean, the system needs to be both state-secure and transition secure. Furthermore, McLean introduces the notion of a secure transition function, the principle of tranquility, and the concept of a set of subjects authorized to change security levels. Specifically, the policy provides a model for a group of subjects to access common objects. When a subject requests any type of access to any object, the system downgrades all subjects and objects to the lowest level, and thereby adds the accesses to the access permission matrix, which then allows access.

2.8 Dion's Security Policy [7]

Dion's model is concerned with both integrity and confidentiality of information and it is often considered the most complete model. It basically combines the security models of Biba and Bell-LaPadul. In Dion's model, each subject is assigned three confidentiality labels and three integrity labels:

The Absolute Confidentiality Label (ACL) is assigned to a subject during its creation and remains constant for the whole life cycle of the subject.
Read Confidentiality Label (RCL) is a maximum level of confidentiality to which the subject has read authorization.
Write Confidentiality Label (WCL) is a minimum level of confidentiality to which the subject is authorized to write.

Absolute Integrity Label (AIL) is assigned to a subject during its creation and remains constant for the whole life cycle of the subject.
Read Integrity Label (RIL) is a minimum level of integrity to which the subject has read authorization.
Write Integrity Label (WIL) is a maximum level of integrity to which the subject is authorized to write.

The following set of rules must be held true for each subject *s*:

WCL(s) <= ACL(s) <= RCL(s)
RIL(s) <= AIL(s) <= WIL(s)

Dion's model enables information flow by means of establishing one-way information channels between objects:

ACL(o1)<=RCL(s)
RIL(s) <=AIL(o1)
WCL(s)<=ACL(o2)
AIL(o2)<=WIL(s)

2.9 Policy of Information Flow Analysis

This model describes the flow of information and is not concerned with describing the rights of subjects to access objects. Within the framework of this policy, the source of information, the receiver, and the rights of a subject are assigned. This idea has merit because it allows for a more precise (compared to a discretionary policy) definition of access rights to information flow. This policy could be considered a subset of the Dion model because it uses a similar principle for controlling and restricting information flow.

2.10 Secure Auction Service—Secure Bidding (SB) [12]

Provides security to commercial transactions and could be considered as analogous to an auction. The following are the principles employed in the model:

- Each auction has predetermined time frameworks.
- Each auction item cannot be rejected by a winning bidder, but can be taken off the auction by the seller.
- The winning bidder is the one with the maximum price declared.
- After the beginning of the auction the item price cannot be changed even by the auctioneer.
- Lost bids do not result in the loss of money by bidders.
- Bidders can simultaneously participate in several auctions.

2.11 Role-Compatibility Model (RC) [9]

Within the framework of this model, terms such as targets and inquiries are introduced.

Specifically, subjects inquire about access to targets (objects). The nature of inquires can vary. In order to be executed, the inquiries need to be in the access control list (ACL). Some examples of the targeted objects can be: **FILE** – file, **DIR** – directory, **DEV** – device, **IPC** - Object IPC: the semaphores, divided memory, etc. and **USER** – users.

Moreover, each type of the targeted object can have a subtype. For example, the file can be Normal, Confidential, or System. The Role concept defines a certain class of subjects, setting the privileges to the members of this class according to certain target subtypes and other classes such as: **Name** – name of a role, **Role Comp** – compatible roles (the roles to which the given role can be converted without change of the owner), **Admin Roles** – the roles which the given role can administer. Such a variety of role settings (actually about 20) provides for great flexibility in a security system and results in a high degree of flexibility. For example, it is possible to adjust role settings so that system administrators can add users, assign passwords, delete user accounts but not edit */etc/passwd* or */etc/shadow* manually.

3 Security Threats and Policy Implementation

Each of the security policies described in this article is subject to the following threats:

- Hidden channels. Methods of data transfer that circumvent the security policy mechanism and authorization. There are two predominant types: hidden channels that exploit time and hidden channels that exploit data.
- Back door entries built-in by developers.
- Time outs to a system response (reaction).
- Buffer overflows.
- Equipment and software failures or disruptions.

In addition, the necessity to manually set the attributes of security objects can cause the following problems:

1. The task of administering access rights tends to be subjective and routine. Decisions on assigning attributes are made by humans. Humans, sometimes make mistakes. Therefore, human factors can interfere with system security. Moreover, there could also be a time factor because administrators often do not have time to correct a problem. In some cases it may be beyond their control because they may simply be informed about the problem too late. Also there may be a knowledge factor because administrators do not know all the details and specifications for a given system. Details such as the information types processed by users or existing human relationships among a workgroups collaborating on a specific tasks are often not obvious to administrators.
2. Due to growing volume of the information processed, an increased number of users, and the complexity of the problems users solve, the administrators' task has become increasingly more complex.
3. The principle of division of access rights into user groups (workgroups) and the concept of inheriting privileges to access child elements of the file system are widely used. These result in complete positional dependence which does not ensure the

necessary level of file security. For example: nothing prevents a "Top Secret" file from being placed into the public (general) directory. The security policy described below is capable of preventing some of the threats discussed above.

3.1 Policy of object existence

According to the Computer Press, the major reasons for the destruction of information include [16]: 52 % - Unpremeditated actions of personnel, 15 % - Fires, 10 % - Deliberate personnel actions, 10 % - Equipment failure, 10 % - Flood and 3 % - Other. Ultimately, the cumulative share of information destruction due to actions of people accounts for 62 % of the destruction, this figure was further broken down by [16] as follows: 81 %- current personnel, 13 %- external people and 6 % - former employees. In the aggregate, this statistical data demonstrates the importance of considering the human factors and implementing measures to decrease their impact.

The policies discussed above focus on granting a subject access rights to an object while failing to detect suspicious behavior of that subject when granted access rights to the object. Moreover, these policies prevent the system from eliminating the threats related to that subject's suspicious behavior. Due to this design strategy, the majority of modern security policies allow for "denial of service" (DoS) hacker attacks. According to the classification developed in the present article, a policy is suggested related to the concept of object existence. It is based on the following principles:

1. Assigning a safe corridor. Under the notion of a safe corridor, we will be able to define the state of the equipment and software. For example, if, in a Windows operating system, it is not possible to increase a swap file, such condition should be classified as unsafe.

2. Unsafe state should be: a) identifiable (detectable); b) impermissible; c) recorded in a log file/protocol; and d) the system should be equipped with a strategy to terminate an unsafe state.

3. There should be relational integrity within the system's description.

4. There should be periodic control over the subject's rights to exist in a system.

5. The system should be process-oriented, and not user-oriented. In a system, the subject is not a user, but a user process.

6. The level of detail of the system's description should be maximized and include analysis of its operating states.

7. Principle of multiple secure/safe corridors.

Let us consider the principles listed above more closely.

3.2 "Safe Corridor" Principle

Depending on the complexity of implementation and the type of resources required, several levels of corridor analysis are possible.

1. Whether the process meets its own min/max corridor parameters.
2. Considering change rates for process parameters over a period of time.
3. Considering change rates for parameters of all system processes over a period of time.
4. Analysis of min/max parameters of all other processes in a system by a certain process.
5. Consideration of a processes own behavior history.
6. Considering the history of the behavior of the whole system.

3.3 Relational Integrity of System Description

Every state of a system should be identifiable. Unfortunately, this principle is difficult to implement, since it requires the following: a detailed description of activity for each process and knowledge of the complete behavior history for each process in a system. Due to the fact that during its activity a process changes the memory conditions, enormous memory sizes would be required for registering the complete history of process behavior. It is possible that with the introduction of reversible computing systems (i.e., computing systems capable of returning to any of their previous states) the solution described above would be feasible. At present, this is a theoretical issue in the area of calculation theory[10].

3.4 Periodic Control over Subject's Right to Exist in a System

Traditionally, one of the major principles of computer system security is granting subjects access rights to its objects. This principle works well for identifying the subject and for making a decision about whether or not this subject shall be granted access rights to an object. However, this method does not take into account dynamics of subject behavior and does not provide for performing a quantitative estimation of the adequacy of the subject's behavior during the use of that resource. For example, a subject that has been assigned a right to pass through the door may misuse this right and create difficulties for other system processes by accessing the door (port) with a frequency of 10 times per second. In this case, the resource would be hardly accessible for other subjects. The majority of modern DoS attacks are possible just because security policies do not take into account the dynamics of system behavior. Therefore, it is necessary to control the behavior of the subject continuously and to verify whether its behavior is in conformity with the security policy.

3.5 Process-Oriented System

In a system, the subject is not the user, but the process of the user. Processes are more important and more informative in the scheme of the system security. A user is only one of the attributes of a process, whereas each process has numerous attributes. The activity

of a user is easily detectable through the activity of the processes associated with that user.

3.6 Maximum Level System Detail and Analysis of Its Operating States

A maximum level of detail is necessary to ensure that the information about each state of the system is complete. All process attributes should be registered. The system should be able to identify every combination of its attributes. Ideally, it would be preferred to maintain the history of all transitions. Unfortunately, such an approach demands memory resources that are not available at present, as well as some untraditional calculation methods. At present, implementing the principles of the suggested approach is beyond the existing technological capabilities. For this reason, registering behavior dynamics may be narrowed down to registering the most important attributes of a process.

3.7 Principle of Multiple Secure Corridors

The suggested policy implies color-coding of the corridors based on the degree of trust for the subject (process) or object (resource). For example incoming e-mail files, Internet-cache files, and third-party files should be associated with the red corridor. Temporary and core files represent a great threat to system security as they are usually located in directories with minimum access rights. We suggest introducing a file attribute called ***temporary***. If an attribute is marked as temporary, it would indicate that the file in question can exist in the system, but only for the duration of the existence of the process which created this file.

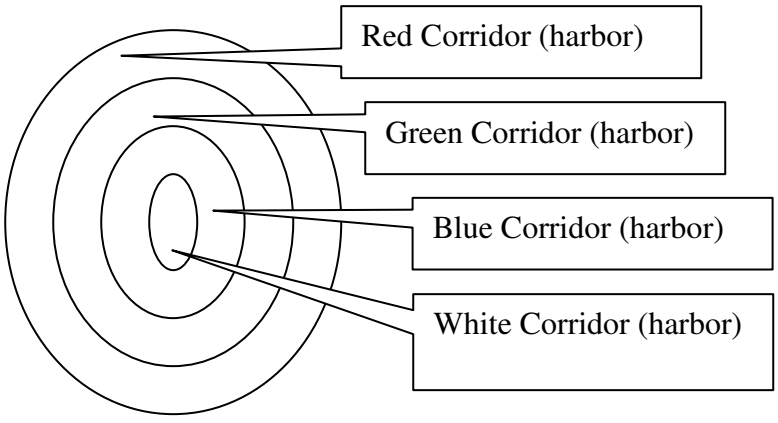

Figure 1: Security Corridors (Harbors)

3.8 Passive Principle of Content Control

As mentioned above, taking into account the human factor is just as important for system security as controlling unauthorized access or viruses. The concept of passive and active control of information flow contents can serve as a tool to reduce the influence of the human factor. In a multitasking operation system, each user works with a number of information flows and has certain access privileges to them. The user works in a certain information space. As the system has multiple users who work independently (exchange data, solve common problems), these information spaces frequently intersect. Such intersection areas are potentially dangerous. Thus, within the framework of a discretionary model, the security level of a file can be lowered by any user who has read access to it. To do so, it is enough for the user to have write access to a directory with a lower level of security than the users' own level. At the intersection of these policies, however, certain logical gaps are possible. The method described in the present article suggests analyzing the information flow content (memory, file, network dataflow) and assigning appropriate security labels based on the content analysis. For example, a combination of words "Submarine," "Top Secret," "Author: Ivanov," "Department: Research and Development," "Project: Alpha" in a file can unequivocally indicate that the access of users and user groups need to be restricted [14].

The benefit of this approach is the absence of dependency of this file on its location in a file system. The file can be located anywhere in a system; its access mode will be position-independent. In this case, file contents and not file attributes will be subject to control. The drawback of this method is that the user can alter the data security level by incorporating corresponding keywords into this level, i.e., by sending a "Trojan horse" to a higher security level. To prevent the user from altering the data security level, a secret code-identifier may be included into the body of a document (for example, "Class: XDFWE34A").

Any user with read access can lower the file security level by removing or replacing (coding or encrypting) keywords and by recording a message to a file at a lower security level. Thus, the mandatory principle here is not applicable in its classical implication. This vulnerability can be avoided if:

1. Security levels do not intersect.
2. The active principle of content control is enforced. The principal suggests utilizing multiple-key content encryption described in [2].
3. The flow control is implemented according to the principles introduced in [5], which generally prevent regular users from lowering or raising the security level of an information flow and allows only users with corresponding rights to perform these operations.
4. The automatic labeling principle is combined with the mandatory security policy. In this case, the problem of unauthorized altering of identification labels of a document can be solved, because automatic label assignment will transfer the document to a restricted access class. According to the BLP policy, write access in this class is not allowed.

The functions of object transition from one state to another will also be secure, and thus the criticism of the Clark-Wilson model of discretionary access control will be addressed.

Integrating the automatic label assignment into a mandatory security policy will make a system less dependent on the human factors and consequently more secure.

3.9 Active Principle of Content Control

The principle of controlling the content of information flow can be further developed by introducing a dynamic change of file contents based on the content of the file and on the attributes of a subject who has access rights to this file. This concept will be further referred to as the active principle of content control. By the term "active control," we shall apply the rules of lexical transformation of the information depending on the context of user process flow (i.e., a dynamically calculated security label).

It is worth mentioning that traditional anti-virus monitoring is a subset of a system using active control of content. However, traditional anti-virus monitoring does not take into account the contents of a process. For example, an anti-virus program can attempt to modify a read-only file. The active principle of content control can be applied in the following cases:

1. To improve anti-virus protection by enabling it to repair the files dynamically in contrast to being able only to identify dangerous types of viruses.
2. To enable contextual processing of the data flow when a function of transferring the input into the output depends on the attributes of the process providing access to the file. Thus, it is possible to apply the object-oriented principle of accessing system resources.
3. An example of contextual dataflow processing occurs in contextual dynamic encryption, in which case an encrypted key is calculated as a function of the attributes of the process providing access to the file. In this situation, it is possible to create an encryption mode in which the key also depends on the carrier's identifier. In that event, a different key is necessary for reading the information copied on a portable device (such as a CD or a floppy).
4. Another possible solution consists in forming one single data flow as a set of different messages from different correspondents by means of applying the algorithm described in [2]. That encryption principle will reduce the volume of transmitted data and complicate the encryption analysis because the information will have as many interpretations as there are correspondents.

In the case of unauthorized access, destruction becomes possible. In addition, it is possible to implement context-dependent encryption. Although it may be argued that anti-virus control is similar to such a system, the suggested approach is more complete, as it enables control not only over the executed code, but also over the data, the user, and the user privileges.

4 Advantages of the Suggested Approach

1. Positional independence of the user data access rights is achieved. The problem of transferring information confidentiality labels is eliminated.

2. Risk associated with the human factor is decreased. According to the suggested approach, the person controlling the technical conditions of an object (i.e., a system or network administrator) can be prevented from interfering with the security labels of the information flows in a system. This property is unique and unfeasible in systems with manual label assignment.
3. Easy duplication of access rights. This feature is particularly important for large corporations because it is easier to create a set of rules for an organization of multinational scale and then to replicate them rather than have a system administrator assign access rights to files on thousands of computers in that organization.
4. Actual lexical and possibly semantic control of file contents instead of just file attributes can be used.
5. Anti-virus control is viewed as one of the system tasks.
6. Encryption of access privileges decreases the level of a system administrator's knowledge of the particular tasks solved by the users.
7. The privileges of the Security Administrator and the System Administrator can be separated into two non-overlapping groups.
8. Security labels can be encapsulated in the existing protocols (including IP) and can enhance security level administration of the network traffic.
9. Context-dependent flow encryption becomes possible.
10. Behavior of this type of security system becomes highly predictable, since all the system components will have identical settings. Thus, such a system becomes predictable at a macro level.

5 Disadvantages of the Suggested Approach

1. More system resources will be required if adequate performance is to be maintained.
2. It is necessary to document all information flows and access privileges to them.
3. It is necessary to control not only contents, but also information flows.
4. To decrease vulnerability of the information within the tasks performed it is necessary to encrypt access rights during transmission and storage.
5. In a network, it is necessary to replicate access rights to information on the entire path of the document.
6. Object labels are not constant, which contradicts the requirements of the IEEE standard.
7. Unfortunately, the "macro" predictability of the system is a double-edged sword: if the behavior of the system is predictable for the personnel of the company, it also becomes predictable to hackers. It is another reason why such systems should not be applied "as is," without other security measures.

6 Summary and Conclusions

The article suggests dividing security policy into three policies: label administration, access administration, subject existence. More specifically, it describes the policy for managing security labels based on the analysis of file contents, memory, and network traffic. Depending on the contents, users and user groups are automatically assigned corresponding access rights. The Dion model is suggested for access administration

policy, since it provides a high level of information detail. The article sets forth the policy of subject existence based on a periodic assessment of the subject behavior and the conformity of this behavior to a set of suggested corridors.

The advantages of the policy for managing security labels and the policy of subject existence are outlined. We conclude that the computer system security can be significantly improved if these policies are implemented in combination with the Dion model for granting access rights. Due to its positive features, the suggested method can be recommended for application in distributed calculation systems, since it eliminates the problem of security label transfer. In addition, this model is suitable for creating authorized information channels. This paper examined applied security policies that are often employed both in government and business. A review of the literature has revealed that the common methods of assigning security attributes to system objects and granting user access rights does not provide the required level of information security in today's hostile environment, thus putting the information at risk.

Therefore, in this paper, the principle of creating a security subsystem within an operating system was suggested, which was designed to help solve the above mentioned problem, as well as significantly reduce the risk from human error in: controlling access rights of system objects and administering the operating system. To attack this problem, the paper recommends dividing security policy into the following three subsystems: managing security labels, granting access rights in accordance with such security labels, and verifying the right of the system objects to exist in the system. Specifically, the security policies described in this paper were designed to deal with the following threats: hidden channels, back doors built-in by developers, measurement of time of a system response (reaction), buffers overflow, and equipment and software failures or disruptions.

References

1. Bell, D.E. and L.J. La Padula. Secure Computer System. Bedford, MA, 1974.

2. Erosh, I and S. L. Erosh. "Arithmetic Codes with Correlation of Multiple Errors". Problems of Information Transfer. 3(4). pp. 56–63, 1967.

3. Clark, D.D. and D.R. Wilson. "A Comparison of Commercial and Military Computer Security Policies". Proceedings of the 1987 IEEE Symposium on Security and Privacy, pp. 184-194, May 1987.

4. Brewer, D.F., and M.J. Nash "The Chinese Wall Security Policy". IEEE Symposium on Security and Privacy, 1989.

5. Podkorytov, D. Working paper on Flow Control, Kurgan State University, 2002.

6. Zegzhda, D. and A. Ivashko. "Basics of Information System Security". Moscow:Hot Line - Telekom, 2000. page 425.

7. Dion, L.C. “A Complete Protection Model”. Proceedings of the 1981 IEEE Symposium on Security and Privacy, pp. 49-55, April 1981.

8. Harrison, M., Ruzzo, M. and J. Ulman “Protection in Operating Systems.”. Communication of the ACM, 1976.

9. “Methods and Means of Ensuring Information Security”. Evaluation Criteria for Security of Information Technologies. Part 1. Introduction and General Model. Moscow: IPK Publisher of Standrads, 2002.

10. Galatenko, V. A. Information Security Basics. http://www.osp.ru/dbms/1996/01/49.htm /

11. http://www.trustedbsd.org

12. Franklin, M. and M. Reiter. “The Design and Implementation of a Secure Auction Service”. Proceedings of the IEEE Symposium on Security and Privacy. Pages 2-14, May 1995.

13. McLean, J. “Security Models”. Encyclopedia of Software Engineering, 1994.

14. Tsaregorodtsev, A. V. Information Security in Distributed Managing Systems: Monograph.. Moscow: RUDN, 2003.

15. Sandu, R. S. “The Typed Access Matrix Model”. Proceedings of IEEE Symposium on Security and Privacy, Oakland, California, May 4-6, 1992, pages 122-136.

16. http://redilem.sjf.stuba.sk/files/tm/final/M4%20Communication%20and%20labour%20psychology.pdf