Mandatory Access Control Policies Based on Vague Requirements

Juraj Masar, Jiri Bartos, Cyril Klimes
University of Ostrava

Abstract

This article focuses on mandatory access control and security policies within an operating system. It proposes a general methodology for the selection and deployment of policies based on vague descriptions, which are representing operational, functional and security requirements imposed on the operating system. The methodology is extended by generating customized policies and supported by an expert system, as a tool, that offers to minimize, or even completely eliminate, the need for a security consultant as an expert in the problem domain, for designing and selecting policies to harden the operating system and furthermore to design a secure operating system. This methodology moves this key responsibility to either the user or the system administrator.

In this article, the methodology is used to generate components of operating system and with conjunction with SELinux reference policy to generate the customized policies, also called custom policies, which will allow setting the security level of certain applications vaguely.

1. Introduction

An operating system is a program which, among other things, manages the hardware resources of a computer system and creates an environment for application software [1]. Since computer systems are very often used for interpreting and processing of the application data of the critical nature, there is a necessity that the operating system is secure. A proposal for secure operating systems and their implementation is currently the subject of intense scientific research, as secure operating system must include the design and well as implementation of various security mechanisms in such a way as to guarantee the enforcement of security policy, despite the security threats which it faces [2]. For these operating systems it is characteristic that they implement security mechanisms that are conducting a mandatory access control.

A mandatory access control represents a way of managing access, where the access of an entity to the object is controlled on the grounds of security policies, or more precisely security models, such as Bell-LaPadula model of confidentiality, or Biba model of integrity [3]. For clarification, the term “security model” also refers to a security policy, “entities” are defined as users, processes or threads, and the term “objects” represents files, folders, network ports and other system resources. Entities are assigned to a domain and an object type. A security policy then, using its rules, determines operations that can be performed in a domain by an entity with an object of a specific type [2].

From a practical point of view, security policies can be changed centrally by trusted administrators. Users, on the other side, are not allowed to make such changes as well as to change the security settings of objects (e.g. files). Any access of an entity (e.g. process) to a particular file is checked and it is being enforced that this access is in accordance with the authorization rules of the security policy. This action is conducted by the concrete implementation of a mandatory access control – mostly at the kernel level of an operating system (assuming that the operating system consists of a kernel and a set of supporting libraries and applications that communicate with the kernel through system calls). The most common implementations that ensure the mandatory access control are SELinux or AppArmor and others [2]. The number of existing operating systems, however, allows the administrator to use a pre-set security policy. Its rules are often broken up into so-called “modules” or multiple security policies that can be easily modified or created as new ones. The deployment of such policies is usually done by compiling of these policies and linking them within the kernel of the operating system.

2. Selection and deployment of policies

The selection and deployment of security policies for the mandatory access control is a time-consuming and also knowledge-intensive process, which requires expert knowledge of a security consultant or an administrator with an expertise in a computer security. Both are the experts in the problem domain who are analyzing operational, functional and security requirements during the design and implementation of a secure operating system before its deployment. These requirements are often of a vague nature and therefore cannot be interpreted exactly. Having expert knowledge during the implementation of a secure operating system is, therefore, crucial.

In case that this expert is not available, the operating system or its applications may not work correctly, or the operating system will not even start.

In the worst case there may be a security incident that can lead to violations of confidentiality, integrity and availability of the data and applications.

3. The methodology and expert system

A solution to the problems above is the proposal of an expert system. This system will advise the selection and application of the most relevant and appropriate security policies that are in accordance with specified requirements for the operation, functionality and also the security of the operating system.

By automating of this process, the need for the presence of human security experts is minimized or even completely replaced, which will reduce the time and also the financial costs of developing a secure operating system. The responsibility will be transferred to such an architect who has inadequate knowledge in this field. The responsibility can even be transferred to the user or administrator of the operating system who can describe the requirements by using his vague language.

Note: In the proposed methodology the architect, user and administrator of the operating system will all be considered as users of the expert system and will be hereinafter referred to only as a "user".

3.1. Proposed methodology

The proposed general methodology deals with the selection and deployment of security policies for a mandatory access control and it is designed for multiple operating systems. It consists of six steps, and it is described in the following figure (Figure 1.) and it is an extension of the methodology intended for designing of a secure operating system as described in [4].

3.1.1. Questionnaire. In the first step, the expert in the problem domain formulates the questions and answers that will be the part of the questionnaire. This questionnaire will be the basis on which the user interacts with the expert system. The questions that require answers from the user are used to design the operating system and are represented by fuzzy linguistic variables. These are the requirements for the operation, functionality and also the security of the operation system.

Answers to each question are represented by fuzzy linguistic values. The obvious advantage of such approach is that the user without greater knowledge of the operating system design can understand the questions and can answer them. As questions and answers in this questionnaire are of a general nature, it can be applied to multiple operating systems.

3.1.2. Specification of the requirements. The user of the expert system specifies requirements for the proposed operating system. The questions prepared by experts and the answers are constructed in the native language and this construction is conducted by selecting the appropriate language values that are vague.

3.1.3. Finding the security policies. In this step, the expert system is conducting the search for individual security policies that meet the specified requirements of the completed questionnaire on its knowledge base.

3.1.4. Evaluation of the appropriateness of the security policies. Based on the properties of each requirement, the expert system evaluates the relevance of security policies which were found in the previous step. Their evaluation represents the adequacy of their deployment.

3.1.5. Visualization and selection of the security policies. In this step the visualization of the evaluated security policies is made. The user can then choose a solution that is recommended by the expert system.

3.1.6. Deployment. The last step is aggregating the security policies that have been selected and that are being deployed by implementation into the kernel of the operating system.

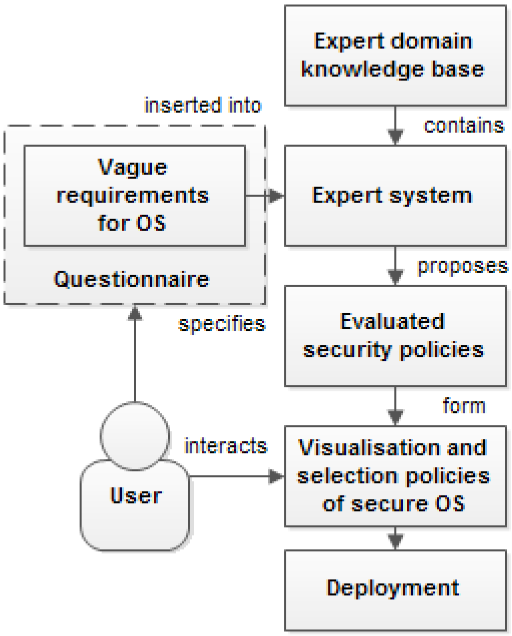

Expert domain knowledge base
inserted into
contains
Vague requirements for OS
Expert system
Questionnaire
proposes
specifies
Evaluated security policies
form
interacts
Visualisation and selection policies of secure OS
User
Deployment

Figure 1. Overview of the proposed methodology

3.2. Reasoning model

To support this methodology, the expert system (Figure 2) using a general model of decision making under uncertainty, which is described in detail in [5], [6] and used in [7], [8], [9], is proposed and consists of the following four processes.

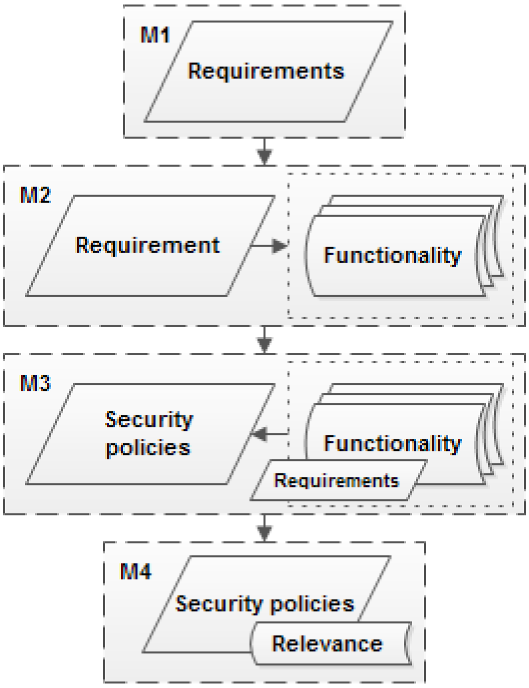


Figure 2. Processes of the proposed expert system

3.2.1. Process M1. Performs completion of the input data of the modeled reality and makes a selection of information that is relevant to the resulting solution. Vague requirements on the operating system that the user has specified in the questionnaire and thus extend the functionality of the operating system model, together with their associated security policies, are entering this process.

An example of such vague requirements:

- Amount of data – many
- Number of users – small

Also, the requirements that do not have a vague character are present in this process. For example:

- Platform – Linux,
- Platform – BSD
- Platform – MS Windows

3.2.2 Process M2. The process creates the sets of admissible solutions based on the rules stored in the knowledge base and important information that is needed to be dealt with. The input information is then enriched with the result of the inference process and the inadmissible data is discarded. In the context of the overall problem, this process creates the admissible set of functionality within the operating system according to the input requirements and reduces the unsatisfactory functionality.

An example of an admissible set for the "Data protection" requirement is the functionality that is supporting local file systems (ext2, ext3, ntfs, fat32).

When a particular platform is specified, for example platform "Linux", then there is a reduction of functionality (ntfs, fat32) which is therefore not considered in the decision-making process.

3.2.3. Process M3. This process models the effects of available solutions. Based on the set of admissible solutions – the sets of functionality which have been selected from the previous step, process M3, in the context of other input requirements, is evaluating the appropriateness of the functionality and hence security policies as they relate to such functionality.

An example: IF "Data protection – high" AND "Platform – Linux" THEN "ext2, ext3", WHERE evaluation of “ext3” > evaluation of “ext2”.

3.2.4. Process M4. This step makes a selection of the most suitable solution. In this step occurs the most suitable set of security policies, which should be deployed in the operating system and this set is selected based on the rules that are stored in the knowledge base. For example, this process can select policies with only high and medium evaluation.

4. Results

For the purpose of the practical verification of this general methodology, a working prototype that is described in detail in this section was developed and implemented.

4.1. Questionnaire

The user is communicating with the expert system using the questionnaire that is prepared by the expert and has the following structure (the user’s responses are summarized in Table 1.).

4.1.1 Requirements. Requirements of functionality of the operating system that will be covered with the security policies:

- Data protection – includes synchronization and backup software, local and distributed file systems, tools for managing disk arrays, tools for monitoring the physical status of hard drives, etc.
- Network security – includes software for firewall management, intrusion detection systems, software for monitoring network traffic and network interfaces, analysis tools, VPN services support, etc.
- Remote access – functionality for remote access to the operating system. This includes

support for various terminals and protocols, as well as client and server mechanisms.
- Communication service – includes software for transfer and delivery electronic messages, software for scanning and filtering messages, chat software, instant messaging, telephony and VoIP software, etc.
- File sharing – this functionality includes distributed file systems, application protocols for file transferring based on client-server and peer to peer architecture, etc.

Note: The functionality (supported by a set of policies) is linked to the individual requirements. The set of functionalities is of course, not a definitive one. This set is, along with the security policies, stored in a knowledgebase of the expert system and can be easily expanded.

4.1.2 Platform. The selection of a specific platform, for which the security policy is designated:
- Linux
- BSD

4.1.3 Properties. The properties determine character of the requirements and can be defined in a vague or an exact form.

They are described by fuzzy values like “a few”, “medium”, “many”. In some cases, the properties are for better linguistic interpretation described by fuzzy values like “low”, “medium” and “high” or even by “small”, “medium” and “big”.

Data protection:
- Number of users – the number of the local and the network users that create or access to the data.
- Amount of data – the amount of the data that needs to be protected.
- Network storage devices – the number of the storage devices on the computer network where the data is stored for protection.
- Backup complexity – the overall complexity of solution (mechanism) for data backup in the sense of its implementation, time or financial resources.

Network security:
- Number of clients – the number of the users that are active in the computer network.
- Amount of traffic – the amount of the traffic that is transferred over the computer network in the form of the data per one day.

Remote access:
- Number of users – the number of the users that will use remote access service.
- Number of sessions – the number of the terminal sessions opened by one or multiple users.
- Security – the overall security level of remote access solution (mechanism).

Communication service:
- Number of users – the number of the users that will use the communication service.
- Message recipients – the number of the users that will be recipients of one message.
- Collaboration – the cooperation of multiple users to achieve the common goal.
- Usability – the ease of use and also the learnability of the client communication software by the user.
- Security – the overall security level of the communication service.

File sharing:
- Number of users – the number of the users that share the data.
- Data per user – the amount of data that is shared by individual users.
- File description – the level of details that is used for the description of one file.
- Number of sessions – the number of the file downloads and the file uploads together.
- Network storage devices – the number of the storage devices on the computer network where the data of the individual users is stored.
- File systems support – the support of various local file systems on the same storage device.
- Security – the overall security level of the file sharing service.

Table 1. Questionnaire with answers regarding the requirements

Requirement	Platform	Properties	
Data protection	Linux	Number of users	small
		Amount of data	big
		Network storage devices	few
		Backup complexity	high
Network security	Linux	Number of clients	medium
		Amount of traffic	medium
		Importance	important
Remote access	Linux	Number of users	small
		Number of sessions	small
		Security	medium
Communication service	Linux	Number of users	small
		Message recipients	few
		Collaboration	high
		Usability	high
		Security	high
File sharing service	Linux	Number of users	small
		Data per user	medium
		File description	low
		Number of sessions	medium
		Network storage devices	few
		File systems support	low
		Security	high

The “importance” property is used for the detailed selection of the security policies for the corresponding functionality. The higher level of the “importance”, the more policies will be selected for the corresponding functionality. These fuzzy values are as follows:

- Very significant
- Significant
- Important
- Less important
- Unimportant

4.2. Knowledge base

The knowledge base of the expert system consists of a two sets of IF-THEN rules, the purpose of which is to evaluate the relevance of the security policies and the relevance of the individual settings that are used for generating custom policies.

For the preparation and testing of the IF-THEN rules LFLC 2000 software (Linguistic Fuzzy Logic Controller) was used. It is a specialized tool based on the theory of fuzzy sets and fuzzy logic, which allows the derivation of the conclusions based on a vague description of a situation with linguistically formulated fuzzy IF-THEN rules [10].

4.2.1 Policies selection. The first set of IF-THEN rules is used for the selection of appropriate policies.

An example of the IF-THEN rules that are used for the "Data protection" requirement:

```
IF (requirement is "Data Protection"
AND number_of_users is small
AND amount_of_data is big
AND network_storage_devices is few) THEN
functionality "Backup" is big

IF (requirement is "Data Protection"
AND number_of_users is small
AND amount_of_data is big
AND network_storage_devices is few) THEN
functionality "Synchronization" is medium

IF (requirement is "Data Protection"
AND number_of_users is small
AND amount_of_data is big
AND network_storage_devices is few) THEN
functionality "Replication" is small

IF (functionality "Backup" is big
AND platform is "Linux"
AND backup_complexity is high) THEN
policy "bacula" is very big

IF (functionality "Backup" is big
AND platform is "Linux"
AND backup_complexity is low) THEN
policy "backup_script" is very big
```

For example, the IF-THEN rules that correspond with the "Communication service" requirement are as follows:

```
IF (requirement is "Communication service"
AND number_of_users is small
AND message_recipients is few
AND collaboration is high) THEN
functionality "Instant_messaging" is big

IF (requirement is "Communication service"
AND number_of_users is small
AND message_recipients is few
AND collaboration is high) THEN
functionality "Email" is small

IF (functionality "Instant_messaging" is big
AND platform is "Linux"
AND usability is high
AND security is high) THEN
policy "jabber" is very big

IF (functionality "Instant_messaging" is big
AND platform is "Linux"
AND usability is high
AND security is high) THEN
policy "ircd" is very small
```

4.2.2 Customized policies. The second set of the IF-THEN rules is used for generating custom policies. The custom policies are derived from the templates that were created as a part of the operating system and generated from individual settings that are specific for each policy. These settings are called in the SELinux environment as "Boolean settings" or as "Policy Booleans" only.

For example the IF-THEN rules that are used for evaluation of the Boolean settings for customizing policy of the "ftp" service are as follows.

```
IF (policy is "ftp"
AND number_of_users is small
AND data_per_user is medium
AND security is high) THEN
boolean "ftp_home_dir" is big

IF (policy is "ftp"
AND number_of_users is big
AND number_of_sessions is big) THEN
boolean "ftpd_use_passive_mode" is very big

IF (policy is "ftp"
AND number_of_users is big
AND data_per_user is many
AND number_of_session is big
AND security is high) THEN
boolean "ftpd_anon_write" is very small
```

4.2.3 Fuzzy processing. The LFLC 2000 software tool is conducting the fuzzy processing (Figure 3.) of provided inputs based on the fuzzy IF-THEN rules.

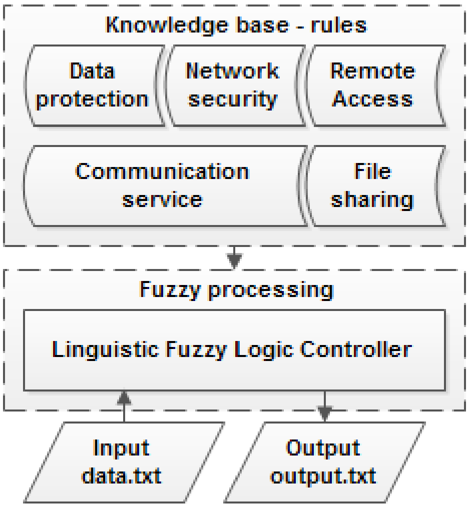


Figure 3. Fuzzy processing scheme of LFLC tool that is used for the evaluation

The results are stored in the output file:

- data.txt – the input file representing the data from the user, the content is shown in the Table 2.
- output.txt – the output file that contains the evaluated security policies and the Boolean settings for customizing security policy.
- Rules – the set of IF-THEN rules.
-

Table 2. Content of data.txt file (questionnaire)

Requirement	Fuzzy variable	Value and linguistic interpretation	
Data protection	dp_number_of_users	25	small
	dp_amount_of_data	1500 (GB)	big
	dp_net_storage_devices	1	few
	dp_backup_complexity	1	high
Network security	ns_number_of_clients	90	medium
	ns_amount_of_traffic	800 (GB)	medium
	ns_importance	4	important
Remote access	ra_number_of_users	40	small
	ra_number_of_sessions	90	small
	ra_security	2	medium
Communication service	cs_number_of_users	30	small
	cs_message_recipients	5	few
	cs_collaboration	1	high
	cs_usability	1	high
	cs_security	1	high
File sharing service	fs_number_of_users	20	small
	fs_data_per_user	50 (GB)	medium
	fs_file_description	3	low
	fs_number_of_sessions	70	medium
	fs_net_storage_devices	1	few
	fs_file_systems_support	3	low
	fs_security	1	high

Note: the context of these entry requirements for the operating system (data protection, network security, remote access communication service and file sharing service) is not directly related to each other and needs to be viewed separately. The aim was mainly to demonstrate the suitability assessment of the security policies.

The evaluation process of the input data is a straightforward defuzzyfication process that is realized by the defuzzyfication of fuzzy sets, one of them is shown in the Figure 4.

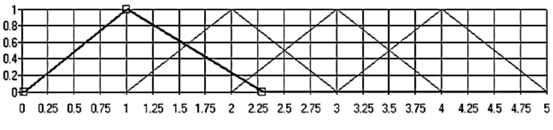


Figure 4. Form of fuzzy set corresponding to the evaluation and defuzzyfication of 4 input attributes

4.3. Visualization and discussion

The role of the visualization is to interpret the appropriateness of the security policies in a graphical manner and present it to the user.

The visualization has two layers. The first layer contains a list of policies that are categorized on the basis of the requirements from the questionnaire and also contains a chart of the suitability for each policy. The second layer shows individual Boolean settings that are recommended for activation when customizing security policies from the first layer. It is linked to each security policy.

The darkest colors represent the best and the most relevant security policies, which should be included in the proposed operating system and most relevant Boolean settings which should be activated.

Conversely light colors represent the policies and Boolean settings which are not too significant or needed.

4.3.1 Data protection. The most appropriate security policies for the requirement of the data protection are shown in Figure 5.

As the most appropriate policies tools working with disk arrays “raid” and backup software “bacula” (because of the number of local and network users that create or access to data is “small”, the amount of data that need protection is “big” and on the computer network are “a few” storage devices where the data is stored and protected) were selected. Also the complexity of the backup solution is “high”.

The security policy for the distributed file system “glusterd” is not needed and the policy for the high performance journaling file system, the “xfs”, was evaluated as medium.

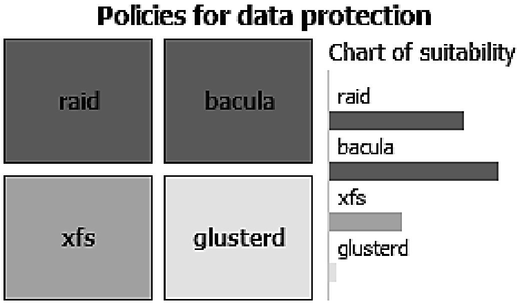


Figure 5. Visualization of data protection policies

4.3.2 Network security. The selection in the area of the network security policies is shown in Figure 6.

Since the importance of the network security is of a lower degree, set to the “important”, the result is that the “iptables” as a firewall software is the most appropriate policy and the others are not needed to saturate this requirement.

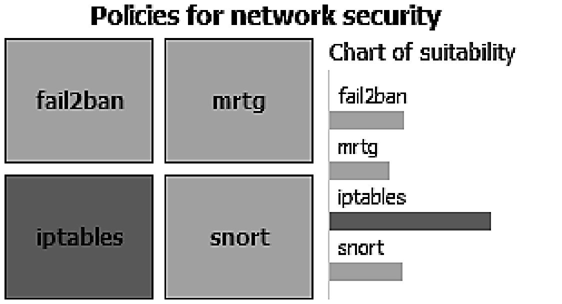


Figure 6. Visualization of network security policies

4.3.3 Remote access. The selection of the security policies for the requirement of the remote access is shown in Figure 7.

The policies for the "ssh" client and "sshd" server were selected as the most appropriate ones because the number of users and sessions is "small" with the "medium" level of security. The security policy for the "telnet" client has small appropriateness since the telnet protocol is not too secure without additional security mechanisms like cryptography. The policy for restricted shell "rssh" was evaluated as medium.

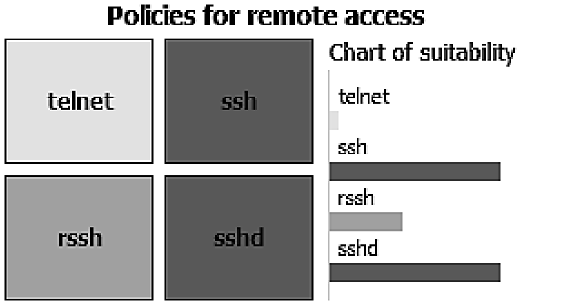


Figure 7. Visualization of remote access policies

4.3.4 Communication service. The security policies that are recommended for the requirement of the communication service are illustrated in Figure 8.

As the most appropriate policy was selected the policy for the "jabber" service, which is used for instant messaging. This selection was mainly based on the "small" number of the users who are sending messages to the "few" recipients and on the three properties, the user's collaboration, the usability of the client software and the security of the service, which was set to the "high" level.

The security policies that were evaluated as not too significant, do not meet the entry requirements (the input properties) from the questionnaire.

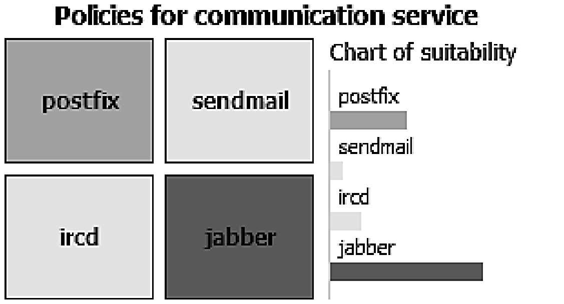


Figure 8. Visualization of communication service policies

4.3.5 File sharing service. The evaluated security policies that are related to the requirement of the file sharing service are shown in Figure 9.

As the most recommended security policy for this requirement was selected the policy that supports the file transfer protocol, "ftp" service. It is mainly because the number of the users is "small", the amount of the data shared by the individual users is "medium", the description of the files is set as "low", the number of the sessions is up to "medium" and on the computer network there are "few" storage devices.

The policy for the "samba" networking system has small appropriateness. It requires lower security level because this networking system is not too suitable in big wide area networks with too many unknown connections (this is included in security). It requires higher support of the file systems and more network storage devices with not too many sessions per device.

The "apache" web service and its security policy was evaluated roughly as "medium", since it differs from the "ftp" service by the property as is the file description and the policy that supports distributed file system "afs" is evaluated as small because it requires "big" number of the users, "medium" or "big" amount of the data and more network storage devices.

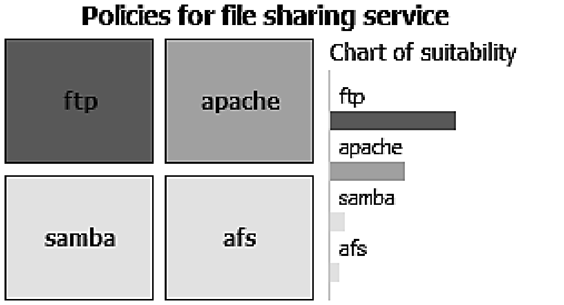


Figure 9. Visualization of file sharing service policies

4.3.6 Customizing the "ftp" policy. The evaluated Boolean settings that are used for customizing the "ftp" policy are shown in Figure 10.

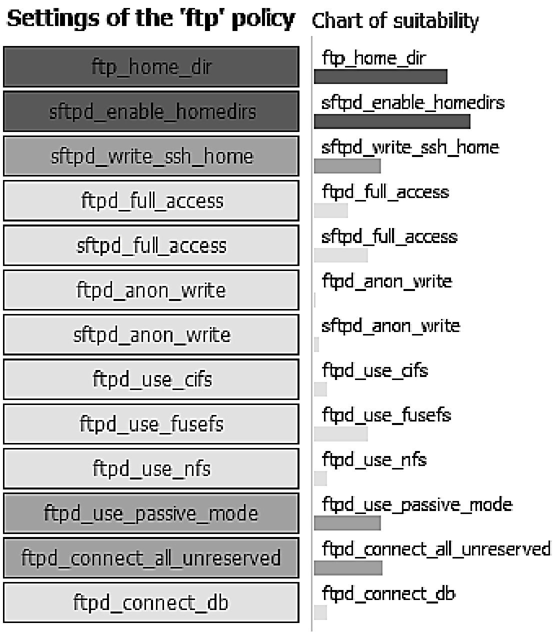


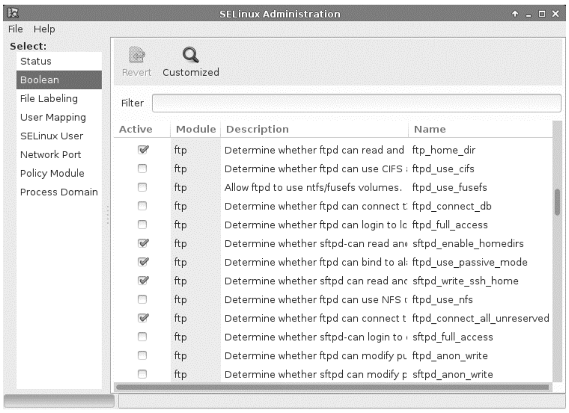
Figure 10. Visualization of the evaluated Boolean settings of the "ftp" policy

The most suitable Boolean settings, which are recommended for activation when customizing this policy, are settings that allow read and write files in home directories of users. This decision was based on the "high" level of the file sharing security, on which are preferred the segregated directories for the specified "medium" amount of the data that is shared by the "small" number of the users. For this reason were the Boolean settings for the full access that is one of the possibilities of access to the data evaluated as small. The full access is used to login of local users and for reading and writing all files on the system that is governed by discretionary access control.

The settings that are related to the modifying files by anonymous users are not recommended by the expert system, they were evaluated as very small.

The Boolean settings for the control of the network connections were recommended as medium, since the "medium" number of sessions and the "small" number of users was specified. Thanks to this, the usage of the database is not recommended and it was evaluated as small. The database is often used for authenticating users and for the accounting purposes.

Note: for creating a new policy modules and templates was used system-config-selinux tool. It is shown in Figure 11.



Active	Module	Description	Name
☑	ftp	Determine whether ftpd can read and	ftp_home_dir
□	ftp	Determine whether ftpd can use CIFS :	ftpd_use_cifs
□	ftp	Allow ftpd to use ntfs/fusefs volumes.	ftpd_use_fusefs
□	ftp	Determine whether ftpd can connect t:	ftpd_connect_db
□	ftp	Determine whether ftpd can login to lc	ftpd_full_access
☑	ftp	Determine whether sftpd-can read an	sftpd_enable_homedirs
☑	ftp	Determine whether ftpd can bind to al:	ftpd_use_passive_mode
☑	ftp	Determine whether sftpd can read anc	sftpd_write_ssh_home
□	ftp	Determine whether ftpd can use NFS (	ftpd_use_nfs
☑	ftp	Determine whether ftpd can connect t	ftpd_connect_all_unreserved
□	ftp	Determine whether sftpd-can login to (	sftpd_full_access
□	ftp	Determine whether ftpd can modify pu	ftpd_anon_write
□	ftp	Determine whether sftpd can modify p	sftpd_anon_write

Figure 11. System-config-selinux tool

5. Conclusion

The proposed general methodology is applicable to multiple operating systems and was tested on existing policies which are part of SELinux reference policy.

The selection of the security policies can be easily extended by adding additional IF-THEN rules to the knowledge base of the proposed expert system. It can be also tuned by adding more entry requirements that are represented by the properties in questionnaire.

Within the selection of the security policies it is also possible to customize the individual policies that allow setting the level of the security of certain applications vaguely.

6. References

[1] T. Jaeger, *Operating System Security*, Morgan and Claypool Publishers, California, 2008.

[2] C. Pfleeger, *Security in Computing*, Prentice Hall, Massachusetts, 2006.

[3] B. Walek, J. Masar, "Methodology for design of safe operating systems", *Global Journal on Technology*, Academic World Education & Research Center, Turkey, 2013, pp. 634 – 638.

[4] C. Klimes, "Expert system utilization for modeling the decision making processes upon indetermination", *Acta Electrotechnica et Informatica*, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovak Republic, 2008, pp. 40 – 46.

[5] C. Klimes, "Model of adaptation under infeterminacy", *Kybernetika*, Institute of Information Theory and Automation, 2011, pp. 356 – 369.

[6] J. Bartoš, B. Walek, P. Smolka, J. Procházka, and C. Klimeš, "Fuzzy modeling tools for information system testing", *17th International Conference on Soft Computing Mendel 2011*, Brno, 2011, pp. 154 - 161.

[7] B. Walek, C. Klimeš, “Fuzzy tool for conceptual modeling under uncertainty”, *Proceedings of the SPIE, Volume 8349*, USA, 2011.

[8] B. Walek, J. Bartoš, and C. Klimeš, “Process-oriented component modeling under uncertainty”, *AWERProcedia Information Technology and Computer Science*, Turkey, 2012.

[9] H. Habiballa , V. Novák, A. Dvořák, and V. Pavliska, “Using software package LFLC 2000”, 2nd International Conference Aplimat, Bratislava , 2003, pp. 355-358.

[10] C. PeBenito, F. Mayer, and K. MacMillan, “Reference policy for security enhanced linux”, *SELinux Symposium*, 2006.