Automation security in a machine-building plant

K Yu Zhigalov¹, E R Guzueva² and S I Polovneva³

¹V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences
²Chechen State University, 17a, Dudaev Boulevard, Groznyy, 364015, Russia
³Irkutsk National Research Technical University, 83, Lermontov str., Irkutsk, 664074, Russian Federation

E-mail: kshakalov@mail.ru

Abstract. The operating system is the most important software component of any computer, and therefore, the security of the enterprise information system as a whole depends on the level of implementation of the security policy in a single operating system. In general, Linux operating systems (OS) cannot be considered as a ready-to-use security system. There are many recommendations on improving Linux security, but the authors of these recommendations do not guarantee that their recommendations are sufficient to achieve the required level of system security. This study attempts to systematize available methods for increasing the level of security of application software for Linux-like OSs operating as part of control systems at nuclear power facilities. The required level of security of the entire system can be achieved at the application software level.

1. Introduction

Due to increased security threats and the modification of viruses, Linux cannot be considered as a ready-to-use security system.

The main and crucial problem in all security systems for control systems is unauthorized access (UA). UA threats and attempts are preventable and may be suppressed (thus not creating the possibility of their implementation), as well as their successful implementation may be prevented [1, 2].

This requires the following:

• clearly identify the codes of applicable operations in the system and all entities/objects that have the right to perform them. The operating system software shall monitor the possible UA attempts, notify their launch and block the operations used to launch them in time. For example, in case of repeated application of the wrong password at logon, the activity of the user under whose name the login is attempted is blocked;

• reduce the codes of applicable operations when working with the system to the minimum level. For example, using a single utility for data backup. In other words, specific actions in the system should be performed by the only possible way, and not several;

• use of sets of interoperability rights that entitle only authorized transactions;

• completely exclude the use of private rights to obtain new ones belonging to other users. To do this, it is necessary to develop a private security policy (SP) as an addition to the Linux OS SP to comply with the above requirements;
• all network infrastructure, PCs and operating systems will be considered the security policy objects. Everything else, personnel, software and equipment that directly interact with the objects are subjects of the SP.

2. Methods and materials
The modeling of concepts taking into account the possible harm of developed SP is often classified as the “violator”. A violator is a subject with the right of access to objects or when trying to obtain it in an accidental or deliberate manner, in the future to use these rights for private purposes. In this case, to access privileges and handle data. The violator by its actions may not harm by making changes that interfere with the correct functioning of the system or change the settings of the system, but nevertheless this ability to carry out such activities remains, thereby posing a potential threat. It may be contemplated herein that any subject may be a potential violator. It should also be noted here that any subject that does not plan any malicious activities may be classified as the “violator”. For example, a user may mistakenly be assigned the superuser rights, which he or she can accidentally use [3].

The human factor is one of the most important and pressing security issues. Here, even an experienced user from the support services that can use an identifier and a key with system administrator rights may be considered a “violator”. In this case, the attributes of a user with higher access privileges are usually used. Weakly protected systems usually apply a security policy, where a user with low privileges uses their access rights to expand them. In this case, the “violator” may not deliberately compromise the system by leaving the software active after the end of the session, thereby allowing further access to passwords or IDs [4, 5]. The above actions are classified as a dangerous type of attack due to the fact that it is difficult to track. Besides, the threat of unauthorized access may be performed not only by the subject, but also by objects, and may manifest itself with an incorrectly configured operating system or application programs with incorrect algorithms. In this case, the threat comes from assigning erroneous rights to the default user, who should not have such rights according to the SP.

3. Results
The problem of emergency should be considered separately at the preliminary stages of SP development and be independent of the developed SP. In the future, these tasks will be closely interrelated and will have to be solved as a whole. In fact, the development of a safe UA system can become the main problem of eliminating or preventing emergency or critical situations, in particular if the actions are carried out in the manual control mode. Therefore, on the one hand, in case of emergency, the SP shall switch to the reduced safety mode so as not to interfere with the operational actions of the maintenance personnel. However, on the other hand, in an emergency state, the system will be in the most vulnerable state, so an attacker can create or falsify such a situation and take advantage of the moment, which may further lead to disastrous consequences.

Therefore, Remote Access or Remote Administration should only be implemented if absolutely necessary, because they are a major vulnerability to external network attacks.

A simple rule to improve security – the smaller the number of users in the system, the lower the risk of dangerous situations associated with improper assignment of rights. Based on this, it is necessary to take actions to reduce the number of subject, thereby reducing the risks for violator activity in relation to objects (network administrators, technical support staff with access rights to the system). Here, it is important to exclude the possibility of exposure of subjects/objects on behalf of other subjects/objects. The communication of objects via information channels is desirable with authentication of received and transmitted information. It is mandatory to number packets with their further encryption when transmitted to network objects. The receiving subjects shall decrypt packets and analyze counters. In many cases, it can be considered a failure if the values are not equal to the actual ones when transmitting a packet (calculated by the receiving party) or a network segment, a new device may appear, which may mean an intrusion attempt. Modern network devices (switches,
gateways) allow automatically identifying unauthorized packets or packets with unknown network addresses.

The most reliable but difficult solution is the development of personal software, which has all the necessary tools and a single operator environment. Here, it is desirable to have a console with the right to access only favorite users. The developed software shall have all the system administration tools.

It is necessary to highlight the recommended stages in the development of such software:

• identifying elementary operations and compilation of their full list, execution of which will be carried out at the request of the personnel (operators);
• identifying and developing a complete list of activities to be performed when working with the system. Work refers to the sequential execution of interrelated operations. If any operation that is part of the required work cannot be performed by an operator, then this operator cannot perform the work;
• developing a complete list of responsibilities of maintenance personnel;
• grouping of work by individual responsibilities;
• assigning responsibilities to individual users (individuals).

The hierarchy is built during all stages, an example of which is shown in Fig. 1, where uniquely mapped operations, work, responsibilities and users themselves are given.

After building such a hierarchy, at the application software level it is possible to control – allow/prohibit the execution of all operations and works by a single operator. Ideally, the correctly designed system itself provides the operator with the means for which he has the rights.

![Figure 1. Distribution of rights and responsibilities among users](image)

The proposed approach to the problem is unique since the hierarchy tree is developed once at the stage of software design and development, and is not subsequently modified while using the system. The main thing is that the SP does not require continuous support and the system administrator is not present in the hierarchy at all, thus being limited to personnel who change users’ passwords as a password administrator. Such a person will not be able to harm the system, as in the case of using standard Linux SP, in which the system administrator can perform any actions (copying, deleting, editing or formatting).

In any case, the most vulnerable chain in such a system is the password administrator, since no one restricts it to assigning a password to any user and login to the system using his “login”. There are many tools for preventing such actions, such as biometric data identification systems, for example, fingerprint scanners. The user ID shall correspond to two confirmations – password and fingerprint.
Login actions are carried out in stages: entering a “login”, entering a password, and as a confirmation – a fingerprint. Such a system should store fingerprints of all users registered and having access to the object in its memory.

The main difference in this approach is the inability to act on behalf of someone else. Such applications can guarantee the security of the system. If such software cannot be developed, then the user is allowed to use only the software necessary for his work. This approach is quite common and all its disadvantages are listed above.

Restriction and access control to objects. All attempts to access objects must be stopped at the stage of processing network packets. Only packets with known MAC addresses must be sent over Ethernet networks. The appearance of an “alien” Ethernet packet in the network is a connection to the network of another device and the security system must respond accordingly with a corresponding signal. For full tracking of non-native packets, specialized software can be used, which function is to control all traffic on a certain segment of the network.

To prevent non-native packets from spreading to other segments of the network, the functionality of switches is also used, the configuration of which allows them setting up their internal inter-port forwarding and routing tables with sufficiently secure routing tables.

Prerequisites for creating a unified operator environment. Building a fairly easy-to-operate and UA-protected multi-user information system is quite difficult. This is mainly caused by the fact that in the initial stages of software design it is quite difficult to answer the following questions:

• which users the software is calculated for and what is the range of end users;
• how fully the system should support multi-user mode;
• what risks and where the UA threat may come from.

Linux functionality is designed in such a way that most developers and users who rely on this OS in their projects can solve any of their tasks based on standard SP. In other words, the operating system is universal, but there are many packages that provide a wide range of tasks and require specific conditions for their operation. The OS concept is based on a time-tested security policy.

However, the OS requires users of a rather professional level to work with standard tools included in the distribution, which is also not guaranteed in many cases.

The versatility of the system also poses a threat, with possible incorrect configuration of the system, since with a variety of configurations it is very difficult to take into account many factors. However, Linux followers argue that the simpler the system, and the smaller the use of programs developed by third-party manufacturers, the less likely it is to occur that failures and situations that arose due to software conflicts in the software are not taken into account at all stages of development.

The Linux security policy requires increased attention and support from the system administrator. Like any user, an administrator may make a mistake even having many rights in the system. Therefore, the presence of such a person with system rights is not desirable, but on the other hand it is necessary only when using Linux OS SP.

The problem of having an administrator is supplemented by the already expressed opinion about the presence of professional users at the facility.

The following recommendations can be considered to solve these problems:

1. Persist number of employees, and the employees rarely change. In this situation, there is no need to create new accounts, change access rights and privileges, which excludes the use of standard tools for configuring Linux OS SP and the use of qualified personnel.

2. At the facility, the responsibilities of all staff are strictly limited and cannot change over time, and each employee is assigned a unique set of software tools, which he can use only. In the case of a change of employee, the unique set of rights remains unchanged.

3. For the SP system, it is quite critical to support multi-user mode, i.e. only one person can work with the system at a time. Remote work with the system cannot be allowed.

4. Each technical support employee should use only software created by the system developer, where there is no need for standard Linux tools.
Linux functionality allows combining everything listed in the Unified Operator Environment (UOE). By using UOE, you can simplify and automate all possible activities. In this case, all users will not be able to go beyond the scope of the UOE and do not allow running standard administration utilities. The above features are not available to users.

UOE serves as some kind of intermediary between utilities and the user allowing only a certain user operating and to the extent authorized by the internal security policy of UOE. Each user of the system works only in a specially designed shell and uses only the means that this shell provides. No other tools are available to the user. The shell itself is presented in the form of windows with access to various tools and a strictly defined set of actions is performed working with a certain window. For all users, working in the system is presented as a session. Connecting to the system, the user opens a session, when he leaves the system, closes it. Another user can log in only if the previous user has closed the session, essentially only one user can work with the system. Only one window is available before the session is opened when the user performs the authentication. After confirming his identity, the shell is transferred to the appropriate mode. Self-tuning for the type of work performed by the connecting operator is meant by the mode.

The following are options for shell self-tuning:

• each user has a strictly defined list of rights that specifies the windows he can work with. At the same time, the windows that the user is not allowed to work with are not displayed;

• each user has a specific shell, which is previously created as a separate program and loaded immediately after the session is opened, or in a single program, a separate part of it is responsible for working with a specific user.

4. Conclusion

The study suggests the following recommendations on the design of the user interface:

1. Try to use as few windows as possible.
2. Group control elements in the window by their functionality.
3. The interface and its design shall prevent erroneous operator actions.
4. Correspond to a single interface element style.
5. It is recommended to avoid “highlighting”.
6. Reduce the color palette as much as possible and use a strict stylistic character.
7. All application programs must be designed with a strictly defined screen resolution in mind.
8. Activation of any control must always result in unambiguous visual changes expected by the operator.
9. The status of all displayed control elements must correspond to the current state of the control elements.
10. Ensure confirmation to the user that the system has accepted his request and that the request has been completed correctly.
11. All fault or error notifications must be displayed in a short form (1-2 lines of text), and it must be possible to obtain detailed information on the error.
12. Avoid complex and intricate reference systems as much as possible.
13. Other windows should not be available until one window is closed.
14. Avoid the ubiquitous use of control elements with “Apply” response.
15. Remove the ability to drag windows. Upon activation it is desirable that each window should appear strictly at a specific location and have a predetermined, constant size.
16. The availability of the “Freeze” button is everywhere, the function of which is that after clicking it any input will be blocked until the password of the user who clicked the button is entered.
17. Log availability where all operator actions should be displayed and recorded.

References

[1] Klyuev R V, Bosikov I I, Mayer A V, Gavrina O A 2020 Sustainable Development of Mountain Territories V.12 2(44) 283-290
[2] Klyuev R, Madaeva M, Umarova M 2020 International Ural Conference on Electrical Power Engineering (UralCon) INSPEC Accession Number: 20034041 356-361

[3] Bovkun A, Korodyuk I 2019 Analysis of the development of small innovative enterprises in the construction industry. IOP Conf. Ser.: Mater. Sci. Eng. 667

[4] Skorobogatova Y A, Bovkun A S, Ivanov M Yu, Shilova O S 2020 The role and place of small innovative enterprises in the construction industry in the modern economy of single industry towns. IOP Conf. Ser.: Mater. Sci. Eng. 880

[5] Repinskiy O D, Konyukhov V Yu, Bovkun A S, Schupletsov A F 2021 Improving the competitiveness of Russian industry in the production of measuring and analytical equipment. Journal of Physics: Conference Series 1728