SECURE AND RESILIENT
COMPUTING FOR INDUSTRY AND HUMAN DOMAINS

Volume 2
SECURE AND RESILIENT
SYSTEMS, NETWORKS AND INFRASTRUCTURES
Edited by Vyacheslav Kharchenko

Secure and Resilient Industrial Control Systems

Computer Systems and Networks Security and Resilience

Security and Resilience of Web- and Cloud-Systems

Risk Analysis of Systems of Systems Security and Resilience

Human-Machine Engineering for Security Critical and Resilient Systems

Security Management Systems

БЪЛГАРСКА АКАДЕМИЯ на НАУКИТЕ 1869

CITY UNIVERSITY OF LONDON EST 1894

TUT

KTH

cini

UNIVERSITY OF BANKING OF THE NATIONAL BANK OF UKRAINE

LIMESYSTEMS

radiy

Volume 2. SECURE AND RESILIENT SYSTEMS, NETWORKS AND INFRASTRUCTURES

Tempus 2013 SEREIN 2016

MULTI-LECTURE BOOK

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS

VOLUME 2
SECURE AND RESILIENT SYSTEMS NETWORKS AND INFRASTRUCTURES

2017

Co-funded by the Tempus Programme of the European Union

Co-funded by the
Tempus Programme
of the European Union

Ministry of Education and Science of Ukraine
National Aerospace University n. a. N. E. Zhukovsky
"Kharkiv Aviation Institute"

V. Sklyar, V. Kharchenko, E. Babeshko, A. Kovalenko, O.Illiashenko, O. Rusin, A. Panarin, S. Razgonov, D. Ostapec, I. Zhukovyts'kyy, S. Stirenko, O. Tarasyuk, A. Gorbenko, A. Romanovsky, O. Biloborodov, I. Skarha-Bandurova, E. Brezhniev, A. Stadnik, A. Orekhov, T. Lutskiv, V. Mokhor, O. Bakalynskyi, A. Zhylin, V. Tsurkan, M. Q. Al-sudani, Yu. Ponochovnyi

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS.

Secure and resilient systems, networks and infrastructures

Multi-book, Volume 2

V. S. Kharchenko eds.

Tempus project
SEREIN 543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR
Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains

2017

V. Sklyar, V. Kharchenko, E. Babeshko, A. Kovalenko, O.Illiashenko, O. Rusin, A. Panarin, S. Razgonov, D. Ostapec, I. Zhukovyts'kyy, S. Stirenko, O. Tarasyuk, A. Gorbenko, A. Romanovsky, O. Biloborodov, I. Skarha-Bandurova, E. Brezhniev, A. Stadnik, A. Orekhov, T. Lutskiv, V. Mokhor, O. Bakalynskyi, A. Zhylin, V. Tsurkan, M. Q. Al-sudani, Yu. Ponochovnyi. **Secure and resilient computing for industry and human domains. Volume 2. Secure and resilient systems, networks and infrastructures** / Edited by Kharchenko V. S. – Department of Education and Science of Ukraine, National Aerospace University named after N. E. Zhukovsky "KhAI", 2017.

Reviewers:
Dr. Peter Popov, Centre for Software Reliability, School of Informatics, City Universi-ty of London
Prof. Stefano Russo, Consorzio Interuniversitario Nazionale per l'Informatica (Na-ples, Italy)
Prof. Todor Tagarev, Centre for Security and Defence Management, Institute of Information and Communication Technologies of the Bulgarian Academy of Sciences;
Prof. Jüri Vain, School of Information Technologies, Department of Software Tallinn University of Technology

The second volume of the three volume book called "Secure and resilient computing for industry and human domains" contains materials of the lecture parts of the study modules for MSc and PhD level of education as well as lecture part of in-service training modules developed in the framework of the SEREIN project "Modernization of Postgraduate Studies on Security Resilience for Human and Industry Related Domains"[1] (543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR) funded under the Tempus programme are given. The book material covers fundamentals issue of secure and resilient computing, in particular, description of related standards, methods of cryptography, software security assurance and post-quantum computing methods review.

The descriptions of trainings, which are intended for studying with technologies and means of assessing security guarantees, are given in accordance with international standards and requirements. Courses syllabuses and description of practicums are placed in the correspondent notes on practicums and in-service training modules.

Designed for engineers who are currently or tend to design, develop and implement information security systems, for verification teams and professionals in the field of quality assessment and assurance of cyber security of IT systems, for masters and PhD students from universities that study in the areas of information security, computer science, computer and software engineering, as well as for lecturers of the corresponding courses.

The materials in the book are given in a form "as is", desktop publishing of this book is available in hard copy only.

[1] *This project has been funded with support from the European Commission. This publication (communication) reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.*

36 ASSESSMENT OF SMART BUILDING AUTOMATION SYSTEMS REALIABILITY AND CYBER SECURITY USING ATTACK AND FAULT TREES

As noted in Chapter 35, in several cases maintenance of Building automation system (BAS) architecture components stops at the operation phase. However, due to circumstances, it is impossible to refuse application of such components or they might have low cost. Moreover, when developing specifications for information and control systems of smart buildings to assess the reliability and cyber-security the selection of the non-failure operating probability criterion (NOP) of the system can be justified.

In this Chapter, we discuss the application of the Attack Tree Analysis (ATA) technology to assess the impact of each component of the system architecture on its reliability and cyber security. Using ATA does not take into account recovery and maintenance, but it allows monitoring any attacks on components and assessing the impact of these attacks on the system as a whole. In the second part of the Chapter, strategies of developing Markov models for describing the recovery of system components after an attack or a software failure are discussed. The use of ATA or Markov models is usually justified by the customer's requirements for a specific criterion for assessing the quality of the system.

36.1 A conceptual approach to assessing reliability and cyber-security of smart building information and control systems

In this Chapter, with respect to the BAS, the main requirement of the user (client) is to ensure a given system availability, the second requirement is to ensure the cyber security of the system and information throughout the life cycle.

For the three-level BAS architecture considered in the thesis, the system-wide availability is influenced by the components of all its levels. The failure of the communication level component directly affects the availability of the system, since the impossibility of transferring the administration commands isolates the lower-level actuators. In addition, the communication level is most accessible for attacks on its components, which reflects its contribution to system-wide cyber security. Components of other levels (management, automation) also affect the availability of BAS; attacks on them can be identified through monitoring and analysis of system performance. Given the distribution of these levels, it is assumed that single failures of their components do not lead to system shutdown in general.

36.1.1 Basic principles

The architectures of information and control systems of smart buildings can be structurally different from each other, depending on the area where they will be applied (hospitals, departmental buildings, etc.). Fig. 36.1 shows the tree of high-level architecture attacks built using the ATA approach.

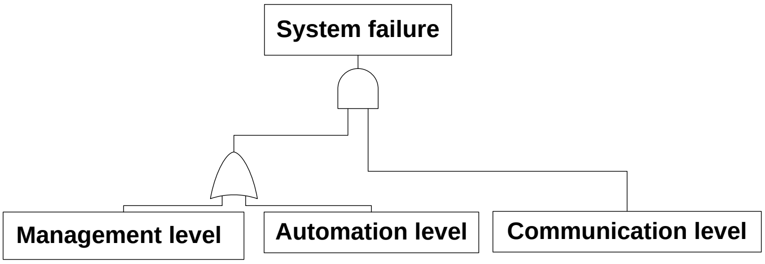


Fig. 36.1 – Presentation of the BAS architecture using the ATA approach

The Attack Tree Analysis is considered as an analytical method in which ways of achieving an undesirable state of the system (in particular, a failure state) are examined. The purpose of the ATA analysis is to assess the reliability and cyber security of the system. This helps architecture developers to understand how the system works with weak points in the project, which can be used by attackers. The ATA analysis shows which requirements for system components need to be increased to ensure cyber security and reliability throughout the life cycle. When using this toolkit, the system is analyzed in the context of the surrounding operating environment to find all possible ways of failure occurrence. When constructing the model in the form of an event tree, two types of gates are used (AND, OR). The event after gate "AND" occurs with simultaneous manifestation of changes at the input of the gate. The event at the output of the "OR" gate arises if at least one change in the state of the component occurs at its input.

Fig. 36.1 shows the upper level tree of the ATA analysis of the BAS architecture, including three levels. The ATA tree allows to prioritize each level when creating a complex failure event of the system as a whole. Fig. 36.1 shows that the communication level has the highest priority and direct connection via the "OR" gate to the system failure state. The other two levels are connected to each other through the "AND" gate, they cannot independently lead the system to a fault state, and system failure occurs only when faults occur at these levels simultaneously. Nevertheless, the probability of such an event must be taken into account.

When there is a need to analyze the cyber-security of the system, we should choose a specific event – a failure or attack on the system component as a target of the attacker, and then determine the immediate, necessary and sufficient reasons for achieving this goal. Such reasons may not be fundamental to a system-wide failure, but they are the immediate causes for this event. They are considered as sub-goals, or targets of the second level of the attacker. In determining all immediate, necessary and sufficient reasons, a step-by-step analysis of the tree from top to bottom is performed until the ATA model resolution limit is reached, that is, the atomic failure event of the BAS component.

Taking into account all possible targets for attacks that can be directed to the system and its components at each level, then it is necessary to consider the scenarios of cyber-attacks.

36.1.2 General scheme of the dependability analysis

Taking into account the positions of reliability and cyber security allows expanding the list of causes of failures and weaknesses of the system within the framework of a unified dependability concept. In the direction of reliability, hardware and software defects, as well as interaction defects due to operating personnel errors and attacks on the system are analyzed. On the cyber security aspect, software vulnerabilities, Trojans and backdoors are analyzed (Fig. 36.2).

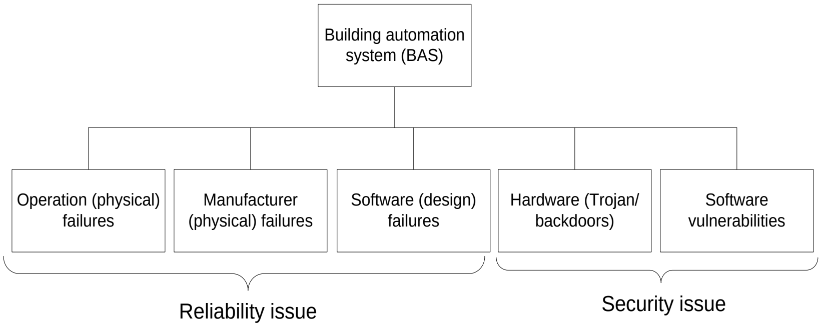

Building automation system (BAS)
Operation (physical) failures
Manufacturer (physical) failures
Software (design) failures
Hardware (Trojan/ backdoors)
Software vulnerabilities
Reliability issue
Security issue

Fig. 36.2 – Causes of failures in BAS components taking into account aspects of reliability and cyber security

36.2 Vulnerability analysis of smart building information and control systems components

According to [1], the BAS architecture has three levels, therefore, vulnerability analysis should be performed for components of these levels. Identifying and assessing the vulnerabilities of these levels helps the developer to manage risks and determine the degree of threat at the design stage of the system. According to the analysis carried out in [2], the main elements of the system architecture that have a high level of threat are FPGA, database, communication. The information obtained in the analysis of vulnerabilities can be used to compile IMECA matrices and forms the basis for designing ATA models.

36.2.1 Analysis of vulnerabilities of FPGA devices

A field-programmable gate array (FPGA) is produced as a ready-to-use electronic device. For application in digital systems, such devices must be programmed. The advantages of FPGA-platforms include simplicity of tuning and cost-effectiveness. In addition, such platforms can be updated during the lifetime, it is simply enough to download a new application code. FPGA-platforms have other advantages, but, nevertheless, their main advantage is the design flexibility. When analyzing the cyber security of FPGA platforms, it is necessary to take into account all the features of the life cycle of both FPGA chips and information and control systems (I&C) based on FPGAs. Participants of the processes are manufacturers of FPGA chips, designers and developers of I&C systems as well as users of I&C systems based on FPGA. Cyber-security analysis for FPGA technology covers the design and development processes as well as the operation of integrated I&C systems. It should be noted that cyber-security vulnerabilities could be introduced by:

- a manufacturer of FPGA chips in the design, production, setup and testing of FPGA microcircuits;
- a developer of I&C systems at the design, coding and testing stages;
- an I&C operator of the system during operation and maintenance.

36.2.2 Analysis of vulnerabilities in databases

Recently, the number of attacks on databases (DBs) has increased. This is due to the growing demand for data stored in the database and the expansion of access to databases via the global network. The databases in I&C systems of smart buildings contain information that is important for the system and its various levels for controlling executive devices.

When we expand access rights to the stored information for several users, this increases the likelihood of data theft. Therefore, in BASs access to the database must be constantly monitored. An attacker seeks to gain access to important information that he can use to attack or monitor the system. Various types of threats that affect the cybersecurity of databases are given below.

1. Abuse of rights and privileges. The threat arises in a situation where database users have more privileges than it is required to perform functional duties. These privileges can be deliberately or unintentionally transmitted to intruders.

2. Vulnerabilities of operating systems, such as Windows, UNIX, Linux, etc., as well as OS services that interact with databases, can act as a means for unauthorized access. Such vulnerabilities can also be used for denial of service (DoS) attacks. As a rule, they are fixed after installing/updating the operating system security patches.

3. Rootkits (rootkits) of databases. A rootkit is a program or procedure that is hidden inside the management system (DBMS) and provides administrative privileges to access data and disable the Intrusion Prevention Systems (IPS). The rootkit can be installed after using the vulnerabilities of the main operating system. Identification of rootkits is performed using periodic audits; when there are no such audits, the presence of a rootkit in the database can remain unnoticed. To gain credentials for entering the database, attackers can use different strategies (social engineering, direct search of passwords), and they can be successful in case of using weak authentication methods. In the presence of a rootkit, the DBMS assumes that the attacker has the identity of legitimate database users.

4. Weakening the requirements for auditing. The presence of simplifications and weaknesses in the mechanisms of DBMS audit and event logging can become a critical threat for the system, especially in industries with strict regulatory requirements. To restore the history, prior to incidents, the protocols PCI, SOX and HIPAA, which allow for advanced logging, are used. It should be noted that the logging of suspicious or undefined operations in the database must be performed automatically. The audit log is the last line of cybersecurity in the database. The records in it allow detecting an intrusion, which in turn will help to track violations of a particular user at a certain point in time.

36.2.3 Analysis of the vulnerabilities in wireless communications

In the architecture of wireless communications, there are four main components [3]. They include the radio frequency data channel; access points providing connection to the network of the organization; transceivers of end devices (laptops, smartphones, etc.); and programs with a user interface. These components may be vulnerable and subject to attack, which will lead to breach of confidentiality, integrity and availability [4]. The following types of attacks on wireless communications are analyzed.

1. Unintentional association, the type of unauthorized access to the company's wireless networks. When a user turns on the computer and connects to a wireless access point that belongs not to a corporate, but

to the neighboring network, it may not even know that this has happened. Such a violation of cybersecurity can reveal valuable information about the company and create a connection between the company's network and a fake network [5]. The same incident can occur with a laptop connected to a wired network.

2. Peer-to-peer networks. Such networks are often organized to exchange data between two wireless devices. Despite the possibility of using enhanced encryption methods, as a rule, they are neglected when creating peer-to-peer networks [6].

3. "Man-in-the-middle" attack: an attacker creates a program access point (AP), which connects corporate users. After that, the attacker connects to a real access point using another wireless card that provides a constant stream of traffic through a transparent hacker network to the real network [7]. Thus, an attacker can listen to the traffic.

4. Denial-of-service attack (DoS). An attacker organizes a constant load on the target access point or network using dummy requests, error messages, messages about premature successful connections, and/or other commands. Due to this attack, users cannot access the network. These attacks are based on abuse of protocols, such as the Extensible Authentication Protocol (EAP).

36.2.4 Scenarios of cyber-attacks on information and control systems of smart buildings

Cyber-attacks are conducted to disrupt the normal operation of the BAS by stealing, modifying or destroying data, or code. One way to conduct cyber-attacks is to hack personal computer systems or I&C systems of organizations, their infrastructure, computer networks, and/or personal computer devices. Typically, the source of cyber-attack is difficult to detect, since an attacker makes efforts to ensure anonymity. Such attacks can be organized not by individuals, but by whole cyber-campaigns within the framework of cyber war or cyber terrorism. The ways to implement cyber-attacks include installing spyware on a PC, destroying the infrastructure of an organization or even a whole state. Every day, the complexity and danger of cyber-attacks increases.

Like random components failures, cyber-attacks can be directed to hardware channels and BAS software. Since the BAS components are accessed from the global network [8], they are all potential targets of cyber-attacks.

Attacks on hardware can use embedded code or errors made to the chip through the fault of the manufacturer. Therefore, a hardware bookmark, virus or worm can be active for some time. Software attacks can be carried out using various tools for monitoring and reading data, for example, scanning the radio channel of wireless devices for transmitting and receiving data.

Scenarios of cyber-attacks on hardware channels or software can cause a system-wide failure through a hardware failure and errors in the software component.

To analyze the cyber security of BAS, it is necessary to analyze and study all possible attacks on the system, to predict how an attacker will attempt to access the system from the inside. [9] The scenarios of cyber-attacks on the BAS can be divided into three parts:

1. The attacker gets access with the help of special tools for monitoring the network. Access is an intermediate goal. At the initial stage, the attacker's goal is to monitor the network and read the inter-level data exchange.

This type of attack cannot be detected for a long time, since it often has no signs of detection during system operation. The way to counter these kinds of attacks is to enhance the cyber security of the network.

2. In the second part of the scenario, the attacker's goal is to disrupt the system. This can be performed by introducing malicious code (virus, worm) into the system. The recovery time of the system after this attack is different and depends on the level that has been attacked:

a) if the attacker seeks to capture the automation level and stop one of its components, it is possible to detect a system error and restore the code by changing or updating the system during recovery. Without removing the code, the system can also save partial operability;

b) if the target of the attacker is the management level, then the recovery process will be difficult, since this level controls all system tasks and it is difficult to conduct maintenance without a complete shutdown of the system. A cyber-attack on the management level causes a long recovery time and high costs for renewal.

3. If an attacker becomes aware of design errors, then cyber attacks can be carried out directly.

The described stages of cyber-attack scenarios are systematized and presented in Fig. 36.3. This scheme can be used to understand the attacker's strategy when he tries to access and attack BAS.

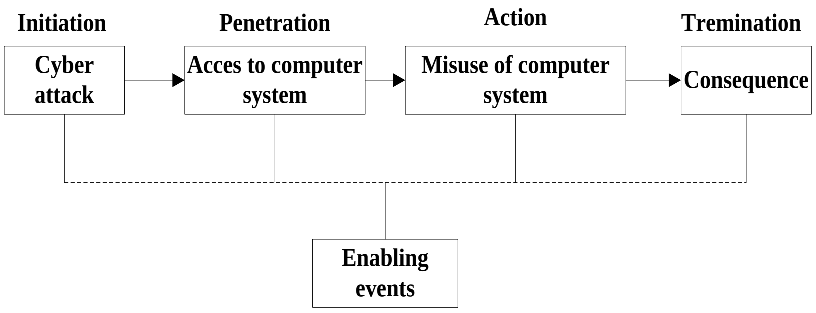


Fig. 36.3 – The main stages of cyber-attack scenarios on BAS

36.3 Development of models for assessing the cyber security of smart building I&CS using FMECA and ATA technologies

The overall goal of attacks can be characterized as a violation of the performance of system functions defined at the design stage. Identification of failures implies the definition of the characteristics of potential mechanisms for their occurrence and an assessment of the probability of failure in real systems during the operational phase. In order to protect the system, developers and users should find answers to the three following questions: "How the system can fail?" "What consequences will the failure have?", and "How much can the system handle?". To answer these questions, FMECA and ATA techniques have been developed, which will be considered further for assessing cyber-attacks on BAS architecture components.

36.3.1 BAS analysis using the FMECA and IMECA methodologies

Failure Modes and Effects Analysis (FMEA) is a technological process that is used to study the potential consequences of failures of the system on it and its environment [10]. If this takes into account the criticality of failures, then the method is called Failure Modes, Effects and Criticality Analysis (FMECA) [11]. FMEA and FMECA are the most popular tools for finding design defects during the development of the system. They also facilitate the search and elimination of defects during the operation of the system. In this paper, in addition to these methods, the method of assessing the types, consequences and criticality of external influences – IMECA – is also used [12]. Unlike FMEA and FMECA, it considers system failures caused by malicious external actions (intrusions). In accordance with the scenario of cyber-attacks discussed in the previous subsection, we can apply IMECA to analyze the cyber security of a BAS within this scenario and measure the level of failures of system architecture components. According to the analysis of cyber security, the components of the system can be divided into subsets of elements (hardware, software). In this paper, FMEA was used to illustrate the impact of attacks on the operability of the system hardware (Table 36.1). IMECA is used to analyze the software component of the system, as shown in Table 36.2.

Table 36.1 – System FMECA analysis of BAS according to cyber-attack scenarios

Architecture level	**Failure type**	**Failure cause**	**Failure consequences**
Management level	Hardware	Operator errors or design defects	This level is represented as a system control unit; a failure will lead to the system shutdown
Management level	Hardware	Design errors or intrusion into components	System downtime and recovery time will be long and costly, since there is a need to modify the hacked component
Automation level	Hardware	End device shutdown	The system works without downtime and with limited data

			entry. The recovery time will be short, since the hacked sensor can be quickly replaced

Table 36.2 – System IMECA analysis of BAS according to cyber-attack scenarios

Architecture level	Component	Types of attack	Cause of failure	Impact on operabilit y	Consequences	
					Cybersecurity	Availability
Communication level	Wi-Fi	Passive	An attacker has access to the wireless network and monitors all transmitted data	Failures	An attacker knows all the transmitted data	Impact on availability is not provided
		Active	After an attack, the access is obtained to enter the network; an attacker breaks the connection between the levels using various tools (viruses, bookmarks)	Denials	The purpose of the attack is to disable the system and completely disable the security system	Full impact on availability, as the system goes into the failure mode until the vulnerability is identified and removed

Management level	DB	Passive	After a successful cyber-attack, an attacker gets access to a database for reading and recording information	Failures	The security of the system is compromised, since an attacker controls the data inside the system	The availability of the system depends on the purpose of the attacker: he can either steal data or damage them and disable the system

36.3.2 Models of components of the BAS architecture in the form of an ATA tree

To begin with, the ATA models presented in Figs. 2.4-2.6 are considered. Increasing the Attack Trees was carried out gradually from below-upwards. Initially, the trees of the components of individual levels were built (examples are given: the ZigBee protocol of the switching level in Fig. 36.5 and the FPGA controllers of the automation level in Fig. 36.4).

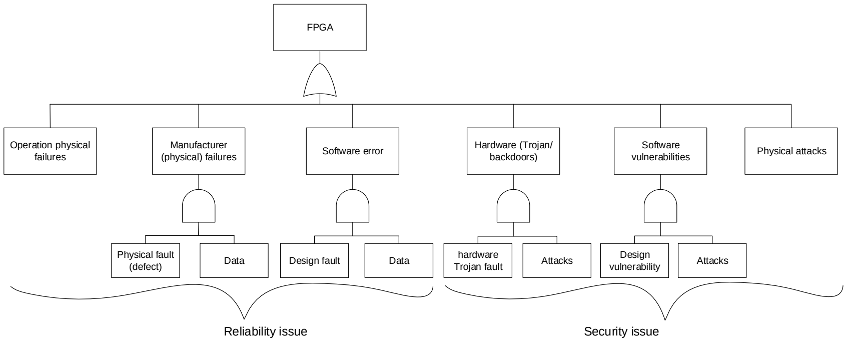


Fig. 36.4 – Attack Tree model of FPGA controllers

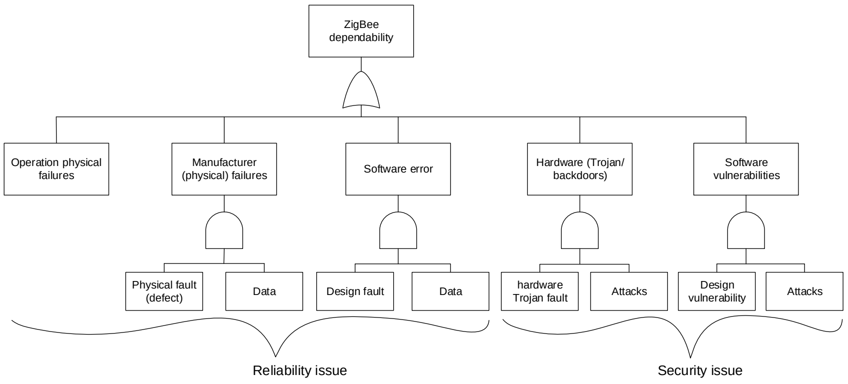


Fig. 36.5 – Attack tree model of ZigBee protocol

Then, an ATA tree was built for the entire BAS system. For this tree, calculations were made of the probability of a failure in a subset of cybersecurity, the results of which are summarized in Table 36.3.

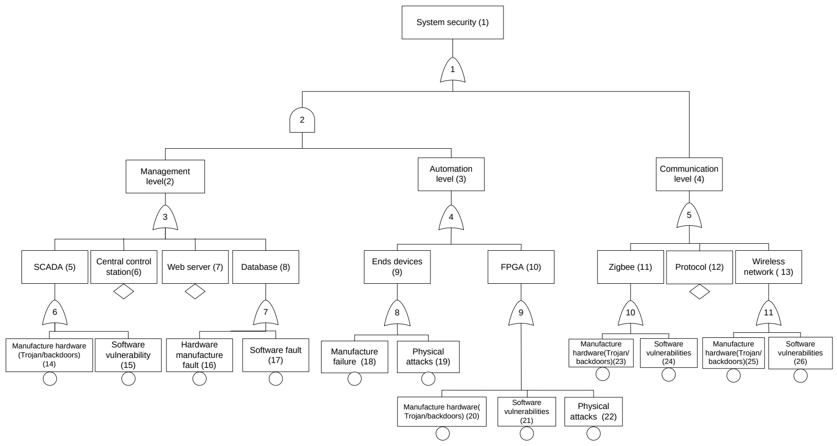

System security (1)
1
2
Management level(2)
Automation level (3)
Communication level (4)
3
4
5
SCADA (5)
Central control station(6)
Web server (7)
Database (8)
Ends devices (9)
FPGA (10)
Zigbee (11)
Protocol (12)
Wireless network (13)
6
7
8
9
10
11
Manufacture hardware (Trojan/backdoors) (14)
Software vulnerability (15)
Hardware manufacture fault (16)
Software fault (17)
Manufacture failure (18)
Physical attacks (19)
Manufacture hardware(Trojan/ backdoors)(23)
Software vulnerabilities (24)
Manufacture hardware(Trojan/ backdoors)(25)
Software vulnerabilities (26)
Manufacture hardware(Trojan/backdoors) (20)
Software vulnerabilities (21)
Physical attacks (22)

Fig. 36.6 – Attack tree model of BAS components for assessing static indicators of cyber security

Table 36.3 – Calculation of a failure probability of the information and control system in a smart building according to cyber security indicators

Architecture level	Component No	Vulnerability class of BAS component	Probability of successful attack	
Management level	1	Manufacture hardware (Trojan/backdoors) (14)	0.0000842	Probability of failure of BAS as a result of external influences (attacks on vulnerabilities) 0.000281468
	2	Software vulnerability (15)	0.0000458	
	3	Hardware manufacture (20)	0.0000789	
	4	Software fault (21)	0.0000523	
	5	Central control station (6)	0.0000157	
	6	Web server (7)	0.0000791	
Automation level	7	Manufacture failure (16)	0.0000825	
	8	Physical attacks (17)	0.0000423	
	9	Manufacture hardware (Trojan/backdoors) (22)	0.0000373	
	10	Software vulnerability (23)	0.0000656	
	11	Physical attacks (24)	0.0000474	
Communication level	12	Manufacture hardware (Trojan/backdoors) (18)	0.0000063	
	13	Software vulnerability (19)	0.0000888	
	14	Manufacture hardware (Trojan/backdoors) (25)	0.0000764	

	15	Software vulnerability (26)	0.0000678	
	16	Protocol (13)	0.0000421	

36.3.3 Models of BAS architecture in the form of FTA and AvTA trees

The approach proposed in the work allows to identify the causes of failures in a complex multi-level system, which is especially important when analyzing the vulnerabilities of individual components of lower levels. The model considered earlier (Fig. 36.1) needs to be improved for the subsequent combination of two types of failure trees (FTA – Fault Tree Analysis and ATA – Attack Tree Analysis) and accounting for recovery processes (AvTA-Availability Tree Analysis).

The developed BAS models in the form of separate trees (FTA, ATA and AvTA) are presented in Fig. 36.7 … Fig. 36.9. With the help of the constructed trees, the calculation of the probability of the system failure due to software defects and attacks on vulnerabilities has been made, the results of which are presented in Table 36.4.

Table 36.4 – Calculation of the probability of failure-free operation of the smart building I&C system in terms of reliability and cyber security

Arch. level	Subset	Component	Name of the AvTA input parameter	Value (probability)	
Hardware	Reliability	FPGA	physical operation failure (hardware)	0.0012	Probability of system failure =0.001590089
			physical operation failure (soft hardware error)	0.002	
			manufacture failure (hardware)	0.25	
		ZigBee	physical operation failure (hardware)	0.0021	
			physical operation failure (soft hardware error)	0.1265	
			manufacture failure	0.15157	

			(hardware)		
		Database	physical operation failure (hardware)	0.17664	
			physical operation failure (soft hardware error)	0.20171	
		Rec/hardware	recovery depending on type of failure	0.8	
	Security	FPGA	intrusion failure (severe hardware vulnerability)	0.25185	
			intrusion failure (soft hardware vulnerability)	0.27692	
		Ahw	attack by intruder (hardware)	0.30199	
		Rec/software	recovery depending on type of failure	0.5	
Software	Reliability	FPGA	failure caused by design fault (software)	0.005	
			failure caused by software design (soft software error)	0.015	
			failure caused by ageing(software)	0.025	
		ZigBee	failure caused by design fault (software)	0.035	
			failure caused by software design (soft software error)	0.045	
			failure caused by ageing(software)	0.055	
		Database	failure caused by design fault (software)	0.065	
			failure caused by software design (soft software error)	0.075	
			failure caused by ageing(software)	0.085	

		Rec/hardware	recovery depending on type of failure	0.8	
	Security	FPGA	intrusion failure (severe software vulnerability)	0.0215	
			intrusion failure (soft software vulnerability)	0.078	
			attack by intruder (software)	0.325	
		Database	intrusion failure (severe software vulnerability)	0.445	
			intrusion failure (soft software vulnerability)	0.59675	
			attack by intruder (software)	0.7485	
		ZigBee	intrusion failure (severe software vulnerability)	0.90025	
			intrusion failure (soft software vulnerability)	0.0252	
			attack by intruder (software)	0.0785	
		Rec/software	recovery depending on type of failure	0.5	

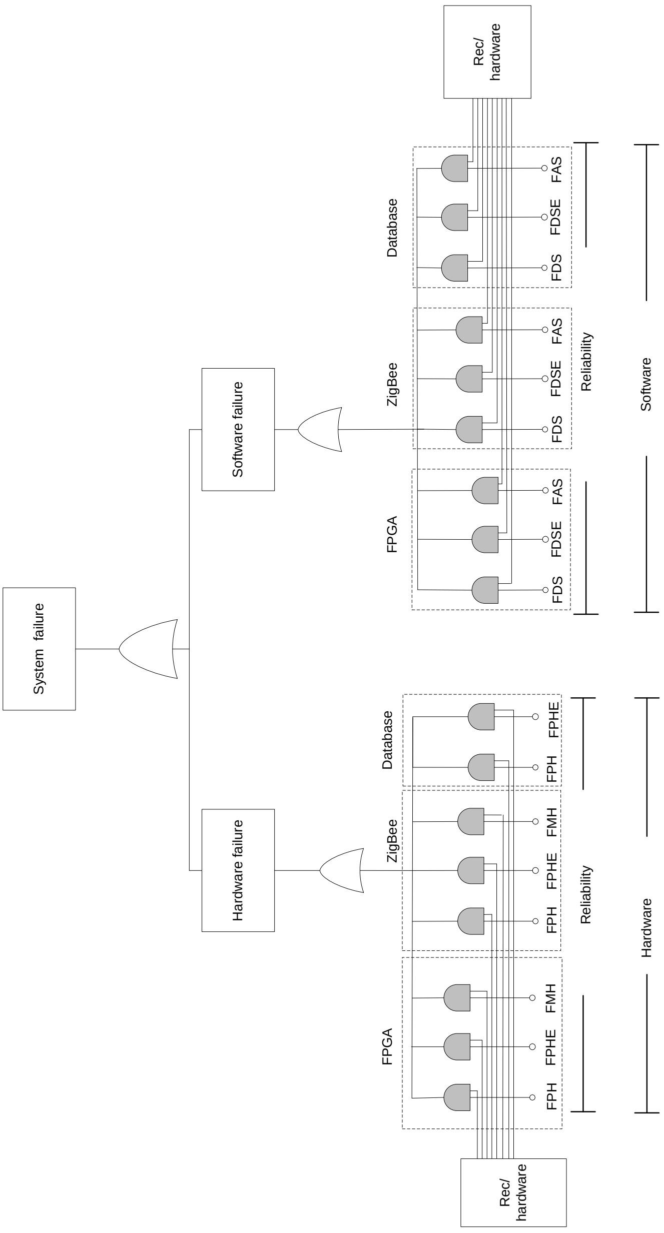


Fig. 36.7 – Fault tree model of BAS components

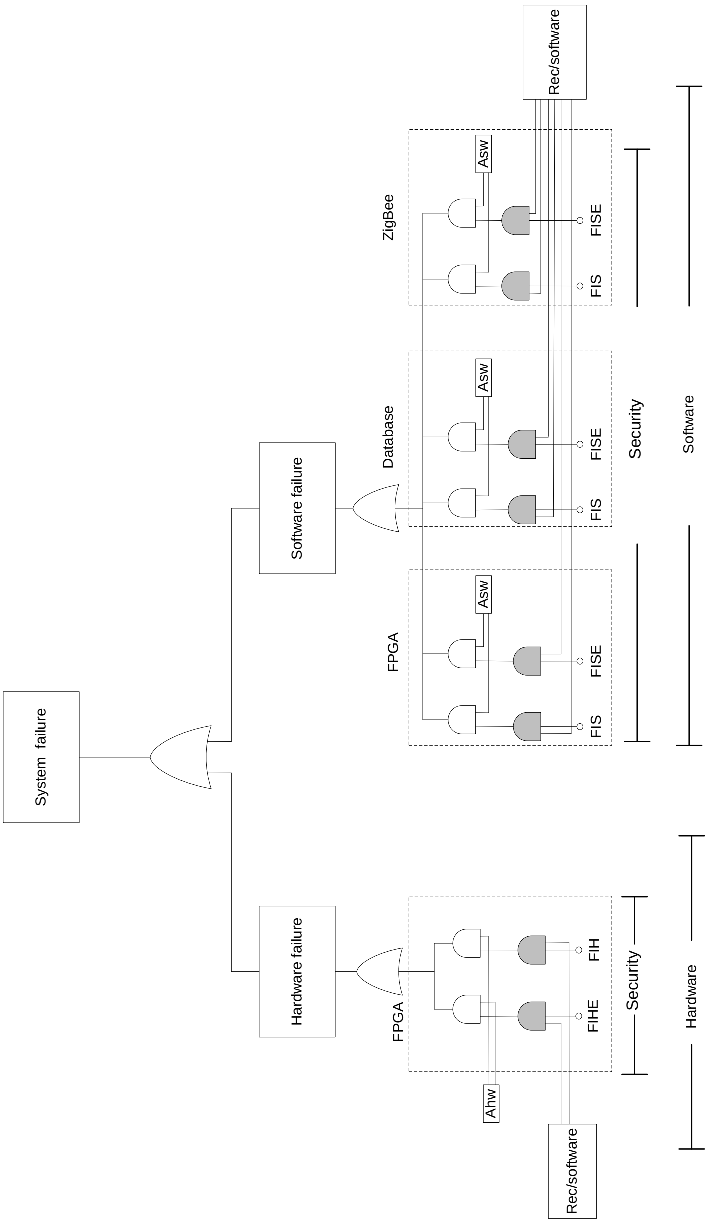


Fig. 36.8 – Attack tree model of BAS components

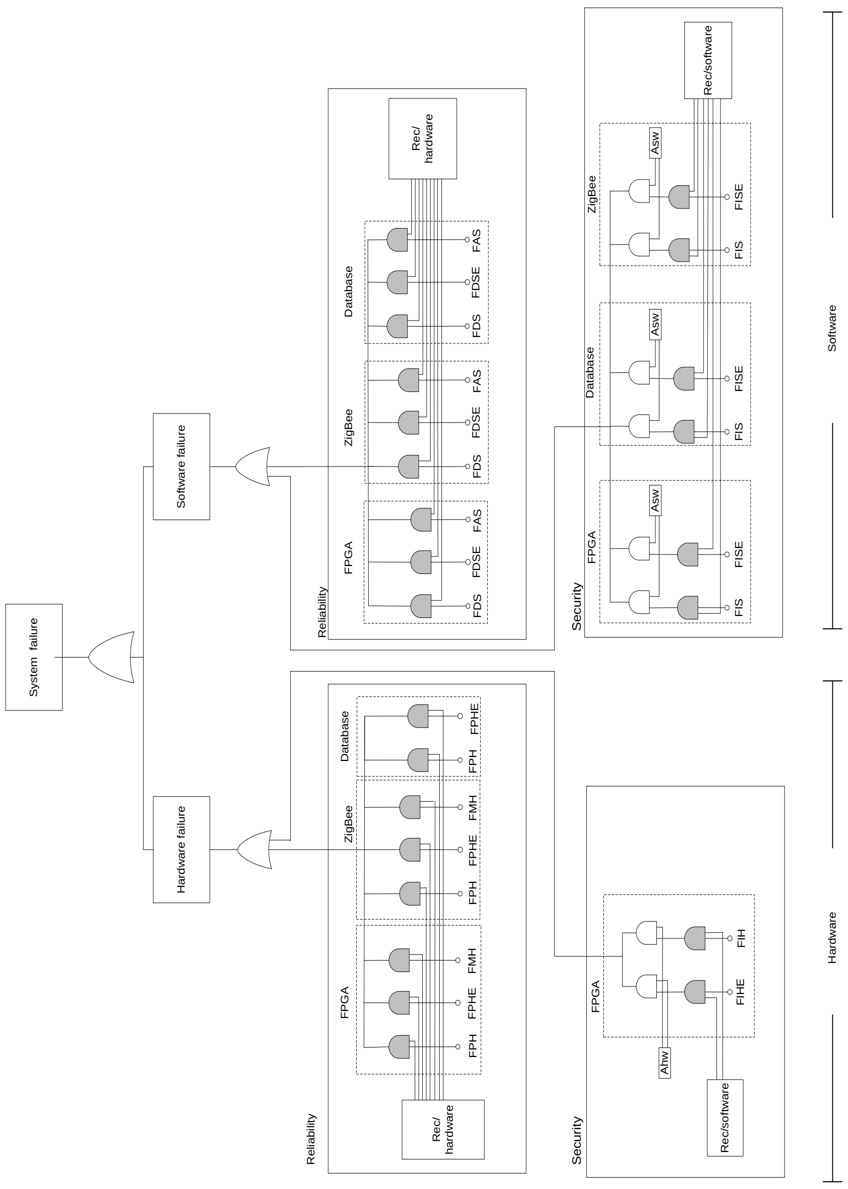


Fig. 36.9 – Availability tree model of BAS components

36.4 Scaling of models for assessing the reliability and cyber security of smart building I&C systems

The project of intellectualization of the university campus buildings presented in Fig. 36.10 provides the installation of sensors and actuators in buildings of different categories. In ordinary residential buildings, the elements of the low-level intelligent building systems linked to the BAS are located, the control level of which is located in a separate data center. The data center is located within the reach of the local network of the communication level. Thus, each zone, denoted as "Arean" in Fig. 36.10, due to ensuring the requirements for autonomy of functioning, is considered as a BAS of the first level (Level 1), which is shown in Fig. 36.10. The administrative building in the "Area 1" zone also has intelligent systems, as well as the servers on which the private cloud is deployed (Private Cloud). This cloud provides a management level over the entire campus. To communicate with the cloud, other zones use the resources of the Internet, because the distances between them cannot be limited to the use of the local network.

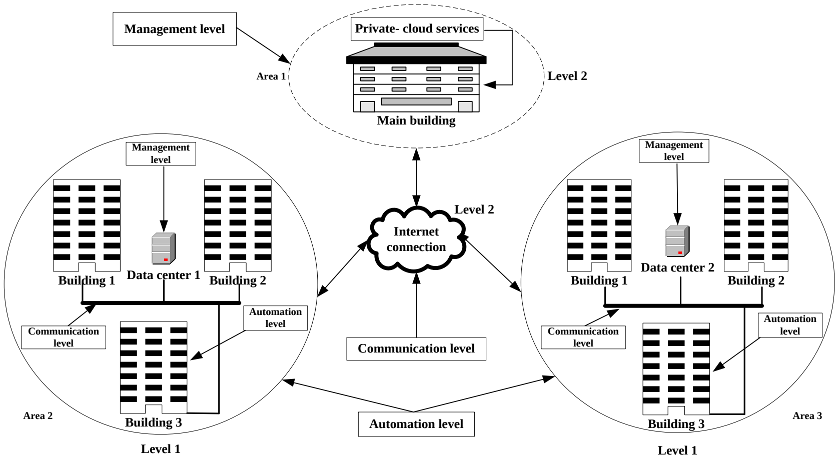


Fig. 36.10 – Design of the architecture of the intellectualization system for the smart university campus

Thus, when scaling tree models of failures and attacks on the university campus according to Fig. 36.10, three levels of architecture are also pointed out. At the management level, Private Cloud servers deployed in the administrative building are considered. The communication level unites all Internet connections between cloud servers and the BAS residential buildings. The automation level is associated with the BAS of residential buildings of the first level.

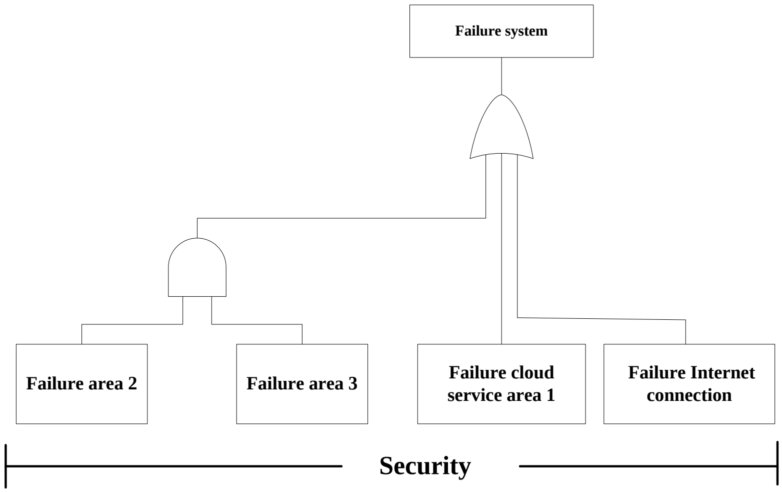


Fig. 36.11 – The tree of attacks (ATA) on components of the university campus intellectual system

When constructing an Attack Tree model for the university campus systems (Fig. 36.11), generalized indicators of the non-failure operating probability of individual zones, cloud servers and the communication level are considered. The last two NOPs were identified in [13,14], and the NOP of the BAS level is determined by the previously developed models of cyber security (Fig. 36.8). The Attack Tree of the university campus is constructed using assumptions about the impossibility of hacking the whole system only by attacking one of the BASs of the first level. This means that attackers in order to transfer the entire system to the failure mode must either crack both BASs of the first level at the same time, or disrupt the cyber security in communication and management levels.

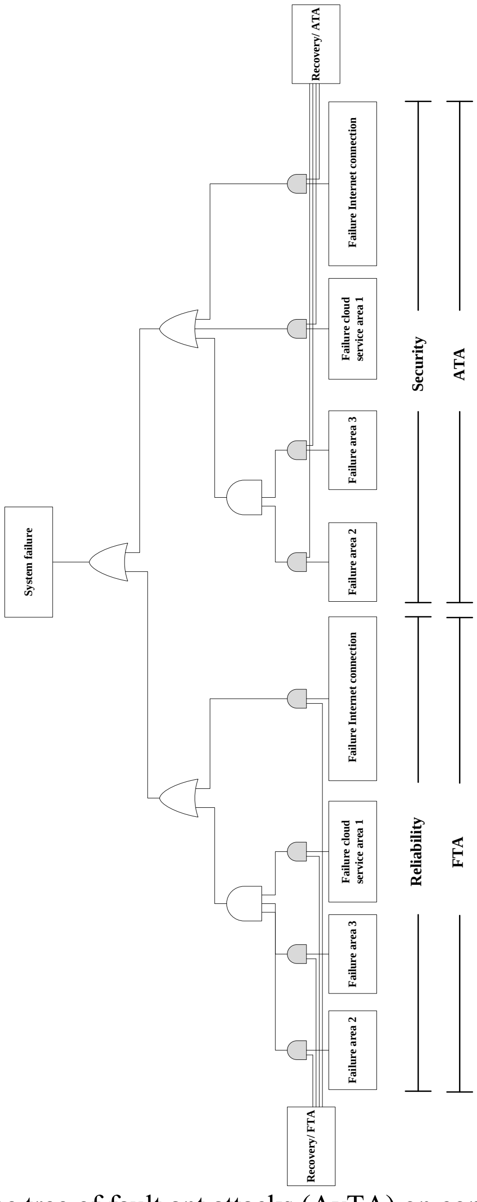


Fig. 36.12 – The tree of fault ant attacks (AvTA) on components of the university campus intellectual system

The Fault Tree model of the university campus intellectual system (Fig. 36.13) also considers the generalized non-failure operating probability indicators of the BAS level obtained with the help of previously developed FTA-models (Fig. 36.7). NOPs of cloud servers and the level of communication were defined in [15]. Due to the autonomy of the operation of systems in different zones, a system-wide failure occurs only if the BASs of these zones simultaneously shutdown, or if the communication level is damaged.

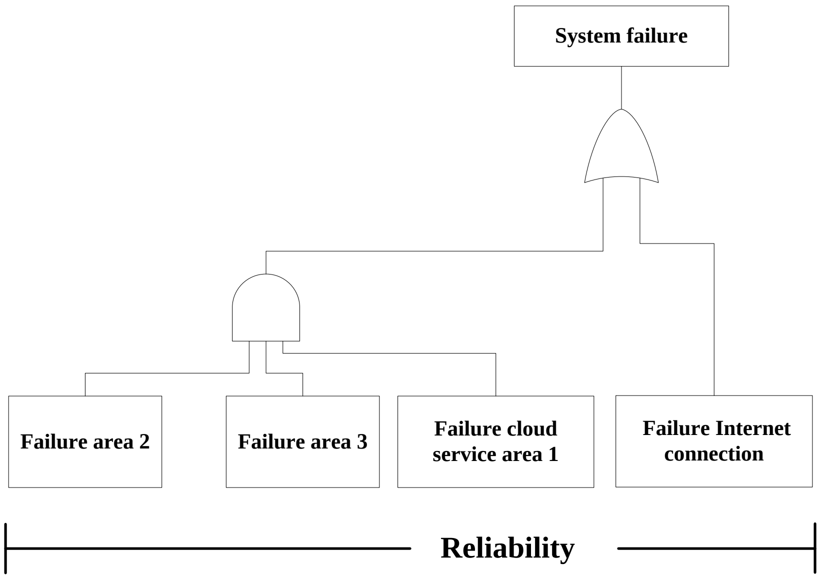


Fig. 36.13 – Fault tree (FTA) model for components of the university campus intellectual systems

Table 36.5 shows the results of calculations of the NOPs for the intellectual system of the university campus, and the AvTA model of the campus is presented in Fig. 36.12.

Table 36.5 – Calculation of the NOP for I&Cs of the smart building according to indicators of reliability and cyber security

Type of	Issues	Parameters	Probability	

Tree					
FTA	Reliability	Failure area 2	0.0012	System probability to failure with recovery=0.006187324	System probability to failure without recovery=0.011139648
		Failure area 3	0.002		
		Failure – cloud services –area 1	0.25		
		Failure Internet connection	0.0021		
		Recovery /FTA	0.8		
ATA	Security	Failure area 2	0.005		
		Failure area 3	0.015		
		Failure – cloud services –area 1	0.0025		
		Failure Internet connection	0.0065		
		Recovery /ATA	0.5		

According results of calculations, it is possible to draw a conclusion that accounting factors of recovery and blocking of attacks allows to specify the importance of NOP value for the intellectual system of the university campus by an order of magnitude.

36.5 Development of a conceptual model for the I&Cs functioning of the smart building taking into account recovery and maintenance

In general, the BAS conceptual model should cover a full set of reasons for system shutdown [16]. At the same time, the dimension and complexity of the model cause the search for ways of its decomposition into smaller models describing the mutually independent causes of failures. Thus, for models of hardware and software failures, it is possible to construct both a generalized model and two separate availability models with the subsequent multiplication of their resulting availability coefficients (or functions).

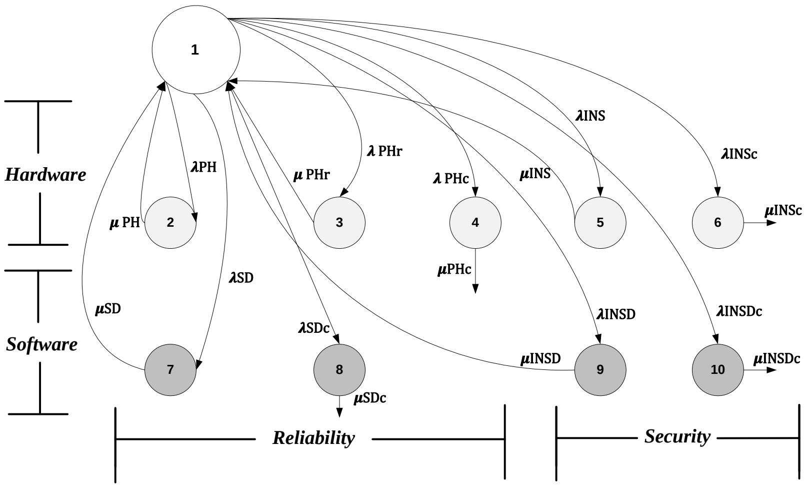


Fig. 36.14 – Conceptual scheme for constructing the general model of BAS functioning taking into account two groups of failure causes

The general concept of building a model with two groups of failure causes (subsets of reliability and cyber security) is presented in Fig. 36.14. The upper level is occupied by the initial working state of the S_1 system. The level below is a subset of the hardware states – the group of states S_2 ... S_6 caused by the manifestations of the faults in hardware. The lower part of the Fig. shows the subset of the states of the software tools S_7 ... S_{10}. Under the condition of changing the parameters of manifestation defects in design and interaction (intrusions), the model will expand in the direction of four vectors from states S_4, S_6, S_8, S_{10}, to final states in which the parameter change stops. Causes and events, which change the parameters of the manifestation of design faults, are described in detail in [17]. Explanations to the definition of the input parameters of the conceptual model are given in Table 36.6.

Table 36.6 – Input parameters of the conceptual model for the I&CS of the smart building

Parameter notation	Detailed description of the input parameter
λPH	Physical operation failure (hardware)

μPH	Physical operation failure (hardware/repair)
λPHr	Physical failure operation (soft error)
μPHr	Physical operation failure (soft hardware error/restart)
λPHc	Physical manufacture failure (hardware)
μPHc	Manufacture failure (hardware/changing design)
λINS	Intrusion failure (soft hardware vulnerability)
μINS	Intrusion failure (soft hardware vulnerability /restart)
λINSc	Intrusion failure (severe hardware vulnerability)
μINSc	Intrusion failure (severe hardware vulnerability/changing design)
λSD	Failure caused by design fault (software)
μSD	Soft error caused by design fault (software/restart)
λSDc	Failure caused by design fault (software)
μSDc	Failure caused by design fault (software/changing code)
λINSD	Intrusion failure (soft software vulnerability)
μINSD	Intrusion failure (soft software vulnerability/restart)
λINSDc	Intrusion failure (severe software vulnerability)
μINSDc	Intrusion failure (severe software vulnerability/changing code)

The logic of the mechanisms for changing the parameters of attacks on the vulnerabilities of the BAS architecture component is as follows. Initially, at the time of putting the system into operation, it contains some set of component vulnerabilities. At the same time, this set contains vulnerabilities known from records in open repositories as well as the so-called "zero day" vulnerabilities (about which there is no information in open repositories).

In the process of functioning, the following events that affect the change in the number of vulnerabilities in the system can take place:

- elimination of single vulnerabilities (both open and "zero day") after attacks of intruders;

- elimination of single vulnerabilities (both open and "zero day") after their detection by users;

- elimination of a group of open vulnerabilities resulting from cyber security maintenance procedures;

- introduction of new vulnerabilities as a result of BAS reconfiguration or software updating.

Fig. 36.15 graphically shows how to resolve single (a) and group (b) vulnerabilities of BAS components.

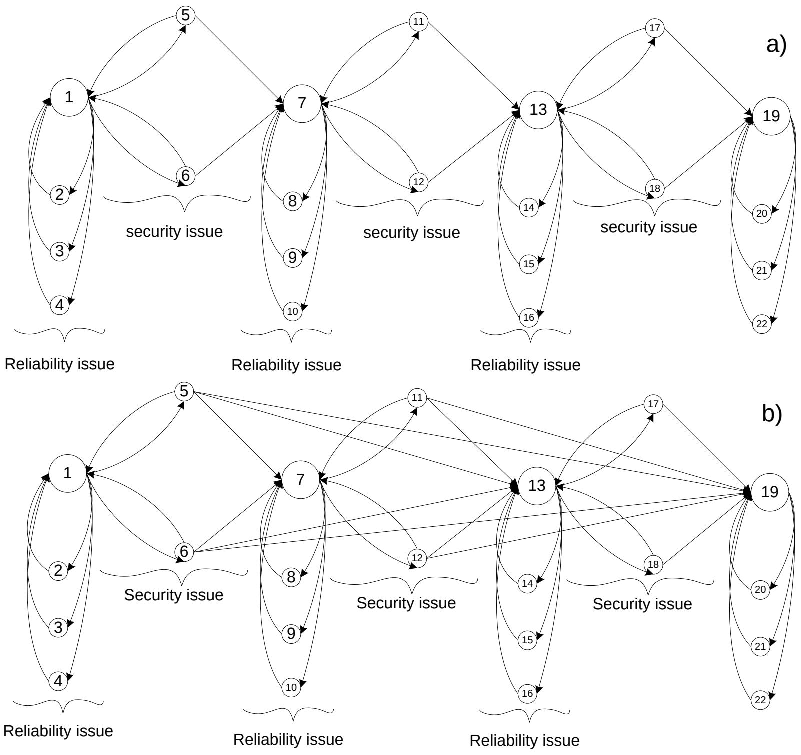


Fig. 36.15 – Dynamics of change in the BAS conceptual model when performing security maintenance procedures with elimination of single (a) and group (b) vulnerabilities

In the interest of further research, it is assumed that the number of failure causes is limited to two subgroups: software defects due to design errors and attacks on software component vulnerabilities. Taking into account such an assumption, the dimension of the conceptual model decreases, as shown in Fig. 36.16, a. Fig. 36.16, b shows a Markov graph of the conceptual model, taking into account the second assumption about the sequential manifestation of defects and attacks on vulnerabilities. In addition, it is assumed that a defect or vulnerability will be eliminated with probabilities PR (PS).

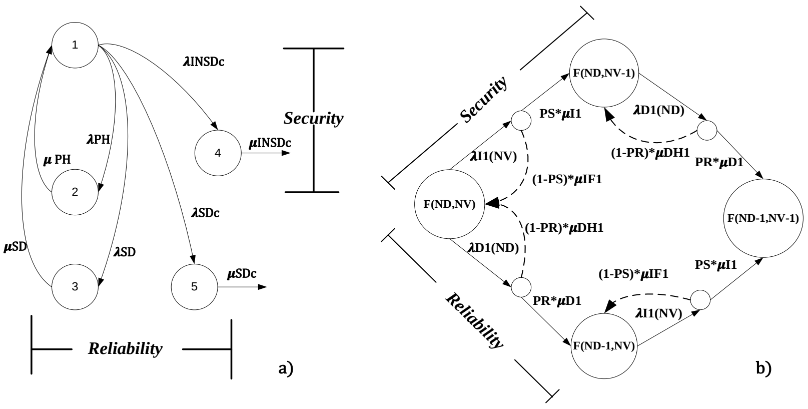


Fig. 36.16 – A simplified graph of the BAS conceptual model (a) and with consideration of the PR (PS) probabilities to eliminate defects and vulnerabilities (b)

In the future, when modeling a system with a number of defects and vulnerabilities more than 1, the dimension of the graph shown in Fig. 36.16, b will increase, but the depicted lozenge will remain the reference fragment of the BAS model.

Conclusions

The chapter presents the existed techniques and conceptual approaches to assessing the reliability and cybersecurity of information and control systems using models in the form of fault and Attack Trees as well as graph models of states and transitions.

The reliability and cyber-security models BASs using AND-OR trees for analysis of failures and attacks has been described. This allowed taking into account the influence of faults and vulnerabilities of BAS components on the probability of failure.

The Attack Tree models for the BAS components and for the system as a whole are considered as well as Fault Tree Models and combined failure and attack models (AvTA), which allow considering the recovery of operability and blocking of attacks.

From the practical point of view, described models and techniques are important as allowing choice a non-maintenance BAS component, and develop more detailed requirements and techniques for assessing the reliability and cyber security.

Questions to self-checking

1. Please describe the main components of Building automation system (BAS) architecture.
2. Which are the main differences between Attack Tree Analysis (ATA), Fault Tree Analysis (FTA) and Availability Tree Analysis (AvTA)?
3. Which are typical vulnerabilities of FPGA devices?
4. Which are typical vulnerabilities in databases?
5. Which are typical vulnerabilities in wireless communications?
6. Which are probable scenarios of cyber-attacks and their consequences for BAS states?
7. Please, describe the main procedures of FMECA and FTA technologies
8. Please, describe the main issues of IMECA and ATA technologies
9. Which are the main steps of modeling of BAS architecture components by use of the ATA?
10. Which states are possible in conceptual model for the BASs functioning taking into account strategies of recovery and maintenance?

References

1. Farooq, Umer, Marrakchi, Zied, Mehrez, Habib. Tree-Based Heterogeneous FPGA Architectures – New York Springer Science+Business Media. 2012. 188 p.

2. Rie Higuchi. Building automation and control systems. The United Kingdom. A multi client study – BSRIA Limited Old Bracknell Lane West, Bracknell, 2013. – 203 p.

3. Hatambeiki, A. Wireless Network Security – San Francisco, California, 2004. – 132 p.

4. D. Nagamalai, B. Dhinakaran, P. Sasikala, S. Lee and J. Lee, Security Threats and Countermeasures in WLAN, – Technologies for Advanced Heterogeneous Networks. AINTEC 2005. Lecture Notes in Computer Science, – vol 3837, – pp. 168-182, – 2005, doi: 10.1007/11599593_13.

5. Vishali R. Security in Wireless Local Area Networks. – International Journal of Computer Science and Information Technology Research. – 2014. – Vol.2, Issue 2. – P. 472-483.

6. K. Scarfone, D. Dicoi, M. Sexton and C. Tibbs, Guide to securing legacy IEEE 802.11 wireless networks, – NIST Special Publication 800-48, – 2008, doi: 10.6028/NIST.SP.800-48r1.

7. R. Jain, Wireless LAN Security II: WEP Attacks, WPA and WPA2, –Washington: University in Saint Louis, – 2009. – 33 p.

8. Mustafa Qahtan Abdulmunem Al-Sudani, V. S. Kharchenko, D. D. Uzun. Vulnerability analysis of wireless networks: case for smart building automation system – Radioelectronic and computer systems. - 2015. – Vol. 2. - P. 69–76.

9. Kharchenko V., Ponochovnyi Y., Abdulmunem AS.M.Q., Andrashov A. Availability Models and Maintenance Strategies for Smart Building Automation Systems Considering Attacks on Component Vulnerabilities. – Advances in Intelligent Systems and Computing, Vol. 582, 2017, P. 186-195. DOI: 10.1007/978-3-319-59415-6_18

10. Technical manual. TM 5-698-4, Failure Modes, Effects and Criticality Analyses (FMECA) for Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) Facilities. – Department of the Army Washington, DC, imp. 29 September 2006. – 75 p.

11. X. Cheng, Z. Xing, Y. Qin, Y. Zhang, S. Pang and J. Xia, Reliability Analysis of Metro Door System Based on FMECA, –Journal of Intelligent Learning Systems and Applications, – vol. 05, no. 04, – pp. 216-220, – 2013, doi: 10.4236/jilsa.2013.54024

12. E. Babeshko, V. Kharchenko and A. Gorbenko, Applying F(I)MEA-technique for SCADA-Based Industrial Control Systems Dependability Assessment and Ensuring, – 2008 Third International Conference on Dependability of Computer Systems DepCoS-RELCOMEX, Szklarska Poreba, – pp. 309-315, – 2008. doi: 10.1109/DepCoS-RELCOMEX.2008.23.

13. T. Novak and A. Treytl, “Functional safety and system security in automation systems - a life cycle model”, – In 2008 IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, – pp. 311-318, – 2008, doi: 10.1109/ETFA.2008.4638412.

14. Feruza Sattarova, Y. Tao-hoon Kim, IT Security Review: Privacy, Protection, Access Control, Assurance and System Security, – In International Journal of Multimedia and Ubiquitous Engineering, – Vol. 2, No. 2, – pp. 17-31, – 2007.

15. Q. Yu and R. J. Johnson, Smart grid communications equipment: EMI, safety, and environmental compliance testing considerations, – Bell Labs Technical Journal, – vol. 16, no. 3, pp. 109-131, – Dec. 2011, doi: 10.1002/bltj.20525

16. K. S. Trivedi, D. S.fdc Kim, A. Roy and D. Medhi, Dependability and security models, – 7th International Workshop on Design of Reliable Communication Networks, – Washington, DC, – pp. 11-20, – 2009. doi: 10.1109/DRCN.2009.5340029.

17. B. Joshi, D. Pradhan and S. Mohanty, Fault Tolerant Nanocomputing, – Lecture Notes in Electrical Engineering, – pp. 7-27, – 2010, doi: 10.1007/978-90-481-8540-5_2.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ К РАЗДЕЛУ 36

AP – Access point
ATA – Attack Tree Analysis
AvTA – Availability Tree Analysis
BAS – Building automation system
DB – Database
DBMS – Database management system
DoS – Denial of service
EAP – Extensible Authentication Protocol
FMEA – Failure Modes and Effects Analysis
FMECA – Failure Modes, Effects and Criticality Analysis
FPGA – Field-programmable gate array
FTA – Fault Tree Analysis
I&CS – Information and control systems
IMECA – Intrusion Modes, Effects and Criticality Analysis
IPS – Intrusion Prevention System
NOP – Non-failure operating probability
PC – Personal Computer

АННОТАЦИЯ

В разделе представлены модели надежности и кибербезопасности информационно-управляющих систем умных домов с использованием И-ИЛИ деревьев анализа отказов и атак учитывающих влияние дефектов и уязвимостей различных компонент их архитектуры и параметров процессов восстановления работоспособности и блокировки атак, позволяющих рассчитать вероятности отказа систем. Учет надежности и кибербезопасности позволяет расширить перечень причин отказов и слабых мест системы в рамках единой концепции гарантоспособности. По направлению надежности анализируются аппаратные и программные дефекты, а также дефекты взаимодействия вследствие ошибок обслуживающего персонала. По аспектом кибербезопасности анализируются уязвимости программных средств, троянские программы и бэкдоры.

У розділі представлені моделі надійності та кібербезпеки інформаційно-керуючих систем розумних будинків з використанням ТА-АБО дерев аналізу відмов і атак шляхом урахування впливу дефектів і вразливостей різних компонент їх архітектури і параметрів процесів відновлення працездатності і блокування атак, що дозволяє розрахувати ймовірності відмови систем. Врахування позицій надійності та кібербезпеки дозволяє розширити перелік причин відмов та слабких місць системи в рамках єдиної концепції гарантоздатності. За напрямком надійності аналізуються апаратні та програмні дефекти, а також дефекти взаємодії внаслідок помилок обслуговуючого персоналу. За аспектом кібербезпеки аналізуються вразливості програмних засобів, троянські програми та бекдори.

Building automation systems models as failure and attack tree and states graph are discussed in the section. The further development was given to the reliability and cyber security model of information and control systems of smart buildings using AND-OR trees of faults and attacks analysis by taking into account the influence of the defects and vulnerabilities of various components of their architecture and the

parameters of the processes of recovery and blocking of attacks, which allows to calculate the probability of failure of the system. Consideration of the reliability and cyber security positions allows to expand the list of causes of failures and weaknesses in the system within the framework of a single concept of dependability. Hardware and software defects as well as defects in interaction due to operating personnel errors and attacks on the system are analyzed in the direction of reliability. The cyber security aspect analyzes vulnerabilities in software, Trojan programs and backdoors.

37 ASSESSMENT OF SMART BUILDING AUTOMATION SYSTEMS AVAILABILITY AND SECURITY CONSIDERING MAINTENANCE STRATEGY

Modification of software tools of different architecture levels of the smart building BAS due to the elimination of design defects and patching of vulnerabilities leads to a change in the parameters of the failure and recovery flows of the system. As it was shown in the previous Chapters, it is preferable to use the apparatus of Markov and semi-Markov processes to study systems with variable parameters [1,2]. In [3], a systematic approach to the construction of multi fragment models is developed, and in [4], models that take into account reliability and security factors for web systems have been developed. However, in known studies, the influence of different maintenance strategies concerning these factors has not been investigated.

Thus, it is necessary to choose a more acceptable approach for constructing Markov models of BAS availability for common and separate maintenance, taking into account the gradual elimination of software defects and vulnerabilities.

37.1 Formalization of mathematical models for availability of intelligent building I&CS

When studying planning and maintenance procedures of BAS architecture software components, an important step is to obtain quantitative values of the probabilistic components of their availability. The use of the Markov modeling apparatus is associated with a certain set of constraints, which does not allow to construct and apply a single unified model. The output is the construction of a complex of models, in which each model allows to obtain similar result indicators, which are convenient for making comparisons and searching for optimal solutions.

The main aspect of modeling the functioning of BAS architecture software components is accounting for the manifestation and elimination of limited sets of software defects and vulnerabilities, and these sets are considered as non-overlapping.

The second aspect is maintenance, in the course of which it is possible to identify and eliminate both defects and vulnerabilities. Maintenance procedures can be carried out throughout the BAS lifecycle, or be limited to a certain number of procedures.

The third aspect is the composition of maintenance activities: they can be aimed only at identifying software defects, or only to identify vulnerabilities, or contain a common set of measures to identify both defects and vulnerabilities. A set of basic models is systematized in Table 37.1.

Table 37.1 – Characteristics of the classification for availability models for smart building I&CS

General characteristics of the model	Model specification	Conventional notions
A) Base model without maintenance	-the number of defects 0..Nd - the number of vulnerabilities 0..Nv - the number of maintenances 0	MBAS1
B) Model with common maintenance	- the number of defects 0..Nd - the number of vulnerabilities 0..Nv - the number of maintenances: unlimited during the system whole life cycle - type of maintenance: common	MBAS2.1
	- the number of defects 0..Nd - the number of vulnerabilities 0..Nv - the number of maintenances: 0..Np - type of maintenances: common	MBAS2.2
C) Model with separate	- the number of defects 0..Nd - the number of vulnerabilities	MBAS3.1

maintenance	0..Nv - the number of maintenances: unlimited during the system whole life cycle - type of service: separate	
	- the number of defects 0..Nd - the number of vulnerabilities 0..Nv - the number of maintenances by defects 0..Ndp, - the number of maintenances by vulnerabilities 0..Ndv - type of service: separate	MBAS3.2

The time intervals for conducting common and separate maintenances include the periods of testing, elimination of detected defects and vulnerabilities, and verification of the modified software. The procedures for finding defects and vulnerabilities differ both in composition and in duration, and their completeness determines the corresponding probabilities of PCS and PCR.

37.2 Models for availability of information and control systems in smart buildings taking into account reliability and safety procedures

37.2.1 Basic model of availability of BAS architecture taking into account software defects and vulnerabilities (MBAS1)

The basic model describes the processes of manifestation and elimination of software defects and vulnerabilities as separate flows of random events. The initial number of defects (Nd) and vulnerabilities (Nv) are the input parameters of the model. In addition, the input parameters are intensities of random event flows common for all Markov models. In the thesis, an example of the BAS architecture is considered, which at the time of putting into operation contains two software defects and two vulnerabilities. Fig. 37.1 shows its marked graph.

The main assumptions are those about the simplest failure and recovery flows that change the state of the system. After the manifestation of a defect (or vulnerability), the system with the probability PR (PS) stops working until they are completely eliminated. With the probability 1-PR (for defects) or 1-PS (for vulnerabilities) the system returns to the previous operable state through restart of the program. In the course of elimination, new defects and vulnerabilities are not introduced. As defects and vulnerabilities occur, they are gradually eliminated. In the particular case of BAS functioning after the defect or vulnerability manifestations, the system stops until they are completely eliminated (i.e., PR = 1 and PS = 1).

The operable states in Fig. 37.1 are shown in large circles with the number of defects and vulnerabilities in them; Inoperable states are shown in small circles without signatures. In the initial state F(Nd, Nv), the system contains 2 software defects and 2 vulnerabilities.

The manifestation of software defects on the graph is illustrated by diagonal transitions with a downward shift (weighted intensities λDi(Nd)), and vulnerabilities – by diagonal transitions with upward shift (weighted intensities λIj(Nv)). After the manifestation of vulnerabilities, they are eliminated with intensities PS*μIj, respectively; the elimination of software defects is performed with PR*μDi intensities. After all defects and vulnerabilities have been removed, the system goes to the F(0,0) state.

The software restart is illustrated by transitions from inoperable states, weighted intensities (1-PR)*μDHi and (1-PS)*μIFi.

The marked state graph and transitions (Fig. 37.2), which includes an endless numbering of states, was constructed using the modified function grPlot_marker. The Kolmogorov SDE is constructed according to the graph of MBAS1 is as follows:

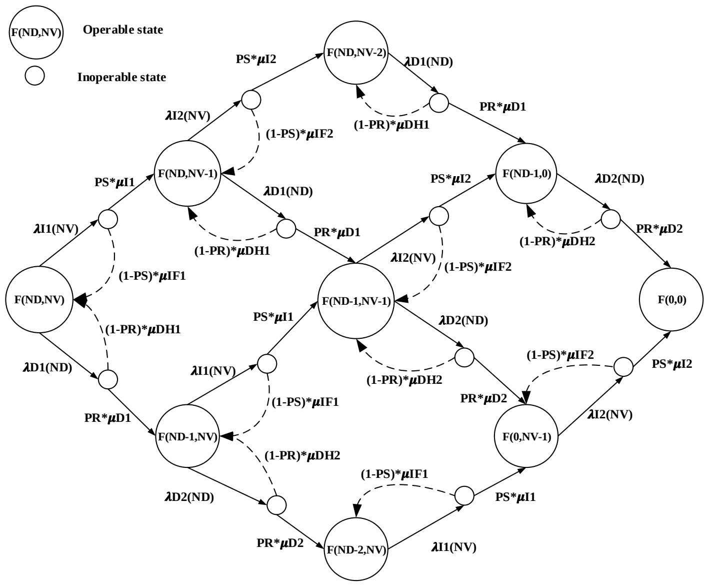


Fig. 37.1 – Marked graph of the base model MBAS1 taking into account the manifestation and elimination of software defects and vulnerabilities (without numbering of states)

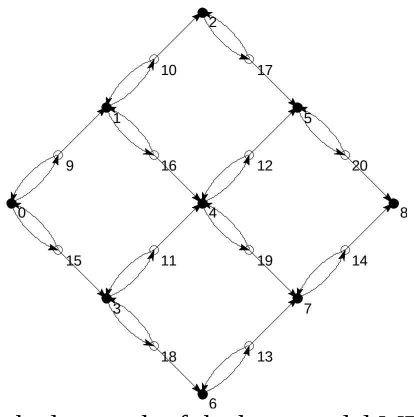


Fig. 37.2 – Marked orgraph of the base model MBAS1 with the numbering of states, built using grPlot_marker

$$
\begin{cases}
dP_0(t)/dt = -(\lambda I_1 + \lambda D_1)P_0(t) + (1-PS)\mu IF_1 P_9(t) + (1-PR)\mu DH_1 P_{15}(t), \\
dP_1(t)/dt = -(\lambda I_2 + \lambda D_1)P_1(t) + PS\mu I_1 P_9(t) + (1-PS)\mu IF_2 P_{10}(t) + \\
\qquad\qquad + (1-PR)\mu DH_1 P_{16}(t), \\
dP_2(t)/dt = -\lambda D_1 P_2(t) + PS\mu I_2 P_{10}(t) + (1-PR)\mu DH_1 P_{17}(t), \\
dP_3(t)/dt = -(\lambda I_1 + \lambda D_2)P_3(t) + (1-PS)\mu IF_1 P_{11}(t) + PR\mu D_1 P_{15}(t) + \\
\qquad\qquad + (1-PR)\mu DH_2 P_{18}(t), \\
dP_4(t)/dt = -(\lambda I_2 + \lambda D_2)P_4(t) + PS\mu I_1 P_{11}(t) + (1-PS)\mu IF_2 P_{12}(t) + \\
\qquad\qquad + PR\mu D_1 P_{16}(t) + (1-PR)\mu DH_2 P_{19}(t), \\
dP_5(t)/dt = -\lambda D_2 P_5(t) + PS\mu I_2 P_{12}(t) + PR\mu D_1 P_{17}(t) + (1-PR)\mu DH_2 P_{20}(t), \\
dP_6(t)/dt = -\lambda I_1 P_6(t) + (1-PS)\mu IF_1 P_{13}(t) + PR\mu D_2 P_{18}(t), \\
dP_7(t)/dt = -\lambda I_2 P_7(t) + PS\mu I_1 P_{13}(t) + (1-PS)\mu IF_2 P_{14}(t) + PR\mu D_2 P_{19}(t), \\
dP_8(t)/dt = PS\mu I_2 P_{14}(t) + PR\mu D_2 P_{20}(t), \\
dP_9(t)/dt = -\big((1-PS)\mu IF_1 + PS\mu I_1\big)P_9(t) + \lambda I_1 P_0(t), \\
dP_{10}(t)/dt = -\big((1-PS)\mu IF_2 + PS\mu I_2\big)P_{10}(t) + \lambda I_2 P_1(t), \\
dP_{11}(t)/dt = -\big((1-PS)\mu IF_1 + PS\mu I_1\big)P_{11}(t) + \lambda I_1 P_3(t), \\
dP_{12}(t)/dt = -\big((1-PS)\mu IF_2 + PS\mu I_2\big)P_{12}(t) + \lambda I_2 P_4(t), \\
dP_{13}(t)/dt = -\big((1-PS)\mu IF_1 + PS\mu I_1\big)P_{13}(t) + \lambda I_1 P_5(t), \\
dP_{14}(t)/dt = -\big((1-PS)\mu IF_2 + PS\mu I_2\big)P_{14}(t) + \lambda I_2 P_6(t), \\
dP_{15}(t)/dt = -\big((1-PR)\mu DH_1 + PR\mu D_1\big)P_{15}(t) + \lambda D_1 P_0(t), \\
dP_{16}(t)/dt = -\big((1-PR)\mu DH_2 + PR\mu D_2\big)P_{16}(t) + \lambda D_2 P_1(t), \\
dP_{17}(t)/dt = -\big((1-PR)\mu DH_1 + PR\mu D_1\big)P_{17}(t) + \lambda D_1 P_2(t), \\
dP_{18}(t)/dt = -\big((1-PR)\mu DH_2 + PR\mu D_2\big)P_{18}(t) + \lambda D_2 P_3(t), \\
dP_{19}(t)/dt = -\big((1-PR)\mu DH_1 + PR\mu D_1\big)P_{19}(t) + \lambda D_1 P_4(t), \\
dP_{20}(t)/dt = -\big((1-PR)\mu DH_2 + PR\mu D_2\big)P_{20}(t) + \lambda D_2 P_5(t), \\
\sum_{i=0}^{20} P_i(t) = 1;
\end{cases}
$$

$$
P_0(0) = 1; \forall i \in [1..20] \Rightarrow P_i(0) = 0. \qquad (37.1)
$$

Table 37.2 – Input parameter values of the MBAS1 model

#	Name	Mathlab-name	Time interval	Value	Measur. Unit
1.	The intensity of the first software defect manifestation λD1	laR(1)	5,45 years	5e-4	1/hour
2.	The intensity of the second software defect manifestation λD2	laR(2)	6,09 years	4.5e-4	1/hour
3.	The intensity of the first software vulnerability manifestation λI1	laS(1)	0,91 year	3e-3	1/hour
4.	The intensity of the second software vulnerability manifestation λI2	laS(2)	0,78 year	3.5e-3	1/hour
5.	The intensity of recovery with elimination of the first software defect μD1	muR(1)	2 hours	0.5	1/hour
6.	The intensity of recovery with elimination of the second software defect μD1	muR(2)	2,5 hours	0.4	1/hour
7.	The intensity of recovery with elimination of the first software vulnerability μI1	muS(1)	2,22 hours	0.45	1/hour
8.	The intensity of recovery with elimination of the second software vulnerability μI2	muS(2)	2,94 hours	0.34	1/hour
9.	The intensity of the restart without elimination of software defects μDH1= μDH2	muRH	12 minutes	5	1/hour
10.	The intensity of the restart without elimination of software vulnerabilities μIF1= μIF2	muSF	10 minutes	6	1/hour
11.	The probability of the software defect elimination during recovery	PR		0.9	

12.	The probability of the software vulnerability elimination during recovery	PS		0.9	
13.	The number of software defects in the system	Nd		2	
14.	The number of software vulnerabilities in the system	Nv		2	

To solve the SDE, the method ode15s was used in the Matlab system for the time interval of [0 ... 50000] hours. To construct the matrix of the Kolmogorov-Chapman system of differential equations, we use the matrixA function [4]. To solve the system of differential equations, the built-in solver Matlab ode15s is used. The availability function is defined as:

$$A(t) = \sum_{i=0}^{(Nd+1)\cdot(Nv+1)-1} P_i(t) \quad . \qquad (37.2)$$

The results of the simulation are shown in Fig.37.3. The graph of the model has the following character of the change in the availability function. At the first stage, the availability of the system is reduced to the minimum, and then it asymptotically tends to the established value.

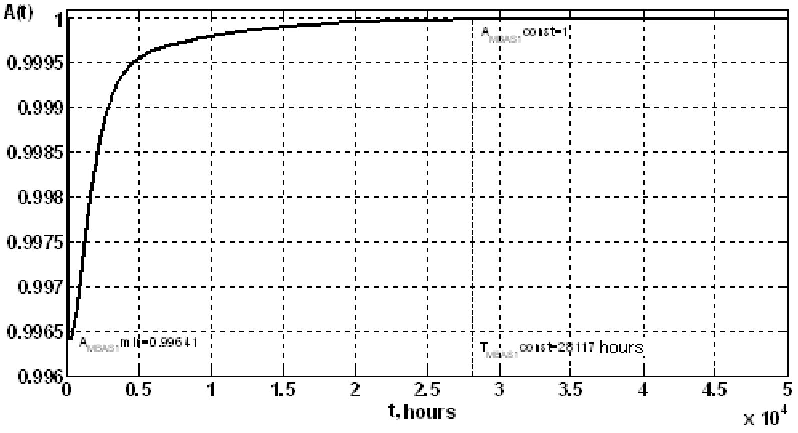


Fig. 37.3 – Results of modeling the availability of the BAS architecture (the resulting indicators are determined with the error of 10^{-5})

Thus, with further analysis of the results, it is necessary to take into account three parameters:

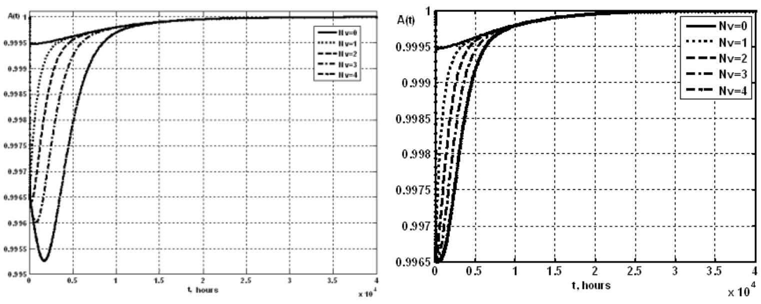
- the minimum value of the availability function $A_{MBAS\ 1}$min= 0;
- the value of the availability function in the steady state $A_{MBAS\ 1}$const= 1;
- the time interval for the transition of the availability function to the steady state$T_{MBAS\ 1}$const=28117 hours.

In a system without maintenance and provided absence of defects and vulnerabilities, availability asymptotically tends to 1. Therefore, it is of further interest to investigate the impact of individual parameters on the values of the availability function at the minimum point and the time interval for the transition of the availability function to the steady state. For the MBAS1 model, the following parameters were selected (Table 37.3):

Table 37.3 – The boundaries of the variable values of the input data of MBAS1

Name	Mathlab-name	Value row	Measuring unit
The number of software vulnerabilities in the system	Nv	[0..4]	
The probability of the software defect elimination during recovery	PR	[0..1]	
The restart intensity without elimination of software vulnerabilities	muSF	[4..10]	1/hour

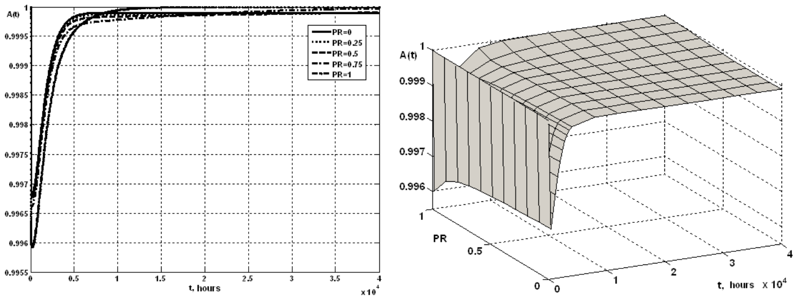
The results of modeling in the form of graphical dependencies are shown in Fig.37.4-Fig.37.6.



a) b)

Fig. 37.4 – Graphs of changing the MBAS1 availability model for different numbers of vulnerabilities Nv: (a) – with λI = var, μI = var; (b) with λI=const, μI=const

The graphs in Fig.37.4 clearly illustrate the behavior of the availability function with different number of vulnerabilities. Obviously, in a system with a large number of vulnerabilities, the latter will be eliminated with a longer time interval. But due to the presence of processes of software defect manifestation and elimination (which is illustrated by the curve with Nv = 0), the period of transition of the availability function to the steady state for systems with different number of vulnerabilities remained at the level of $T_{MBAS\ 1}$const=28117 hours. Fig.37.4 (a) illustrates the dependence of the minimum of the availability function on the parameter Nv, but this dependence is of an indirect nature, since the increase in Nv contributes to the dynamics of the parameters λI and μI. For the purity of the experiment, additional studies were carried out, during which the parameters λI and μI did not change with the increase in the number of Nv vulnerabilities. The result is shown in Fig.37.4 (b), and it is well illustrated that with the growth of Nv, the minimum of the availability function does not change ($A_{MBAS\ 1}$min= 0.9965).

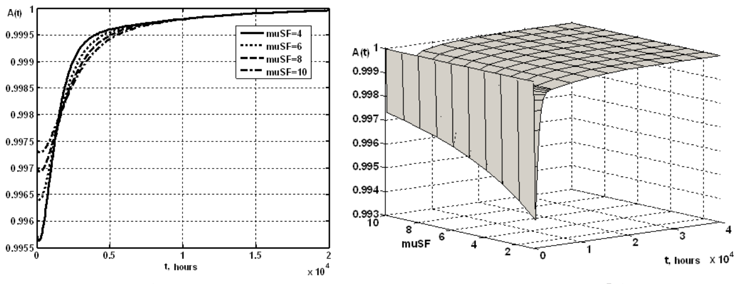


a) b)

Fig. 37.5 – Two- (a) and three-dimensional (b) graphs of the change in the availability function of the MBAS1 model for different values of the probability of eliminating the software defect during recovery

The analysis of the graphs in Fig. 37.5 (a) showed that with the growth of the parameter PR, the process of transition of the availability function to the steady state is accelerated. It is also obvious that when PR = 0, the availability function will never reach a single value (A(t)=1 under t-> ∞), since instead of eliminating the defects of the software, the system will be continuously restarted. The three-dimensional graph in Fig. 37.5 (b) gives more visualization of the availability function behavior depending on the PR parameter. The dependence of the minimum of the availability function on the PR parameter is clearly visible: at PR = 1, the value of $A_{MBAS\ 1}$min= 0.996; with a decrease of PR to zero the value of $A_{MBAS\ 1}$min asymptotically tends to $A_{MBAS\ 1}$min=0,9969.

The analysis of the graph in Fig. 37.6 (b) showed that the value of the muSF parameter (the intensity of the system restart after the manifestation of the vulnerability in the software) will depend on the minimum of the availability function, at muSF = 10 (1/hour) $A_{MBAS\ 1}$min=0.9974; and under muSF = 4 (1/hour) $A_{MBAS\ 1}$min=0.9957. This dependence is non-linear, which is well illustrated by the three-dimensional graph. The two-dimensional graphs in Fig. 37.6 (a) show that the parameter muSF does not affect the rate of transition of the availability function to the steady state. This is due to the influence of manifestation and elimination processes of software defects.



a) b)

Fig. 37.6 – Two- (a) and three-dimensional (b) graphs of the change in the availability function of the MBAS1 model at different values of the restart intensity without eliminating software vulnerabilities

37.2.2 The BAS availability model taking into account common service (MBAS2.1)

This model is an extension of the basic one and includes additional states that allow modeling of the maintenance procedures. The marked graph of the model is shown in Fig. 37.7. When constructing the graph of the model, to increase the visibility it was assumed that the defect or vulnerability was completely eliminated without restarting the system (i.e., PR = PS = 1). However, this assumption concerns only the graphic image in Fig. 37.7 (a); Fig. 37.7 (b); and the subsequent simulation results take into account the restart of the system. In addition to the assumptions listed above, the MBAS2 model assumes that during the common maintenance, it is possible to detect and eliminate one software defect or one vulnerability.

The states simulating common maintenance procedures are shown by shaded circles. The transitions to maintenance states are performed from operational states with a maintenance rate λMj. In the process of maintenance activities, the detection of a software defect occurs with the PCR probability, the detection of vulnerability – with the PCS probability. Simultaneous detection of the software vulnerability and defect occurs with the probability of PCR*PCS. The probability of PF undetectable defects and vulnerabilities complements previous events to the full group:

$$PF+PCS+PCR+PCS*PCR=1. \tag{37.3}$$

Thus, four transitions are possible from the maintenance state:

a) if a vulnerability with a PCS probability is detected, a vertical upward transition is performed, weighted by the PCS*μMs intensity, where μMs is the inverse of the mean detection time and elimination of the vulnerability [5], μMs = 1 / (TdetV + TremV);

b) in case of detection of a software defect with a PCR probability, a vertical downward transition is performed, weighted by the intensity of PCR*μMr, where μMr is the inverse value of the mean detection time and elimination of the defect [6], μMr = 1 / (TdetD + TremD);

c) in case of detection of a software defect and a vulnerability with a PCS*PCR probability, a right-hand transition weighted by the PCS*PCR*μMrs intensity is performed, where μMrs is the inverse of the mean detection and elimination time of the defect and vulnerability,

$$\mu Mrs = \frac{\mu Mr \cdot \mu Ms}{\mu Mr + \mu Ms}; \tag{37.4}$$

d) if the defect and the vulnerability are not detected with PF probability, a return to the previous working state (to the left) weighted by the intensity PF*μMtis performed, where μMt is the inverse of the average maintenance time, μMr=1/T_M.

It should be noted that in this model, we consider maintenance operations that do not anticipate the number of defects and vulnerabilities. Therefore, after removing all vulnerabilities, the transitions from the maintenance states simulating the defect detection are weighted by the parameter (1-PCR)*μMt.

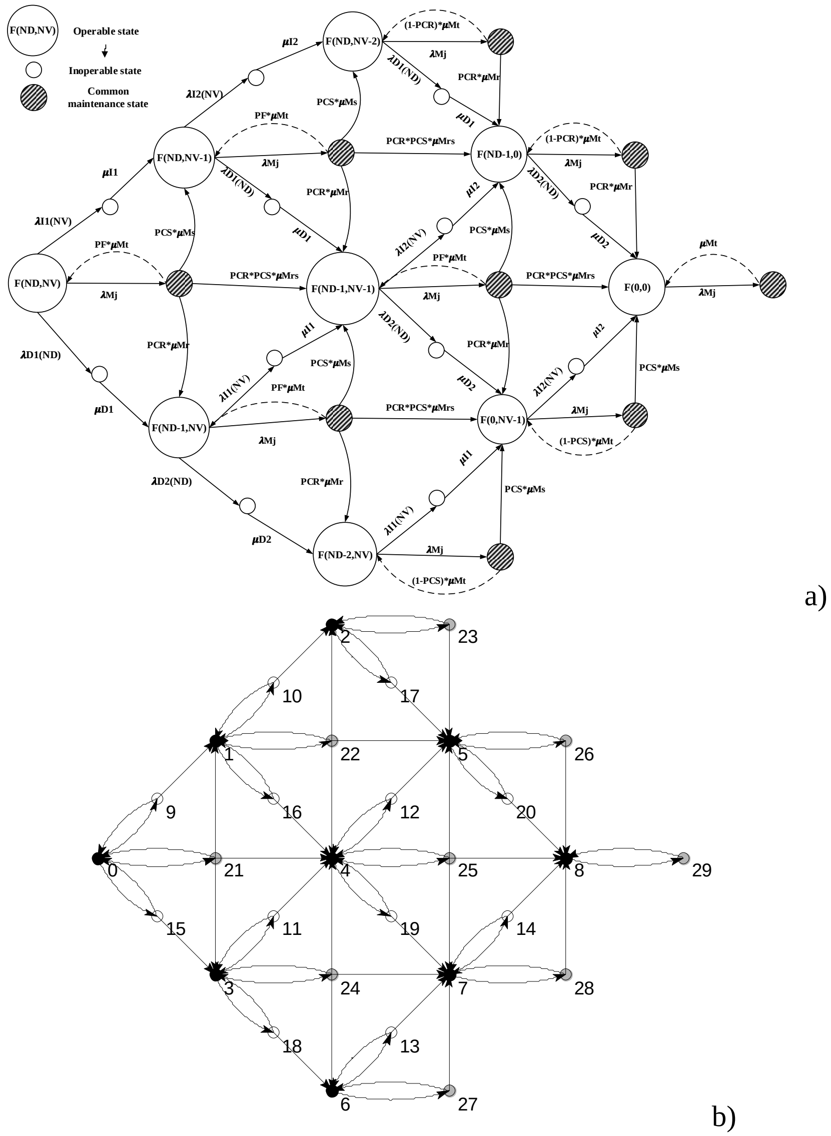


Fig. 37.7 – Marked graph of the MBAS2.1 model taking into account common maintenance (a) and the state number orgraph constructed using the function grPlot_marker (b)

Similarly, transitions simulating the detection of a vulnerability after the removal of all software defects are weighted by the parameter (1-PCS)*μMt. The extreme right state, in which maintenance of the system without defects and vulnerabilities is simulated, has, respectively, a transition weighted by the μMt parameter. The marked orgraph is presented in Fig. 37.7 (b).

To construct the matrix of the Kolmogorov-Chapman system of differential equations, we use the matrixA function [4]. The Kolmogorov SDE solution was performed in the Matlab system using the ode15s method for the time interval [0 ... 50000] hours. The availability function is determined by (37.2). The results of the solution are presented graphically in Fig. 37.8.

Table 37.4 – Values of the input parameters of the MBAS2.1 model

#	**Name**	**Mathlab-name**	**Time interval**	**Value**	**Measur. Unit**
1.	The intensity of the first software defect manifestation λD1	laR(1)	5,45 years	5e-4	1/hour
2.	The intensity of the second software defect manifestation λD2	laR(2)	6,09 years	4.5e-4	1/hour
3.	The intensity of the first software vulnerability manifestation λI1	laS(1)	0,91 year	3e-3	1/hour
4.	The intensity of the second software vulnerability manifestation λI2	laS(2)	0,78 year	3.5e-3	1/hour
5.	The intensity of recovery with elimination of the first software defect μD1	muR(1)	2 hours	0.5	1/hour
6.	The intensity of recovery with elimination of the second software defect μD1	muR(2)	2,5 hours	0.4	1/hour
7.	The intensity of recovery with elimination of the first	muS(1)	2,22 hours	0.45	1/hour

	software vulnerability μI1				
8.	The intensity of recovery with elimination of the second software vulnerability μI2	muS(2)	2,94 hours	0.34	1/hour
9.	The intensity of the restart without elimination of software defects μDH1= μDH2	muRH	12 minutes	5	1/hour
10.	The intensity of the restart without elimination of software vulnerabilities μIF1= μIF2	muSF	10 minutes	6	1/hour
11.	The probability of the software defect elimination during recovery	PR		0.9	
12.	The probability of the software vulnerability elimination during recovery	PS		0.9	
13.	The number of software defects in the system	Nd		2	
14.	The number of software vulnerabilities in the system	Nv		2	
15.	The intensity of maintenance common by vulnerabilities and defects λMj	laMj	100 hours	1e-2	1/hour
16.	The intensity of common maintenance activities μMt	muMt	2,5 hours	0.4	1/hour
17.	The intensity of detection and elimination of vulnerabilities μMs	muMs	5 hours	0.2	1/hour
18.	The intensity of detection and elimination of defects μMr	muMr	3,33 hours	0.3	1/hour
19.	The probability of vulnerability detection during maintenance procedures	PCS		0.4	
20.	The probability of software	PCR		0.2	

	defect detection during maintenance procedures				

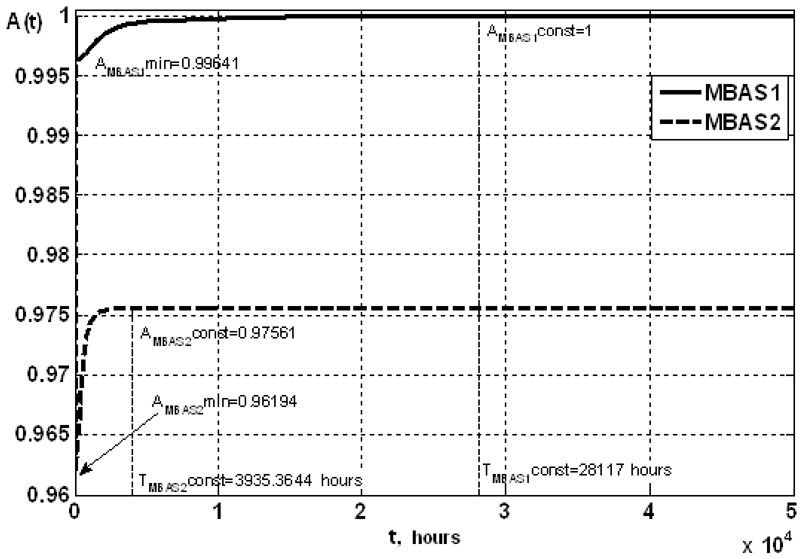


Fig. 37.8 – Graphs of the change in the BAS availability function without maintenance (MBAS1) and with the common maintenance (MBAS2.1) (the resulting indicators are determined with the error of 10^{-5})

The results of the simulation are shown in Fig. 37.8. The graphs of the models have the same nature of the change in the availability function. At the first stage, the availability of the system is reduced to the minimum, then it asymptotically tends to the established value. Thus, with further analysis of the results, it is necessary to take into account three parameters:

- the minimum value of the availability function $A_{MBAS\ i}$min(for the MBAS1 model – 0.9964, for the MBAS2.1 model – 0.96194);

- the value of the availability function in the steady state $A_{MBAS\ i}$const(for MBAS1 model – 1, for MBAS2.1 model – 0.97561);

- the time interval for the transition of the availability function to the steady state $T_{MBAS\ i}$const(for the MBAS1 model – 28117 hours, for the MBAS2.1 model – 3935.36 hours).

As can be seen from the graphs in Fig. 37.8, carrying out maintenance activities reduces both the established value of the availability function and its minimum. The MBAS2.1 model is characterized by a desire for availability to the value determined by the extreme right fragment:

$$A_{MBAS\ 2.1}const = \frac{\mu Mt}{\lambda Mj + \mu Mt}, \qquad (37.5)$$

accordingly, the input parameters λMj and μMt will affect the value of $A_{MBAS\ 2}$const.

Therefore, it is of further interest to investigate the impact of individual parameters on the values of the availability function at the minimum point and the time interval for the transition of the availability function to the steady state.

Given the constraint (37.2), in the MBAS2.1 model, the PCS and PCR parameters can simultaneously assume a maximum value of √2-1 = 0.4142. Otherwise, given the time limit for services, it is possible to "bias" both the identification of vulnerabilities and the detection of software defects. That is, with PCR = 1 -> PCS = 0 and vice versa, with PCS = 1 -> PCR = 0.

In this regard, there arises a problem of finding the optimal, from the point of view of minimizing the time for eliminating defects and vulnerabilities, distributing measures for their detection in the common maintenance cycle. Let us consider the following statement of the problem. In the system with 6 defects and 2 vulnerabilities, we need to determine the values of PCR and PCS, under which $T_{MBAS\ i}$const –>min. In this case, it is necessary to further analyze the indirect impact of parameter selection on the value of $A_{MBAS\ 2.1}$min.

To solve the problem, there is an accepted assumption about the ideality of the measures for identifying defects and vulnerabilities (PF=0), but it will be removed in the future. The values of the variable input parameters are presented in Table 37.5.

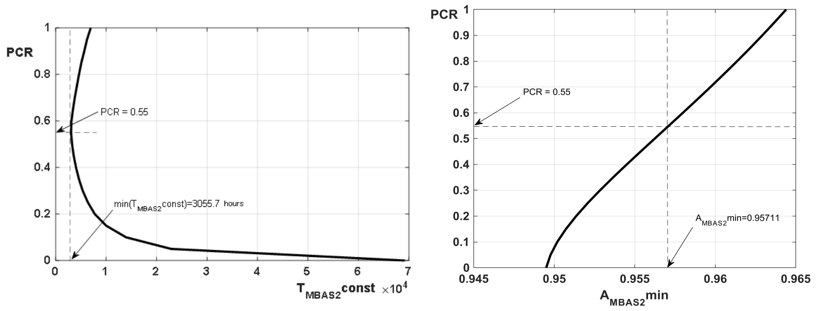
At PF = 0, the value of the PCS parameter is defined as:

$$PCS = \frac{1 - PCR}{1 + PCR}. \qquad (37.6)$$

Table 37.5 – The boundaries of the variable values of the MBAS2.1 model input data

Name	Mathlab-name	Value row
The number of software defects in the system	Nd	[0..6]
The probability of software defect detection and elimination during common maintenance	PCR	[0..1]

To investigate the impact of these parameters, special cyclic software constructs were developed. The results of modeling in the form of graphical dependencies are shown in Fig. 37.9.



a) b)

Fig. 37.9 – Graphs of the dependence of the resulting parameters $T_{MBAS\ 2.1}$const (a) and $A_{MBAS\ 2.1}$min (b) model with the common maintenance (MBAS2.1) on the input PCR parameter

The simulation results showed that the minimum achievable time $T_{MBAS\ 2.1}$const = 3055.7 hours is achieved with the PCR value of 0.55 (in addition, another parameter is PCS = 0.29). However, it should be taken into account that the value of the second result parameter $A_{MBAS\ 2.1}$min=0.95711 is in the middle of the curve in Fig.37.9, b, i.e., the minimization is performed only by the parameter $T_{MBAS\ i}$const.

Based on the studies carried out, the values of the PCR input parameter depend on the initial number of defects under the condition of $T_{MBAS\ i}$const –>min.

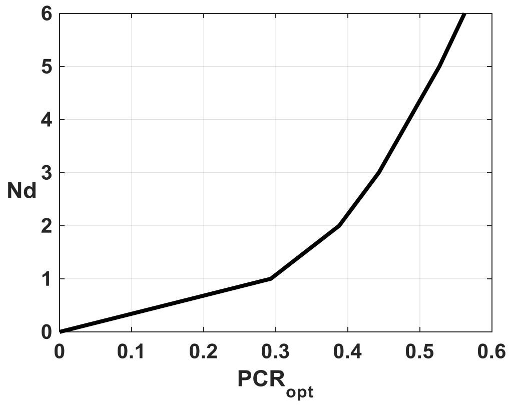


Fig. 37.10 – Graph of the dependence between the optimal PCR parameter (according to the $T_{MBAS\ i}$const –>min criterion) of the common maintenance model (MBAS2.1) and the initial number of defects in the Nd system

The values of PCRopt are tabulated and are presented in Table 37.5. Fig. 37.11 shows the dependence of PCRopt on the input parameters Nd and Nv in three-dimensional space.

Table 37.5 – Tabulated PCRopt values

Nd / Nv	0	1	2	3	4	5	6
0	0	1	1	1	1	1	1
1	0	0,357	0,481	0,544	0,585	0,629	0,677
2	0	0,293	0,388	0,443	0,485	0,527	0,562
3	0	0,254	0,320	0,365	0,436	0,466	0,489
4	0	0,214	0,280	0,329	0,380	0,412	0,430
5	0	0,190	0,246	0,292	0,329	0,360	0,406
6	0	0,167	0,224	0,263	0,308	0,340	0,361

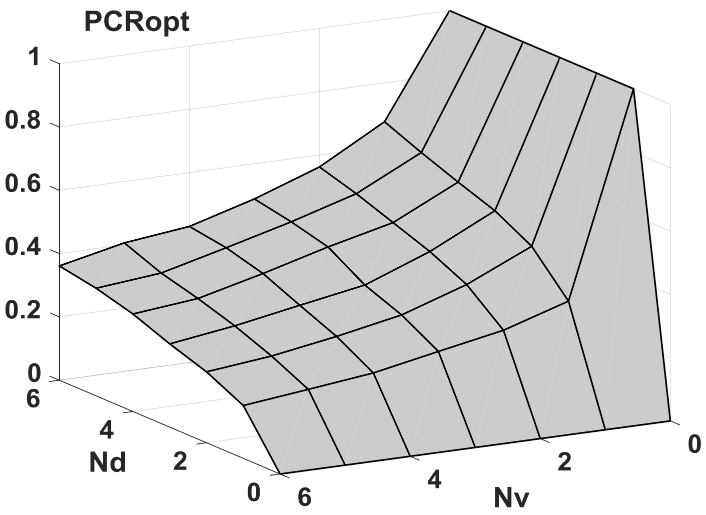
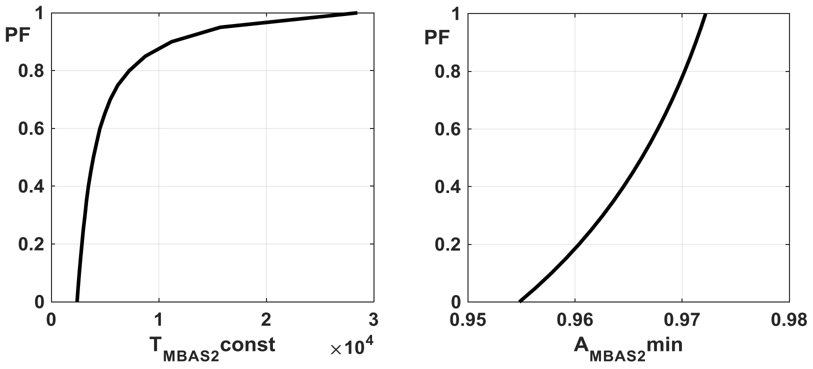


Fig. 37.11 – Three-dimensional graph for the dependence of the optimal PCR parameter (according to the $T_{MBAS\ i}$const –>min criterion) in the common maintenance model (MBAS2.1) on the initial number of Nd defects and the Nv vulnerabilities in the system

We will further consider the impact of the PF parameter on the values of $A_{MBAS\ 2.1}$min and $T_{MBAS\ i}$const. In the process of condition fulfillment, the assumption is made about the uniformity of efforts aimed at identifying defects and vulnerabilities in the common maintenance process (PCR = PCS). Under such condition, the probability of undetectability of defects and vulnerabilities in the maintenance process varies from 0 (at PCR = PCS) to 1 (at PCR = PCS = 0).



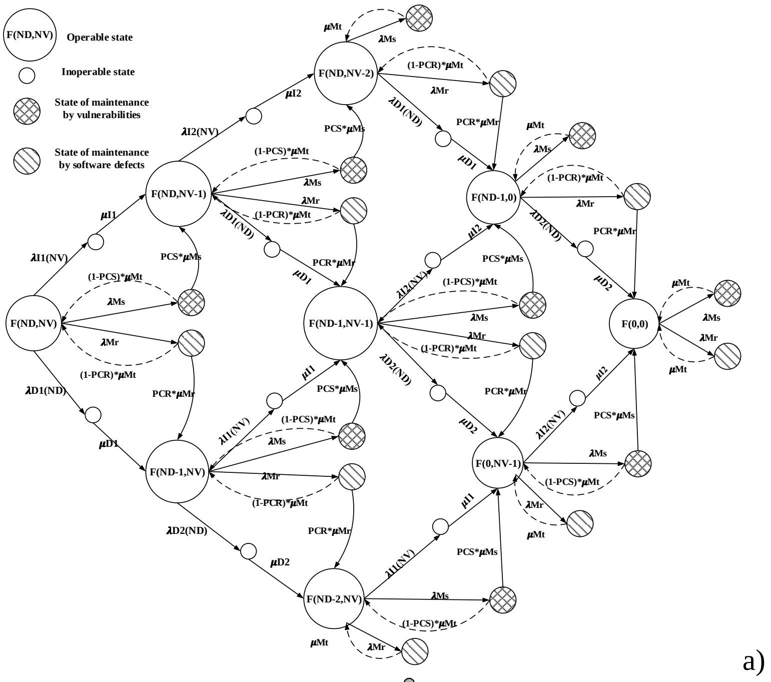
a) b)

Fig. 37.12 – The graph for the dependence of the resulting parameters $T_{MBAS\ 2.1}$const (a) and $A_{MBAS\ 2.1}$min(b) of the model with common maintenance (MBAS2.1) on the input PF parameter

The simulation results (Fig. 37.12) illustrate the fact that the undetection of vulnerabilities and defects in the course of common maintenance delay the time of their elimination (the resulting parameter $T_{MBAS\ 2}$const increases with the probability PF to 1). In this case, the value of the resulting indicator $A_{MBAS\ 2.1}$min improves due to the fact that the common maintenance procedures without eliminating defects and vulnerabilities are shorter (muMt>muMs, muMt>muMr and muMt>muMrs).

37.2.3 The BAS availability model taking into account separate maintenance (MBAS3.1)

The model is also extended with respect to the basic MBAS1 and includes additional states of the separate maintenance procedures. Unlike the previous model, MBAS2.1, the number of maintenance states is doubled, since we consider maintenance procedures, the purpose of which is to identify only software defects, and vice versa, only vulnerabilities. The marked graph of the model is shown in Fig.37.13.



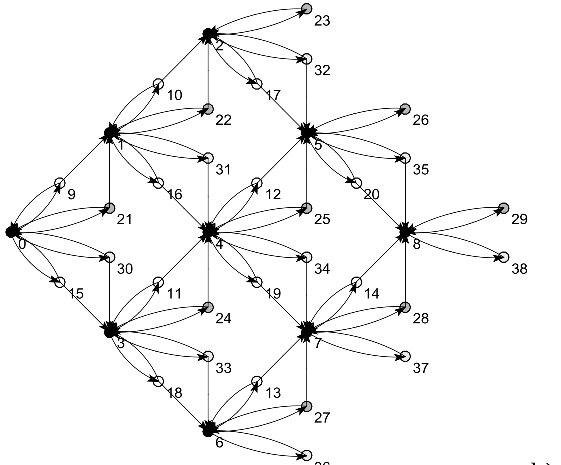


Fig. 37.13 – Marked graph of the MBAS3.1 model taking into account the separate maintenance (a) and the orgraph with the numbering of states built using grPlot_marker (b)

When constructing the graph of the model, to increase the visibility it was assumed that the defect or vulnerability was completely eliminated without restarting the system (i.e., PR = PS = 1). But this assumption concerns only the graphic representation in Fig. 37.13 (a), Fig. 37.13 (b) and the subsequent simulation results take into account the restart of the system.

The states that simulate separate maintenance procedures are shown by circles with different strokes. Transitions to maintenance states are performed from operable states: to vulnerability maintenance states – with the maintenance intensity λMs; to maintenance states for software defects – with the intensity λMr. Since separate maintenance is considered, two complete groups of events are formed: the detection of vulnerability in the maintenance process with the probability of PCS and undetection of vulnerability with probability (1-PCS); detection of a software defect in the maintenance process with a probability of PCR and undetection a defect with probability (1-PCR).

Two transitions are performed from each maintenance state for the vulnerabilities: the first one with the intensity PCS*μMs simulates the identification and elimination of the service vulnerability; the second one with the intensity (1-PCS)*μMt simulates maintenance without revealing vulnerability. If all vulnerabilities are removed, the transition from the maintenance state is weighted by the μMt intensity. Similarly, there is a simulation of transitions from maintenance states to software defects. Transitions with the intensity of PCR*μMr simulate the identification and elimination of a software defect in maintenance; transitions with intensity (1-PCR)*μMt simulate maintenance without detecting defects. If all defects are eliminated, the transitions from the maintenance state are weighted by the μMt intensity. The marked orgraph shown in Fig. 37.13 (b).

To construct the matrix of the Kolmogorov-Chapman system of differential equations, we use the matrixA function [4]. The Kolmogorov SDE solution was performed in the Matlab system using the ode15s method for the time interval of [0 ... 50000] hours. The availability function is determined by (37.1). The results of the solution are presented graphically in Fig. 37.14.

Table 37.6 – Values of the input parameters of the MBAS3.1 availability model

#	Name	Mathlab-name	Time interval	Value	Measur. Unit
1.	The intensity of the first software defect manifestation λD1	laR(1)	5,45 years	5e-4	1/hour
2.	The intensity of the second software defect manifestation λD2	laR(2)	6,09 years	4.5e-4	1/hour
3.	The intensity of the first software vulnerability manifestation λI1	laS(1)	0,91 year	3e-3	1/hour
4.	The intensity of the second software vulnerability manifestation λI2	laS(2)	0,78 year	3.5e-3	1/hour
5.	The intensity of recovery with elimination of the first software defect μD1	muR(1)	2 hours	0.5	1/hour
6.	The intensity of recovery with elimination of the second software defect μD1	muR(2)	2,5 hours	0.4	1/hour
7.	The intensity of recovery with elimination of the first software vulnerability μI1	muS(1)	2,22 hours	0.45	1/hour
8.	The intensity of recovery with elimination of the second software vulnerability μI2	muS(2)	2,94 hours	0.34	1/hour
9.	The intensity of the restart without elimination of software defects μDH1= μDH2	muRH	12 minutes	5	1/hour

10.	The intensity of the restart without elimination of software vulnerabilities μIF1= μIF2	muSF	10 minutes	6	1/hour
11.	The probability of the software defect elimination during recovery	PR		0.9	
12.	The probability of the software vulnerability elimination during recovery	PS		0.9	
13.	The number of software defects in the system	Nd		2	
14.	The number of software vulnerabilities in the system	Nv		2	
15.	The intensity of maintenance common by vulnerabilities and defects λMj	laMj	1000 hours	1e-3	1/hour
16.	The intensity of separate maintenance by vulnerabilities λMs	laMj	200 hours	5e-3	1/hour
17.	The intensity of separate maintenance by defects λMr	laMr	1000 hours	1e-3	1/hour
18.	The intensity of common maintenance performance μMt	muMt	2,5 hours	0.4	1/hour
19.	The intensity of detection and elimination of vulnerabilities μMs	muMs	5 hours	0.2	1/hour
20.	The intensity of detection and elimination	muMr	3,33 hours	0.3	1/hour

	of defects μMr				
21.	The probability of vulnerability detection during maintenance procedures	PCS		0.4	
22.	The probability of software defect detection during maintenance procedures	PCR		0.2	

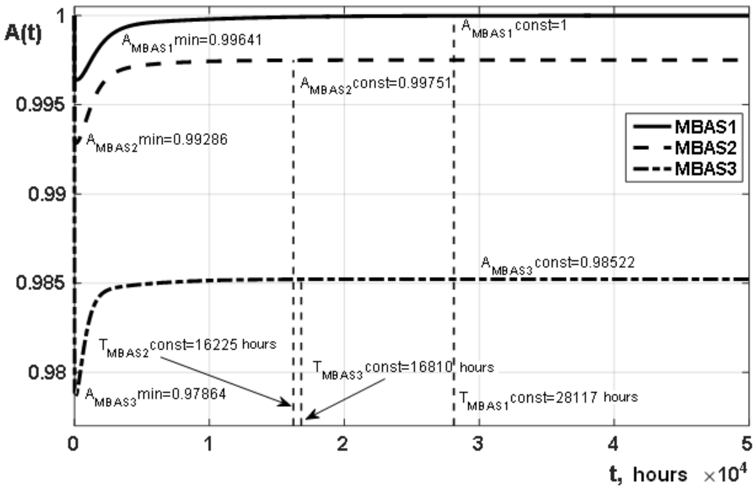


Fig. 37.14 – Graphs of the change in the availability function of the BAS without maintenance (MBAS1), with the common maintenance (MBAS2.1) and separate maintenance (MBAS3.1) (the resulting indicators are determined with the error of 10^{-5})

The simulation results are shown in Fig. 37.14. The graphs of the models have the same nature of the change in the availability function. At the first stage the availability of the system is reduced to the minimum, and then it asymptotically tends to the established value. Thus, with further analysis of the results, it is necessary to take into account three parameters:

- the minimum value of the availability function $A_{MBAS\ i}$min(for the MBAS1 model – 0.99641, for the MBAS2.1 model – 0.99286, for the MBAS3.1 model – 0.97864);

- the availability value in the steady state $A_{MBAS\ i}$const(for the MBAS1 – 1 model, for the MBAS2.1 model – 0.9975, for the MBAS3.1 model – 0.9852);

- the time interval for the transition of the availability function to the steady state $T_{MBAS\ i}$const(for the MBAS1 model – 28117 hours, for the MBAS2.1 model – 16225 hours, for the MBAS3.1 model – 16810 hours).

As can be seen from the graphs in Fig. 37.14, carrying out maintenance activities reduces both the established value of the availability function and its minimum. Due to the accepted assumptions about the gradual elimination of defects and vulnerabilities, the availability of the system without maintenance asymptotically tends to 1.

For models with maintenance, the desire of availability to the value determined by the extreme right fragment is typical, which for the separate maintenance is:

$$A_{MBAS\ 3.1}const = \frac{\mu Mt}{\lambda Mr + \lambda Ms + \mu Mt}. \qquad (37.7)$$

This can explain the gain of the model with the common maintenance by the indicators of the minimum of the availability function (by 0.0142) and the stationary value of the availability function (by 0.0123).

Carrying out the maintenance allows 1.73 times to speed up the identification and elimination of defects and vulnerabilities. In this case, the difference in $T_{MBAS\ i}$constindicators for models with common and separate maintenance is insignificant (less than 1%). But here it is necessary to take into account the fact that MBAS2.1 and MBAS3.1 models were given the same probability values for detecting PCS and PCR defects and vulnerabilities. And if in the model MBAS3.1 PCS and PCR can vary in the range of 0..1 simultaneously, then in the MBAS2.1 model the parameters PCS and PCR can simultaneously take the maximum value of 0.4142.

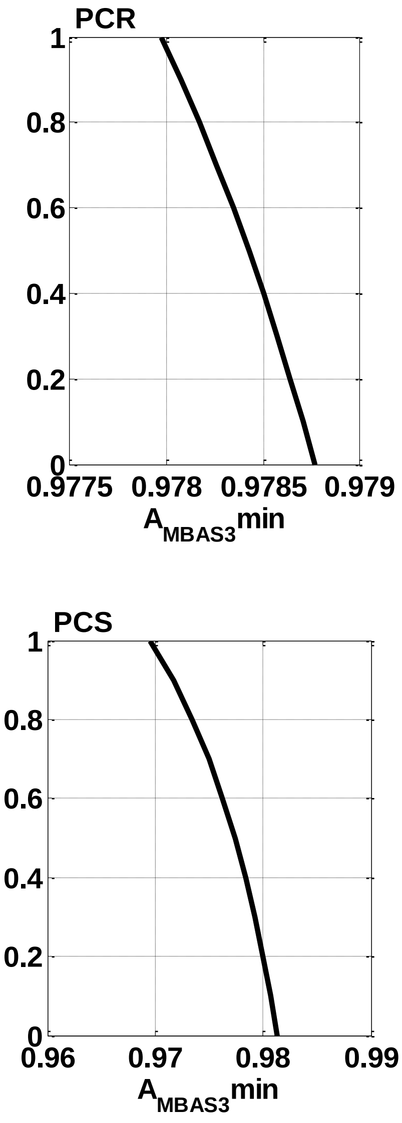
Further, we are interested in the study of the influence of individual parameters on the values of the availability function at the minimum point and the time interval for the transition of the availability function to the steady state.

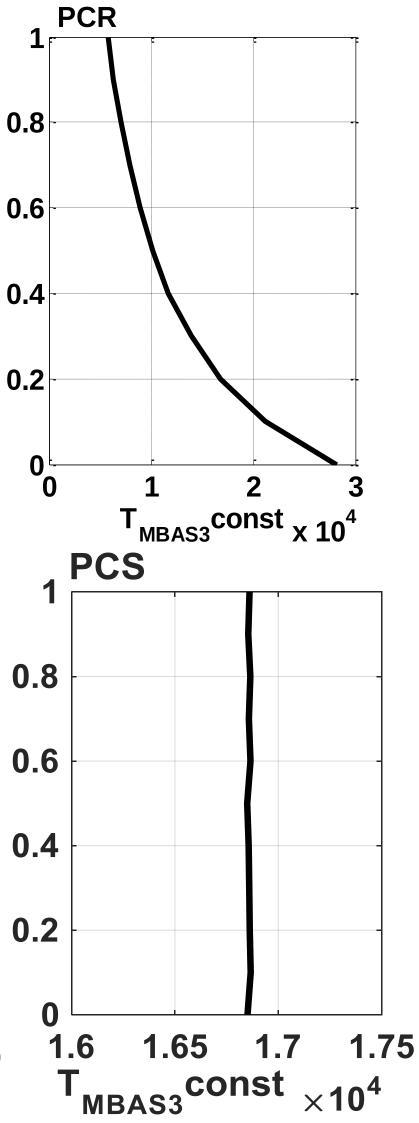
Unlike MBAS2.1, in the current model, PCS and PCR parameters can simultaneously change the value on the interval [0..1]. It is expected that with better detectability of defects and vulnerabilities (PCS = 1 and PCR = 1), there will be an acceleration of the transition of the availability function to the steady state. Then the interest is the problem of studying the impact of the PCS and PCR parameters on the minimum of the availability function of $A_{MBAS\ 3.1}$min with different number of defects and vulnerabilities. In addition, the indirect influence of the input parameters on the value of $T_{MBAS\ 3.1}$const should be further analyzed.

Table 37.7 – The boundaries of the variable values of the MBAS3.1 model input data

Name	Mathlab-name	Value row
The number of software defects in the system	Nd	[0..6]
The number of software vulnerabilities in the system	Nv	[0..6]
The probability of detection and elimination a software defect during separate maintenance	PCR	[0..1]
The probability of detection and elimination a software vulnerability during separate maintenance	PCS	[0..1]

To study the impact of these parameters, special cyclic program constructs were developed. The results of modeling in the form of graphical dependencies are shown in Fig.37.15.





a) b)

Fig. 37.15 – Graph of the dependence of the resulting parameters $A_{MBAS\ 3.1}$min (a) and $T_{MBAS\ 3.1}$const (b) of the model with separate maintenance (MBAS3) on the input PCS and PCR parameters

The analysis of the graphs in Fig. 37.15 confirms the optimality of the parameter PCR = 1 in the MBAS3 model, with the optimality being performed both by the $T_{MBAS\ 3.1}$const–>min criterion and by the $A_{MBAS\ 3.1}$min–>min criterion. At PCS = 1, the optimality is observed by the criterion $A_{MBAS\ 3.1}$min–>min.

The most interesting were the results of the studying the influence of the PCS parameter values on the resulting indicator $T_{MBAS\ 3}$const. If we look at Fig. 37.15 (d), then it seems that the $T_{MBAS\ 3.1}$const values vary randomly with the change in the PCS. However, the spread between the obtained values of $T_{MBAS\ 3.1}$const does not exceed 16 hours, which is 3.4e-5 relative to the boundaries of the investigated time interval. Therefore, in the received configuration, the values of the input PCS parameter have no impact on the $T_{MBAS\ 3.1}$const result. This is explained by the fact that the intensity of the maintenance by vulnerabilities is five times greater than the maintenance intensity by defects, therefore, for any PCS, the system will more get in states of maintenance by vulnerabilities.

Further, it is advisable to compare the models with the common and separate maintenance according to the resulting $T_{MBAS\ i}$constindicator for the optimal values of the input parameters PCS and PCR.

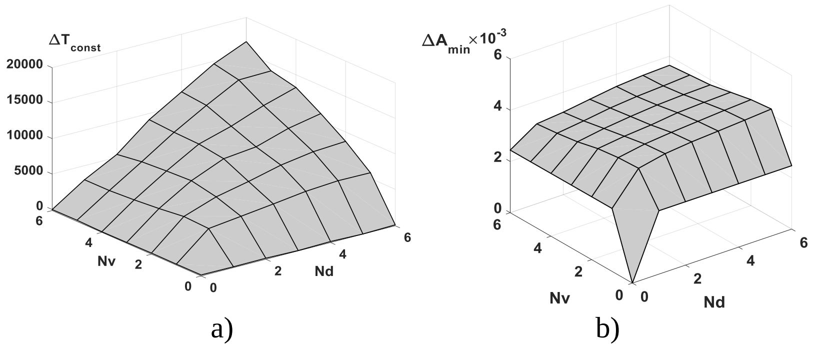


Fig. 37.16 – Dependence of the resultant difference $\Delta T_{MBAS\ i}$const(a) and $\Delta A_{MBAS\ i}$min (b) for models with separate and common service on the input parameters Nd and Nv

During the comparison, the values of the intensities of common and separate maintenance were assumed equal to λMj = λMs = λMr = 1e-3 (1/hour). To increase the visibility, the results are shown in the form of the dependence of the difference deltaTconst=$T_{MBAS\ 3.1}$const–$T_{MBAS\ 2.1}$const on the dimension of the sets of input defects and vulnerabilities (Nd and Nv).

If there are no defects (Nd = 0) or vulnerabilities (Nv = 0) at the initial moment of time or Nv=0, models with common and separate maintenance show a commensurate rate of elimination of vulnerabilities (Nd=0, Nv=[1..6]) or defects (Nd=[1..6], Nv=0): the difference between the indicators $T_{MBAS\ i}$constdoes not exceed 102 hours. This can be explained by the fact that in the model with common maintenance under such conditions the corresponding optimal parameter PCR = 1 (PCS = 1) is adopted.

However, if there are defects and vulnerabilities in the system (Nd> 0, Nv> 0), the advantage of the model with separate maintenance is evident, where defects and vulnerabilities are eliminated faster. This advantage (illustrated by the difference ΔTconst) increases with the initial number of defects and vulnerabilities. In addition, Fig.37.16 (b) illustrates the weak dependence of the difference $\Delta A_{MBAS\ i}$minon the number of defects and vulnerabilities; its dynamics does not exceed 10^{-4}.

37.2.4 BAS availability model with a limited number of common maintenances (MBAS2.2)

This model describes the functioning of the system in the context of common maintenance activities, but unlike the MBAS2.1 model, the number of such activities throughout the life cycle is limited.

The simulation reflects the following principle: at the planning stage of the maintenance procedures, developers can only assume the number of undetected defects and vulnerabilities. In addition, when planning common maintenance, it is impossible to know in advance what will be revealed: a defect, a vulnerability, or both defect and vulnerability. Therefore, it is planned to conduct a certain number of Np maintenance procedures.

Fig. 37.17 shows a marked graph of the BAS architecture with two defects and two vulnerabilities (Nd = 2, Nv = 2), in which six (Np = 6) common maintenance operations are performed. The parameter Np corresponds to the number of vertical diagonals of the rhomboid Fig. of orgraph (on which the common maintenance states are located). The logic of model functioning in this case is the following: the first maintenance (Np = 1) is carried out after the system is put into operation and its state has. Next, different paths of transitions over the states of the model are possible, therefore, the second maintenance (Np = 2) has two probable states and is carried out either after the defect is eliminated (transition from the state F(Nd-1, Nv)), or after the vulnerability is removed (transition from the state F (Nd, Nv-1)) or skipped (if during the first service both the defect and the vulnerability are eliminated). The third maintenance (Np = 3) has already three possible states (with transitions from the states F(Nd, Nv-2), F(Nd-1, Nv-1), F (Nd-2, Nv)) and also can be skipped if in the course of the second maintenance both the defect and the vulnerability have been identified and eliminated. The fourth maintenance (Np = 4) has two possible states (with transitions from the states F(Nd-1,0), F(0, Nv-1)); the fifth and sixth maintenances have one probable state (with the transition from the state F (0,0)).

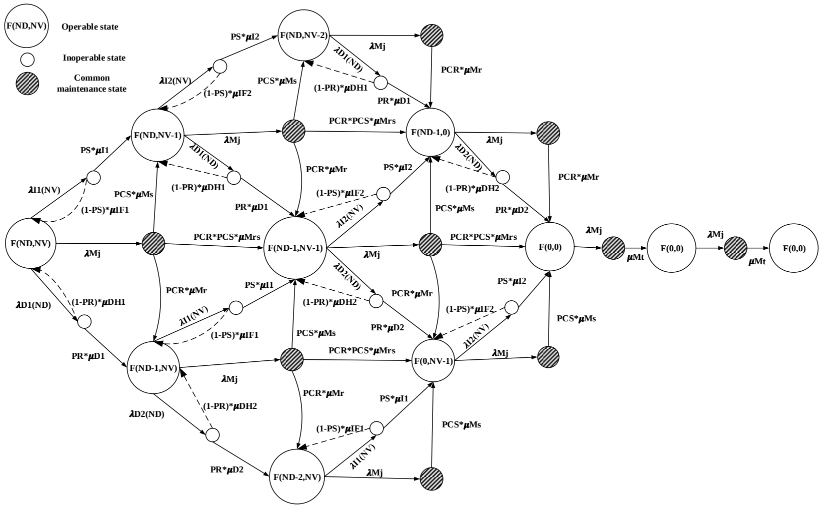


Fig. 37.17 – Marked graph of the MBAS2.2 model taking into account the limited number of common maintenances (Np = 6)

The "indicator" of the termination of common maintenance operations is the counter of their number. However, in the model, such a counter can only be used if the states of the service are passed once, i.e., under the condition of absolute effectiveness of the maintenance operations (PF = 0).

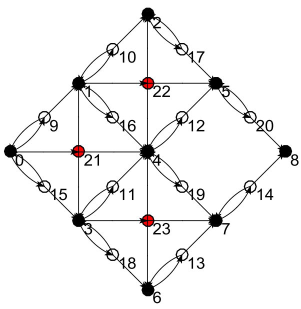
When constructing a model, it is necessary to take into account three versions of the forecasts of the number of common maintenance operations:

a) Np<Nd+Nv;

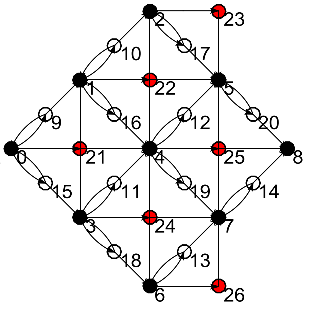
b) Np= Nd+Nv;

c) Np>Nd+Nv.

The marked orgraphs of the models constructed taking into account these variants of the forecasts are shown in Fig. 37.18. Fig. 37.18 a and b show orgraphs of the system with two defects and vulnerabilities, in which the number of scheduled maintenance operations does not exceed 4 (two for Fig. 37.18 a and three for Fig. 37.18b). Fig. 37.18c shows the orgraph of the model, in which the predicted number of maintenance operations (Np = 6) covers all the diagonals and corresponds to the actual number of defects and vulnerabilities in the system. The graph of the model shows that immediately after the elimination of all defects and vulnerabilities, the maintenance procedures are terminated.



a)



b)

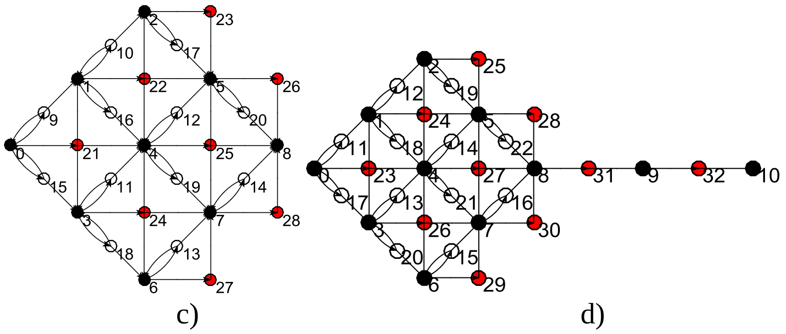


Fig. 37.18 – Marked orgraph of the MBAS2.2 model taking into account the limited number of common maintenance Np = 2 (a), Np = 3 (b), Np = 4 (c), Np = 6 (d).

The orgraph of the model MAS2.2, in which the number of maintenances (Np = 6) exceeds the real number of diagonals in the system (Nd + Nv = 4), is shown in Fig. 37.18. As it can be seen from the graph, after the elimination of all defects and vulnerabilities, the common maintenance procedures are carried out for two more periods, and then terminated. In this regard, the availability function covers additional states and is calculated as:

$$A(t)=\sum_{i=0}^{(Nd+1)*(Nv+1)+Np-(Nd+Nv)-1} P_i(t)\quad. \qquad (37.8)$$

The calculation of the availability indicators is made for the input data from Table 37.7. The values of the PCR parameters are taken from Table 37.5, the parameter PCS is determined from (37.6). To construct the matrix of the Kolmogorov-Chapman system of differential equations, we use the matrixA function [4]. The Kolmogorov SDE solution was performed in the Matlab system using the ode15s method for the time interval [0 ... 50000] hours. The availability function is determined by (37.2). The results of the solution are presented graphically in Fig. 37.19.

Table 37.7 – Values of the input parameters of the MBAS2.2 model

#	**Name**	**Mathlab-name**	**Time interval**	**Value**	**Measur. unit.**
1.	The intensity of the first software defect manifestation λD1	laR(1)	5,45 years	5e-4	1/year
2.	The intensity of the second software defect manifestation λD2	laR(2)	6,09 years	4.5e-4	1/year
3.	The intensity of the first software vulnerability manifestation λI1	laS(1)	0,91 year	3e-3	1/year
4.	The intensity of the second software vulnerability manifestation λI2	laS(2)	0,78 year	3.5e-3	1/year
5.	The intensity of recovery with elimination of the first software defect μD1	muR(1)	2 hours	0.5	1/year
6.	The intensity of recovery with elimination of the second defect μD1	muR(2)	2,5 hours	0.4	1/year
7.	The intensity of recovery with elimination of the first software vulnerability μI1	muS(1)	2,22 hours	0.45	1/year
8.	The intensity of recovery with elimination of the second software vulnerability μI2	muS(2)	2,94 hours	0.34	1/year
9.	The intensity of the restart without elimination of software defects μDH1= μDH2	muRH	12 minutes	5	1/year
10.	The intensity of the restart without elimination of software vulnerabilities μIF1= μIF2	muSF	10 minutes	6	1/year
11.	The probability of the software defect elimination during recovery	PR		0.9	

12.	The probability of the software vulnerability elimination during recovery	PS		0.9	
13.	The number of software defects in the system	Nd		2	
14.	The number of software vulnerabilities in the system	Nv		2	
15.	The intensity of maintenance common by vulnerabilities and defects λMj	laMj	100 minutes	1e-2	1/year
16.	The intensity of common maintenance procedures μMt	muMt	2,5 minutes	0.4	1/year
17.	The intensity of detection and elimination μMs	muMs	5 minutes	0.2	1/year
18.	The intensity of defect detection and elimination μMr	muMr	3,33 minutes	0.3	1/year
19.	The probability of vulnerability detection during maintenance procedures	PCS		0.4409	
20.	The probability of defect detection during maintenance procedures	PCR		0.388	
21.	Predicted number of common maintenance	Np		2	

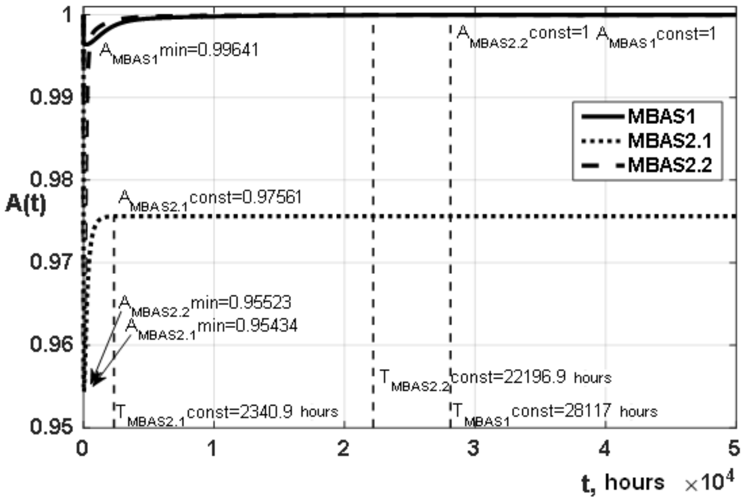


Fig. 37.19 – Graphs of the change in the availability function of the BAS architecture without maintenance (MBAS1), with the common unlimited (MBAS2.1) and limited (MBAS2.2) maintenance (the resulting indicators are determined with the error of 10^{-5})

The analysis of the graphs in Fig. 37.19 showed that the limitation of the number of maintenances in the MBAS2.2 model allows achieving the ideal availability ($A_{MBAS\ 2.2}$const=1) in the steady state. At the same time, the value of the availability minimum for models with limited and unlimited maintenance differs insignificantly (by 8.83e-4). The transition period for the availability function in the MBAS2.2 mode is 9.48 times higher than that of the MBAS2.1 model with unlimited common maintenance; however, the elimination of defects and vulnerabilities in the model with maintenance is faster than in the MBAS1 model (1.27 times).

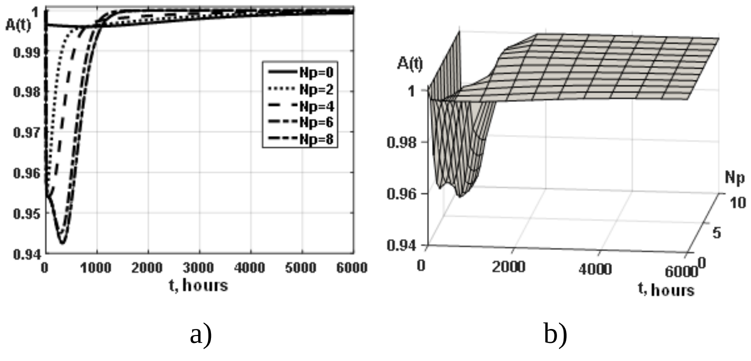
Since interest is caused by a decrease in the detection and elimination of all defects and vulnerabilities, then further we consider the influence of individual input parameters on the resulting indicator $T_{MBAS\ 2.2}$const (in addition, their impact on $A_{MBAS\ 2.2}$min is analyzed). In this case, the dimensionality of the model is increased to Nd= 3, Nv=3, the value of the PCR parameter is also taken from Table 37.5.

Table 37.8 – The boundaries of the variable values of the MBAS2.2 model input data

Name	Mathlab-name	Value row	Measur.unit
Predicted number of common maintenances	Np	[0..10]	
The intensity of maintenance common by vulnerabilities and defects λMj	laMj	[1e-2..1e-4]	1/hour

To study the impact of these parameters, special cyclic program constructs were developed. The results of simulation in the form of graphical dependencies are shown in Fig. 37.20 – Fig. 37.22.

The results of the studying the forecast accuracy impact (Np) showed the expected result. If the lack of defects and vulnerabilities is predicted (Np = 0), the MBAS2.2 model degenerates into MBAS1 (Fig. 37.20, a) and has the highest level of $A_{MBAS\ 2.2}$min (Fig. 37.20, c). With the growth in the number of limited Np maintenances up to Np = 6, the process of identifying and eliminating defects and vulnerabilities as a whole is accelerating. In this case, the graph of the change of $T_{MBAS\ 2.2}$const in Fig. 37.20, d has a specific appearance of a broken curve: up to the limit $Np \leq Nv + Nd$, it shows a decrease in the resultant index and for $Np > Nv + Nd$, the value of $T_{MBAS\ 2.2}$const increases with Np (as unsuccessful maintenance procedures are accumulated). A noticeable explanation in the behavior of $A_{MBAS\ 2.2}$min(Np) at Np = 5 is given by the fact that with such a number of maintenances the "availability" is provided from the maintenance state of the extreme right operable state S15 (Fig.37.21, a). In this case, in Fig. 37.20, a, it is clear that with the appearance of excessive maintenances (Np = 6, Np = 8), the minimum of the availability function shifts along the time axis to the right.



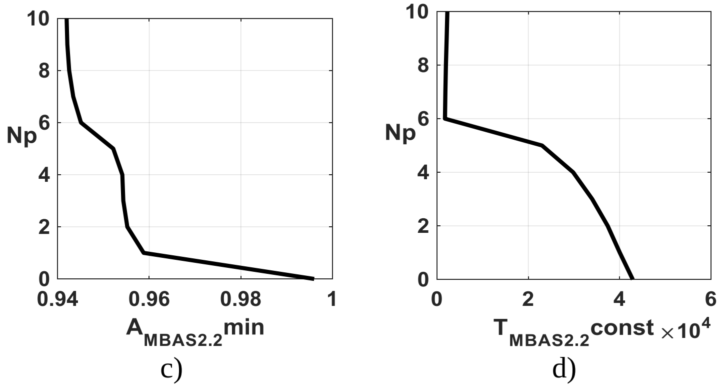
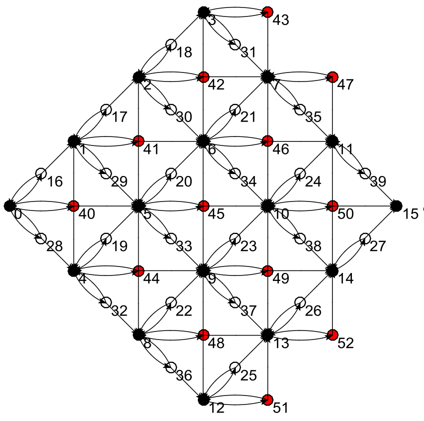


Fig. 37.20 – Graphs of the change in the resulting indicators of the MBAS2.2 model (a, b – availability functions, c – minimum availability function, d – transition period to the steady state with the error of 10^{-5}) with a limited number of common maintenances Np

In the course of the study, it was determined that the minimum resulting indicators of $T_{MBAS\ 2.2}$const are achieved with a forecast of Np = 6, the marked graph for this forecast is shown in Fig. 37.21, b.


a)

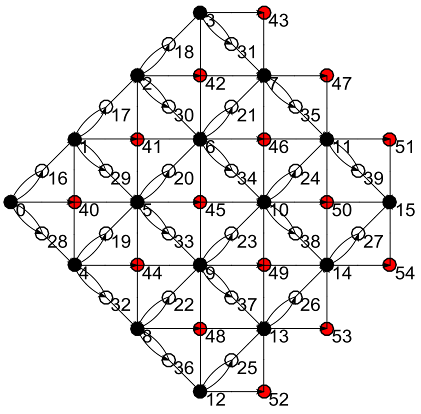
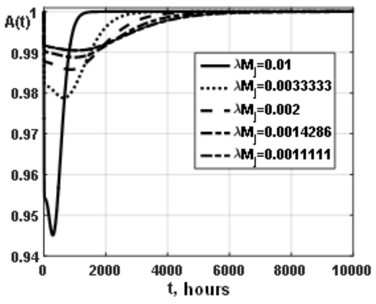
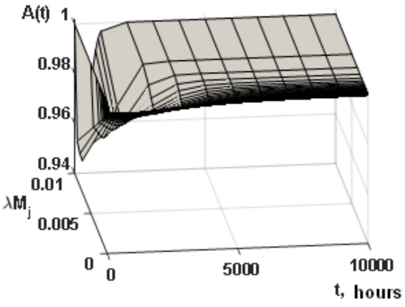

b)

Fig. 37.21. –Orgraphof the BAS architecture, Np = 5 (a) and optimal according to $T_{MBAS\ 2.2}$const–>min criterion of the BAS architecture, Np = 6 (b)

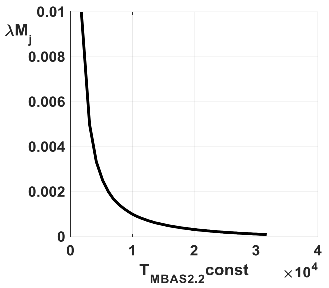
Further, the impact of maintenance intensity, common by the vulnerabilities and defects λMj, on the resulting parameters of $T_{MBAS\ 2.2}$const and $A_{MBAS\ 2.2}$min, is considered. When constructing models, the values of the input parameters Nv = Nd = 3, Np = 6 were adopted.

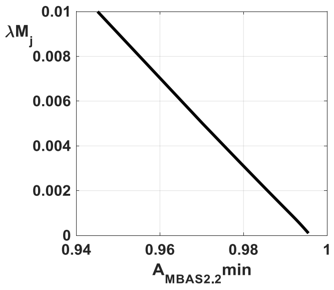


a)



b)





c) d)

Fig. 37.22 – Graphs of the change in the resulting indicators of the MBAS2.2 model (a, b – availability functions, c – minimum availability function, d – transition period to the steady state with the error of 10^{-5}) from the maintenance intensity λMj

The results given in Fig. 37.22 also show the expected result: the more frequent the maintenance procedures are, the faster the defects and vulnerabilities will be identified and corrected. The value of the resulting indicator $A_{MBAS\ 2.2}$min decreases linearly.

37.2.5 The BAS availability model taking into account the limited number of separate maintenance (MBAS3.2)

This model describes system functioning in the context of separate maintenance activities, but unlike the MBAS3.1 model, the number of such activities throughout the life cycle is limited.

Simulation shows the same principle as in the MBAS2.2 model: at the planning stage of the maintenance procedures, developers can only assume the number of undetected defects and vulnerabilities. But unlike the common maintenance model, the MBAS3.2 model knows for sure that only vulnerabilities will be fixed during the maintenance of vulnerabilities, and only defects will be eliminated during defect maintenance. Therefore, in the MBAS3.2 model, the Ndp and Nvp input parameters determine the planned number of maintenances for defects and vulnerabilities, respectively.

The marked graph of the model is shown in Fig. 37.23. When constructing the graph of the model to increase the visibility, it was assumed that the defect or vulnerability was completely eliminated without restarting the system (i.e., PR = PS = 1). But this assumption concerns only the graphic representation in Fig. 37.23; subsequent simulation results take into account the restart of the system.

The graph in Fig. 37.23 is the BAS model with two defects and two vulnerabilities (Nd = 2, Nv = 2), and it additionally describes three maintenances by defects (Ndp = 3) and one maintenance by vulnerability (Nvp = 1). Unlike the MBAS2.2 model, the planned number of maintenances (for example, over defects) determines not the number of vertical diagonals of the rhomboid Fig. of the orgraph, but corresponds to inclined lines in the direction of the shift when eliminating defects (right-down). In detecting and eliminating defects, the logic of the functioning of the MBAS3.2 model is the following: the first maintenance (Ndp = 1) is performed after the system is put into operation and has three probable states (with transitions from the states F(Nd, Nv), F(Nd, Nv-1)), F(Nd, Nv-2)). After maintenance, the detected defect is eliminated, therefore, the second maintenance (Ndp=2) also has three probable states (with transitions from the states F(Nd-1, Nv), F(Nd-1, Nv-1), F(Nd-1, 0)). Since only two defects were initially present in the system, the third maintenance by defects is redundant and an additional fragment is required for its modeling in the graph (it is shown by a dashed Fig. line). The third maintenance also has three probable states.

Since only one maintenance is planned for the vulnerabilities, it will have four probable states with transitions from the states F(Nd, Nv), F(Nd-1, Nv), F(Nd-2, Nv), F(Nd-2, Nv)'. The second vulnerability will be eliminated only after its manifestation.

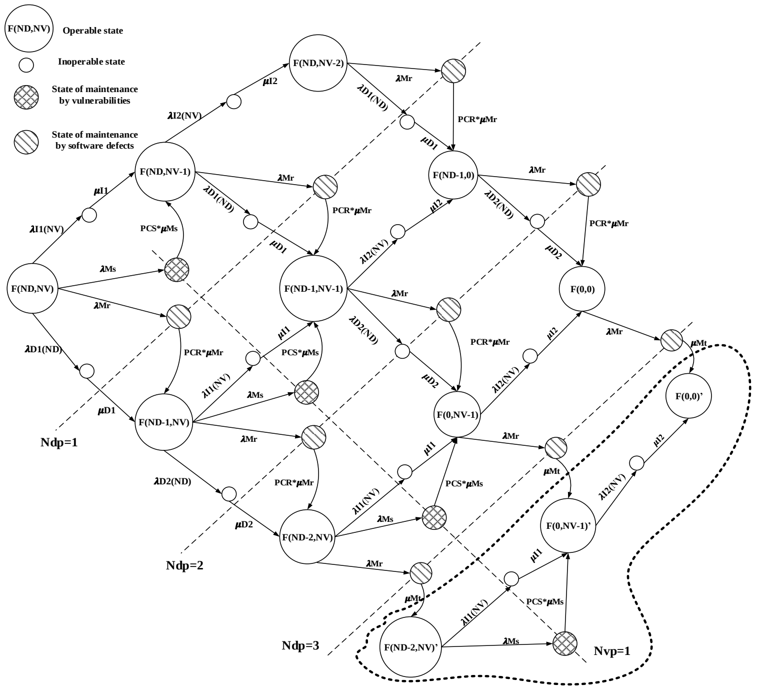


Fig. 37.23 – Marked graph of the MBAS3.2 model taking into account the limited number of separate maintenances by defects (Ndp = 3) and vulnerabilities (Nvp = 1)

When building the model, it is necessary to take into account four variants of the forecasting the initial number of defects and vulnerabilities:

a) (Ndp≤Nd)&(Nvp≤Nv)

b) (Ndp≤Nd)&(Nvp>Nv);

c) (Ndp>Nd)&(Nvp≤Nv);

d) (Ndp>Nd)&(Nvp>Nv).

The marked orgraphs of models constructed with these forecast options are shown in Fig. 37.24. Fig. 37.24, a shows the orgraph of the system with two defects and vulnerabilities, in which the number of maintenances by defects/vulnerabilities does not exceed 2 (two by vulnerabilities and one by defects). To improve the visibility of the state of maintenance over defects are shown in yellow circles, over vulnerabilities – in green. Fig. 37.24, b shows the orgraph of the model, in which the predicted number of maintenance by vulnerabilities exceeds their number in the system. This causes the occurrence of additional operable (S3, S7, S11, S15) and inoperable (S27, S31, S35, S51) states.

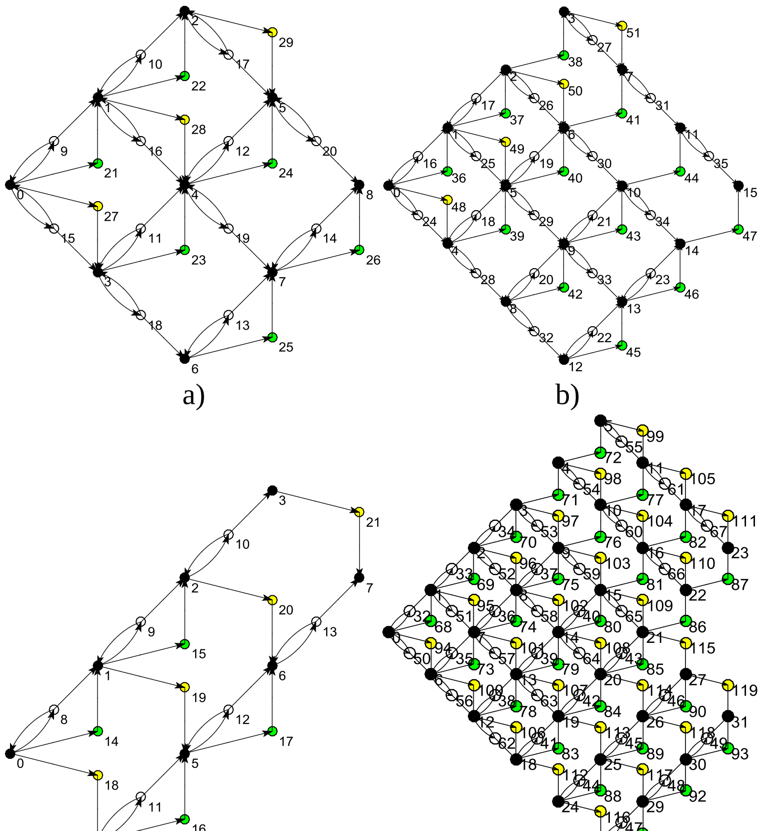


Fig. 37.24 – Marked orgraph of MBAS3.2 model taking into account the limited number of separate maintenances for configurations:
a) Nd=2, Nv=2, Ndp=1, Nvp=2; б) Nd=3, Nv=2, Ndp=1, Nvp=3;
b) Nd=0, Nv=3, Ndp=1, Nvp=2; г) Nd=3, Nv=3, Ndp=5, Nvp=5.

Fig. 37.24, c shows the orgraph of the model, in which defects are absent, but one maintenance is planned to be according to defects. This causes the occurrence of additional operable (S4, S5, S6, S7) and inoperable (S11, S12, S13, S16, S17) states. The orgraph of the MBAS3.2 model, in which the number of planned maintenances by both defects and vulnerabilities (Ndp = 5, Nvp = 5) exceeds their real number in the system (Nd = Nv = 3) and is shown in Fig.37.24. As can be seen from the graph, after the elimination of all defects and vulnerabilities, the maintenance procedures are carried out for two more periods, and then terminated. In this regard, the availability function covers additional states and is calculated as:

$$A(t)=\sum_{i=0}^{N}P_i(t), \qquad (37.9)$$

$$N = (\mathrm{Nd}+1)\cdot(\mathrm{Nv}+1)+(\mathrm{Nd}+1)\times$$
$$\times(\max(\mathrm{Nvp},\mathrm{Nv})-\mathrm{Nv})+(\mathrm{Nv}+1)\times\cdot$$
$$\times(\max(\mathrm{Ndp},\mathrm{Nd})-\mathrm{Nd})$$

The calculation of the availability indicators is performed for the input data from Table 37.9. For comparison with the MBAS2.2 model, the latter model has the PCR taken from Table 37.5; the PCS parameter is determined by (37.6). To construct the matrix of the Kolmogorov-Chapman system of differential equations, we use the matrixA function [4]. The Kolmogorov CDS solution was performed in the Matlab system using the ode15s method for the time interval of [0 ... 50000] hours. The availability function is determined by (37.9). The results of the solution are presented in the graphical form in Fig. 37.25.

Table 37.9 – Values of the input parameters of the MBAS3.2 model

#	Name	Mathlab-name	Time interval	Value	Measur. unit
1.	The intensity of the first software defect manifestation λD1	laR(1)	5,45 years	5e-4	1/year
2.	The intensity of the second software defect manifestation	laR(2)	6,09 years	4.5e-4	1/year

	λD2				
3.	The intensity of the first software vulnerability manifestationλI1	laS(1)	0,91 years	3e-3	1/year
4.	The intensity of the second software vulnerability manifestation λI2	laS(2)	0,78 years	3.5e-3	1/year
5.	The intensity of recovery with elimination of the first software defect μD1	muR(1)	2 hours	0.5	1/year
6.	The intensity of recovery with elimination of the second software defect μD1	muR(2)	2,5 hours	0.4	1/year
7.	The intensity of recovery with elimination of the first software vulnerability μI1	muS(1)	2,22 hours	0.45	1/year
8.	The intensity of recovery with elimination of the second software vulnerability μI2	muS(2)	2,94 hours	0.34	1/year
9.	The intensity of the restart without elimination of software defects μDH1= μDH2	muRH	12 minutes	5	1/year
10.	The intensity of the restart without elimination of software vulnerabilitiesμIF1= μIF2	muSF	10 minutes	6	1/year
11.	The probability of the software defect elimination during recovery	PR		0.9	
12.	The probability of the software vulnerability elimination during recovery	PS		0.9	
13.	The number of software defects in the system	Nd		2	
14.	The number of software vulnerabilities in the system	Nv		2	
15.	The intensity of maintenance common by vulnerabilities and	laMj	1000 hours	1e-3	1/year

	defects λMj				
16.	The intensity of separate maintenance by vulnerabilities λMs	laMs	200 hours	5e-3	1/year
17.	The intensity of separate maintenance by defects λMr	laMr	1000 hours	1e-3	1/year
18.	The intensity of common maintenance performance μMt	muMt	2,5 hours	0.4	1/year
19.	The intensity of detection and elimination of vulnerabilities μMs	muMs	5 hours	0.2	1/year
20.	The intensity of defectdetection and elimination μMr	muMr	3,33 hours	0.3	1/year
21.	The probability of vulnerability detection during maintenance procedures in the MBAS3.2 model	PCS		1	
22.	The probability of software defect detection during maintenance procedures in the MBAS3.2 model	PCR		1	
23.	The probability of vulnerability detection during maintenance procedures in the MBAS2.2 model	PCS		0.4409	
24.	The probability of software defect detection during maintenance procedures in the MBAS2.2 model	PCR		0.388	
25.	Predicted number of common maintenances in the MBAS3.2 model	Nvp		2	
26.	Predicted number of common maintenances in the MBAS3.2 model	Ndp		2	
27.	Predicted number of common	Np		4	

	maintenances in the MBAS2.2 model				

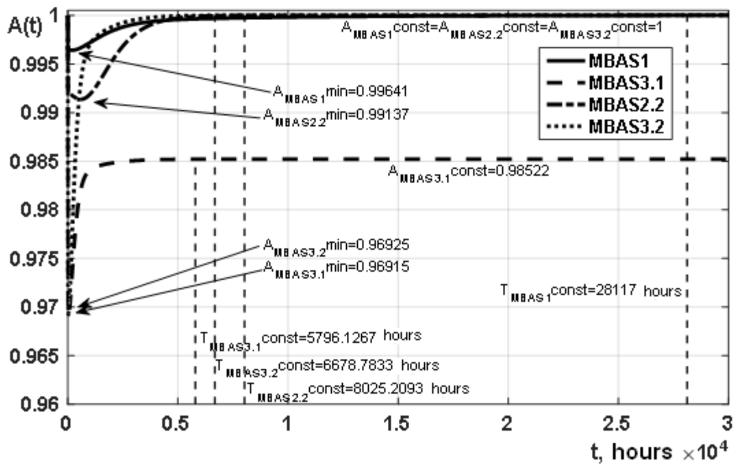


Fig. 37.25 – Graphs of change in the availability function of the BAS architecture without maintenance (MBAS1); with separate unlimited (MBAS3.1), common (MBAS2.2) and separate limited (MBAS3.2) maintenance (the resulting indicators are determined with the error of 10^{-5})

The analysis of the graphs in Fig. 37.25 showed that limiting the number of separate maintenances in the MBAS3.2 model (as in the MBAS2.2 model) allows achieving an ideal availability ($A_{MBAS\,3.2}$const=1) in the steady. Also as in the previous MBAS2.2 model, the minimum availability value for models with limited and unlimited maintenance differs insignificantly (by 9.73e-5). However, common maintenance remains an advantageous one according to the $A_{MBAS\,i}$min (by 0.022) indicator.

If we compare models with limited and unlimited maintenance, then it is clear that the latter (MBAS2.1 in Fig. 37.19 and MBAS3.1 in Fig. 37.25) has a shorter period of transition of the availability function to the steady state. The difference between the resulting $T_{MBAS\,i}$const indicators of models MBAS3.1 and MBAS3.2 is 882.6 hours. The transition period for the availability function to the steady state in the

MBAS3.2 model is 1346.4 hours less than in the limited common maintenance MBAS2.2. In addition, eliminating defects and vulnerabilities in the model with maintenance is faster than in the MBAS1 model (4.2 times).

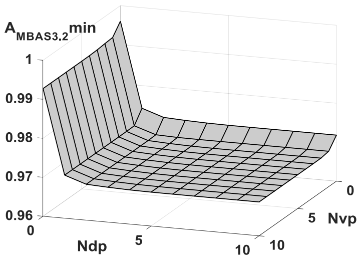
Since interest is caused by a decrease in the detection and elimination of all defects and vulnerabilities, then further we consider the influence of individual input parameters on the resulting indicator $T_{MBAS\ 3.2}$const (in addition, their impact on $A_{MBAS\ 2.2}$min is analyzed). The dimensionality of the model is increased to Nd = 3, Nv = 3.

Table 37.10 – The boundaries of the MBAS3.2 model input values

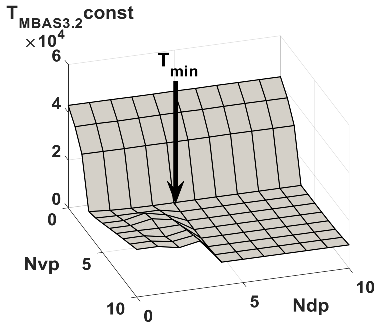
Name	Mathlab-name	Value row	Measur.unit
Predicted number of separate maintenances	Ndp, Nvp	[0..10]	
The intensity of defect detection and elimination μMr	muMr	[0.1..1]	1/hour

The results of modeling in the form of graphical dependencies are shown in Fig. 37.26 – Fig. 37.27.

Dependence of the resulting indicator $A_{MBAS\ 3.2}$min on the number of separate maintenances is shown in Fig. 37.26, a. Analysis of the three-dimensional graph allows to distinguish the following points. The BAS system without maintenance is optimal according to the criterion $A_{MBAS\ 3.2}$min–>max (Ndp=Nvp=0, $A_{MBAS\ 3.2}$min=0,996). The system without maintenance by defects (Ndp = 0, Nvp> 0) exceeds the system without maintenance by vulnerabilities (Nvp = 0, Ndp> 0) by $A_{MBAS\ 3.2}$min by 0.021. In BAS systems with the number of limited separate maintenances greater than the real number of defects and vulnerabilities (Ndp> 3, Nvp> 3), the change in $A_{MBAS\ 3.2}$min does not exceed 6.3e-8.



a)



b)

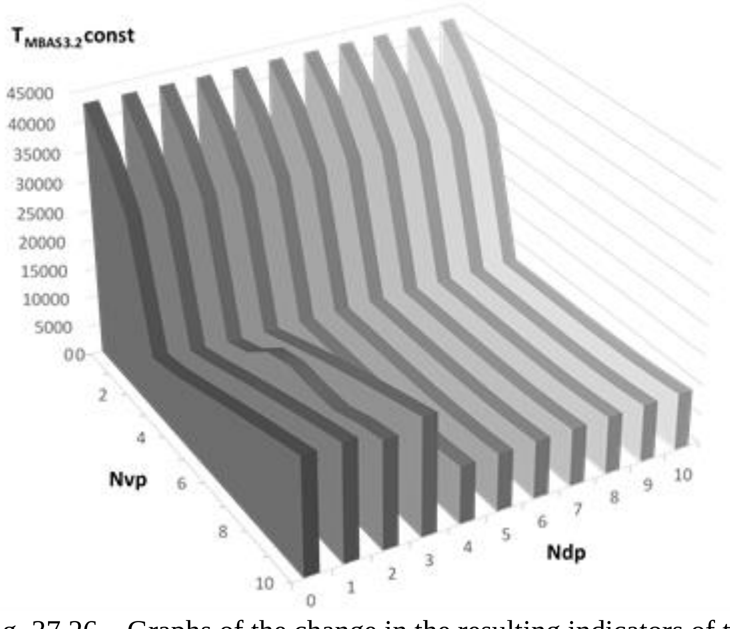


Fig. 37.26 – Graphs of the change in the resulting indicators of the MBAS3.2 model (a – the minimum of the availability function, b – the period of transition to the steady state with the error of 10^{-5}) with a limited number of separate maintenances

Fig. 37.26b shows the dependence of the transition period of the MBAS3.2 availability function in the steady state on the number of separate maintenances. The location of the minimum on the three-dimensional graph is shown by a special metrics and corresponds to the value min($T_{MBAS\ 3.2}$const)=8496,153 hours under the configuration of the number of maintenances Nvp = 3, Ndp = 4. In BAS systems with the number of limited separate maintenances greater than the actual number of defects and vulnerabilities (Ndp> 3, Nvp> 3), the change in the $T_{MBAS\ 3.2}$const does not exceed 1256.546489 hours, but there is a growing trend of $T_{MBAS\ 3.2}$const with an increase in Nvp, which is shown in Fig. 37.27.

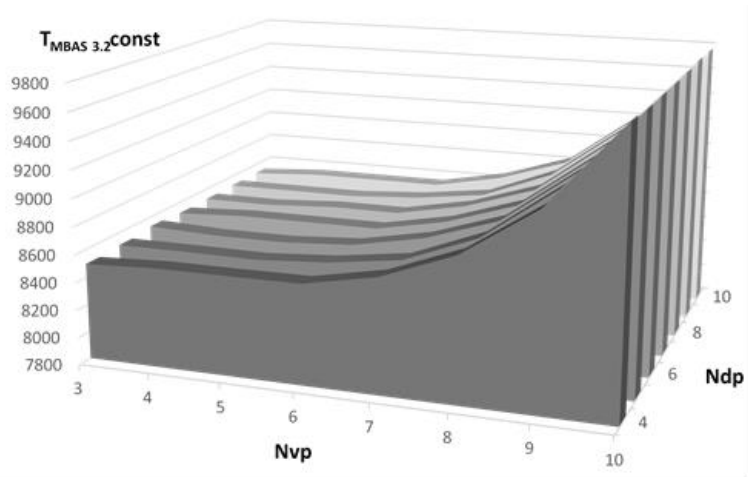
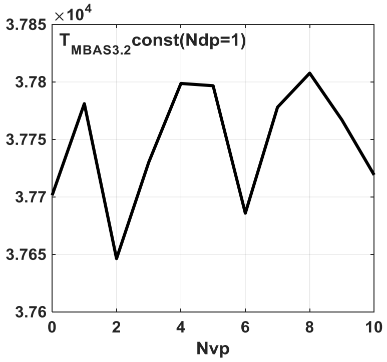
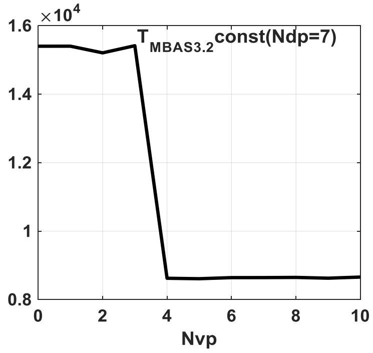


Fig.37.27 – Details of the change of $T_{MBAS\ 3.2}$const in the MBAS3.2 model on the intervals Ndp> 3, Nvp> 3

When analyzing the three-dimensional graph in Fig. 37.26, and over Ndp = const, an insignificant chaotic change in the parameter $T_{MBAS\ 3.2}$const is observed at the intervals Nvp<3 and Nvp> 3 under Ndvp> 3 and for the entire interval Nvp = [0..10] under Ndvp< 3. This is shown in detail in Fig. 37.28.



a)

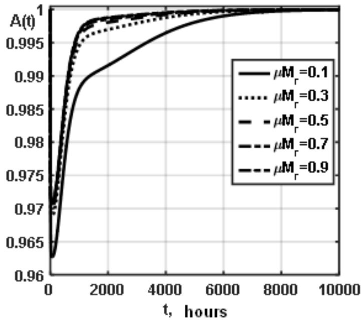


b)

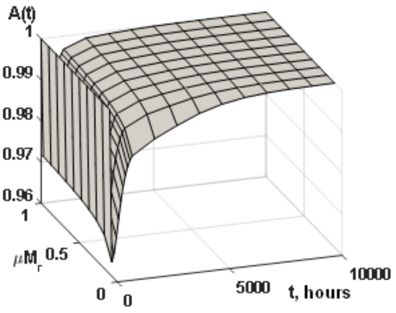
Fig.37.28 – Detailization of the change in $T_{MBAS\ 3.2}$const of the model MBAS3.2 on slices Ndp = 1 (a), Nvp = 7 (b)

Explanation of this dependence follows from the difference in the input parameters λMs and λMr – with their accepted values (λMs = 5e-3 and λMr = 1e-3), the transition to the maintenance state by vulnerabilities is performed with greater intensity.

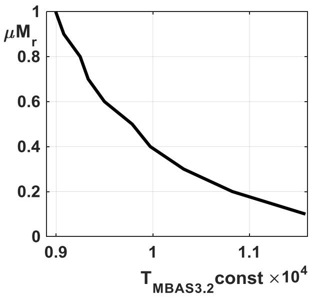
Next, the influence of the intensity of the detecting and eliminating the μMr defect on the resulting parameters of $T_{MBAS\ 3.2}$const and $A_{MBAS\ 3.2}$min is considered. When constructing models, the values of the input parameters Nv = Nd = 3, Nvp = 3, Ndp = 4 were taken.



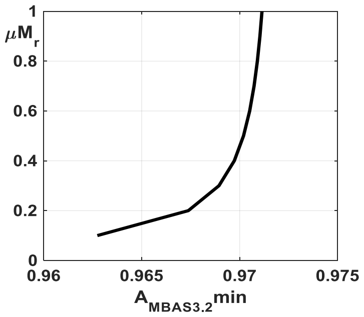
a)



b)



c)



d)

Fig. 37.29 – Graphs of the change in the resulting indicators of the MBAS3.2 model (a, b – availability functions, c – minimum availability function, d – transition period to the steady with the error of 10^{-5}) from the intensity of detection and elimination of the defect μMr

The results shown in Fig. 37.29 also show the expected result: if the maintenance quickly identifies and corrects defects, then the minimum availability function ($A_{MBAS\ 3.2}$min) increases, and the transition period to the steady state decreases. Thus, with a 10-fold acceleration of detection and elimination of defects during maintenance, the value of $A_{MBAS\ 3.2}$min increases by 0.0084, and the period of detection and elimination of all defects and vulnerabilities decreases by 1.2872 times.

37.3 Scaling of availability models for information and control systems of smart buildings

With the expansion of intellectualization systems to the level of the university campus (Fig. 36.7), the number of types of failures and points of cyber-attacks application that determine the state of a system-wide failure potentially increases. Taking into account their step-by-step elimination in the course of security and safety maintenance activities, or after their manifestation, the dimension of the Markov models increases (as the number of model fragments increases). Despite the fact that in this Chapter the typical architecture of BAS for Nd = 2 and Nv = 2 was considered, the developed models simply scale to an arbitrary number of defects and vulnerabilities. The increase in the dimensionality of the models was illustrated in Fig. 37.18, Fig. 37.21 and Fig. 37.24; And the results of calculations of models with increased dimensionality, for example, made it possible to construct the dependence of the PCR parameter (according to the $T_{MBAS\ i}$const –>min criterion) of the common maintenance model (MBAS2.1) on the initial number of defects in the Nd system.

Conclusions

The chapter presents FTA, ATA and Markov models for availability of smart BAS taking into account various variants of recovery and maintenance processes as well as the parameters of software faults and vulnerability attacks.

These models are combined to assess availability, and cyber security, to improve the accuracy of assessing availability indicators and determine the requirements for the coefficient of cyber security and availability (the level of availability of the system in the steady state).

The BAS models and technique considering the different modes and strategies of system maintenance (with and without the elimination of faults and vulnerabilities after their detection, with and without the maintenance procedures, etc.) have been described and analyzed.

Questions to self-checking

1. Please describe the classification for availability models of BASs.
2. Which are the main differences between common and separate maintenance?
3. Which are the main differences between unlimited and limited number of maintenance?
4. Which are the main differences between maintenance by reliability and security?
5. Which are the main steps of base modeling without maintenance MBAS1
6. Which are the main steps of modeling BAS with common unlimited maintenance MBAS2.1?

7. Which are the main steps of modeling BAS with common limited maintenance MBAS2.2?

8. Which are the main steps of modeling BAS with separate unlimited maintenance MBAS3.1?

9. Which are the main steps of modeling BAS with separate limited maintenance MBAS3.2?

10. Please describe the scaling of availability models for BASs.

References

1. K. S. Trivedi, D. S.fdc Kim, A. Roy and D. Medhi, Dependability and security models, – In 7th International Workshop on Design of Reliable Communication Networks, – Washington, DC, – pp. 11-20, – 2009. doi: 10.1109/DRCN.2009.5340029.

2. Q. Yu and R. J. Johnson, Smart grid communications equipment: EMI, safety, and environmental compliance testing considerations, – Bell Labs Technical Journal, – vol. 16, no. 3, – pp. 109-131, – Dec. 2011, doi: 10.1002/bltj.20525.

3. Kharchenko, V., Odarushchenko, O., Odarushchenko, V., Popov, P. Selecting mathematical software for dependability assessment of computer systems described by stiff markov chains. – CCIS, – vol. 1000, – pp. 146–162. – 2013/

4. Kharchenko V., Abdul-Hadi A.M., Boyarchuk A., Ponochovny Y. Web Systems Availability Assessment Considering Attacks on Service Configuration Vulnerabilities. – Advances in Intelligent Systems and Computing, – vol 286. – P.275-284 – 2014. doi: 10.1007/978-3-319-07013-1_26.

5. M. Grottke, H. Sun, R. Fricks and K. Trivedi, Ten Fallacies of Availability and Reliability Analysis, Service Availability. – Lecture Notes in Computer Science, – vol 5017, – pp. 187-206, – 2008, doi: 10.1007/978-3-540-68129-8_15.

6. Kharchenko V., Ponochovnyi Y., Abdulmunem AS.M.Q., Andrashov A. Availability Models and Maintenance Strategies for Smart Building Automation Systems Considering Attacks on Component Vulnerabilities. – Advances in Intelligent Systems and Computing, Vol. 582, 2017, P. 186-195. DOI: 10.1007/978-3-319-59415-6_18.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ К РАЗДЕЛУ 37

BAS	Building automation system
I&CS	Information and control systems
SDE	System of differential equations

АННОТАЦИЯ

В разделе представлены марковские модели готовности информационно-управляющих систем умных домов, учитывающие различные варианты процессов восстановления и обслуживания, а также параметров проявления программных дефектов и атак на уязвимости, что позволяет повысить точность оценки и определить требования к коэффициенту готовности и средствам киберзащиты. Рассмотрены реализации аналитических моделей готовности информационно-управляющих систем умных домов с учетом отказов и атак на компоненты их архитектуры (MBAS1), с учетом проведения неограниченного количества процедур общего и раздельного обслуживания (MBAS2.1, MBAS3.1) и с учетом проведения ограниченного количества процедур общего и раздельного обслуживания (MBAS2.2, MBAS3.2) по надежности и безопасности.

У розділі представлені марковські моделі готовності інформаційно-керуючих систем розумних будинків шляхом врахування різних варіантів процесів відновлення і обслуговування, а також параметрів прояву програмних дефектів і атак на вразливості, що дозволяє підвищити точність оцінювання та визначити виконання вимог до коефіцієнту готовності та засобів кіберзахисту. Розглянуті реалізації аналітичних моделей готовності інформаційно-керуючих систем розумних будинків з урахуванням відмов і атак на компоненти їх архітектури (MBAS1), з урахуванням проведення необмеженої кількості процедур загального і роздільного обслуговування (MBAS2.1, MBAS3.1) і з урахуванням проведення обмеженої кількості процедур загального і роздільного обслуговування (MBAS2.2, MBAS3.2) по надійності і безпеці.

Building automation systems Markov models are discussed in the section. Markov models for availability of information and control systems of smart buildings have been improved by taking into account different variants of recovery and maintenance processes, as well as parameters of manifestation of software defects and vulnerability attacks, which allows to increase the accuracy of evaluation and to determine the fulfillment of the requirements for the availability factor and means of cyber security. Analytical models for the availability of information and control systems of smart homes, taking into account failures and attacks on their architecture components (MBAS1), have been developed considering the unlimited number of common and separate maintenance procedures (MBAS2.1, MBAS3.1) and the limited number of common and separate maintenance (MBAS2.2, MBAS3.2) procedures for reliability and security are discussed.

V. Sklyar, V. Kharchenko, E. Babeshko, A. Kovalenko, O.Illiashenko, O. Rusin, A. Panarin, S. Razgonov, D. Ostapec, I. Zhukovyts'kyy, S. Stirenko, O. Tarasyuk, A. Gorbenko, A. Romanovsky, O. Biloborodov, I. Skarha-Bandurova, E. Brezhniev, A. Stadnik, A. Orekhov, T. Lutskiv, V. Mokhor, O. Bakalynskyi, A. Zhylin, V. Tsurkan, M. Q. Al-sudani, Yu. Ponochovnyi

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS.

Secure and resilient systems, networks and infrastructures

Multi-book, Volume 2

Editor Vyacheslav Kharchenko

National Aerospace University n. a. N. E. Zhukovsky
"Kharkiv Aviation Institute"
17 Chkalova street, Kharkiv, 61070, Ukraine
http://www.khai.edu