Building cybersecurity systems in hybrid cloud ecosystems based on discrete-event simulation

Iryna Tregubenko[1,*,†], Emil Faure[1,2,†], Viktor Kotetunov[1,†] and Yevhen Morshch[1,†]

[1] *State Scientific and Research Institute of Cybersecurity Technologies and Information Protection, M. Zaliznyaka Str., 3/6, Kyiv, 03142, Ukraine*

[2] *Cherkasy State Technological University, Shevchenko Blvd., 460, Cherkasy, 18006, Ukraine*

Abstract

The nature of threats and vulnerabilities of information complex systems and hybrid cloud ecosystems in cyberspace is complex, heterogeneous, multifactorial, flexible and rapidly modifiable. Traditional approaches to building security systems do not take into account many threats that are not obvious from the user's point of view. To study and develop new approaches to building reliable data protection systems and analyze their effectiveness, the vulnerabilities of confidential data in modern cloud environments are assessed, taking into account the modification of modern types of software architecture. Particular attention is paid to the features of microservice software architecture and their impact on the vulnerabilities of cloud ecosystems. It is proposed to use discrete-event simulation approaches to modeling security systems, taking into account the characteristics of hybrid cloud ecosystems.

Keywords

cybersecurity systems, cloud, data protection, microservice architecture, discrete-event simulation

1. Introduction

Cloud technologies have a sustainable development, are constantly modified, in particular in terms of increasing the speed of computing, expanding the capabilities of analytics, and the wider use of big data and artificial intelligence [1, 2]. The nature of threats and vulnerabilities of information complex systems and hybrid cloud ecosystems in cyberspace is complex, heterogeneous, multifactorial, flexible and rapidly changing. Traditional approaches to building security systems do not take into account many threats that are not obvious from the user's point of view.

In order to research and develop new approaches to building reliable data protection systems and analyze their effectiveness, it is necessary to conduct a thorough assessment of vulnerabilities of confidential data in modern cloud environments, taking into account the modification of modern types of software architecture, in particular the impact of the popular microservice architecture of software tools on the vulnerabilities of cloud environments, and to propose appropriate technologies for modeling protection systems, taking into account the characteristics of hybrid cloud ecosystems.

2. Related works

Ensuring security in cloud environments remains a pressing issue against the backdrop of the growing dependence of supporting the vital processes of the information society on cloud platforms and services based on cloud environments.

CH&CMiGIN'25: Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks, June 20–22, 2025, Kyiv, Ukraine

[*] Corresponding author.

[†] These authors contributed equally.

✉ wr.irttri@gmail.com (I. Tregubenko); e.faure@chdtu.edu.ua (E. Faure); v.kotetunov@gmail.com (V. Kotetunov); yevhen.phd@gmail.com (Y. Morshch)

🌐 http://irttri.com/ (I. Tregubenko)

iD 0000-0003-4991-5511 (I. Tregubenko); 0000-0002-2046-481X (E. Faure); 0000-0002-0214-3369 (V. Kotetunov); 0000-0003-0131-2332 (Y. Morshch)

In addition, the large amounts of data already stored in popular clouds such as Azure, Amazon, Google Cloud, etc. are constantly increasing and will contain a significant portion of files with confidential information, including configuration files, API keys, credentials, etc. The problem of leakage of conference information due to incorrect settings of cloud segments remains and can lead to security breaches and unauthorized access to various cloud services such as databases, cloud infrastructure, third-party APIs, etc.

Modern software, most applications, are not developed from scratch; some of the necessary functions are used from the ready-made functionality of other services, often cloud-based. At the same time, access to services is protected by various types of credentials: passwords, tokens, usernames, certificates, API keys, etc. In turn, such confidential credentials may not be reliably protected at the software development stage. As a result, the likelihood of misuse of resources and unauthorized access to information increases significantly and can lead to critical consequences [3, 4]. Examples are known. Many researchers confirm frequent leaks of confidential data through various channels [3, 5, 6, 7]:

- modern version control platforms;
- virtual server images;
- incorrectly configured cloud storage;
- large language models based on code sharing platforms;
- mobile applications;
- mini applications, micro services;
- Docker containers, etc.

In addition to quantifying confidential data leakage, studies suggest various approaches to increasing the efficiency of detection processes [8, 9, 10, 11], which can be useful for further improving methods and models of data protection in cloud environments.

3. Features of protecting cloud applications with microservice architecture

The transition to new software architectures is leading to an increase in the problem of confidential data leakage and end-user losses. It is interesting to investigate the effectiveness of security processes in applications and platforms based on microservice architecture. Such hybrid ecosystems consist of many elements: the parent application server, the developer server, the hosting service, the services used by the developers, the client application, etc. A mandatory component of the joint efficient operation of all these elements is their constant joint authentication, which differs from user authentication on the platform or in applications. This is a type of technical authentication that allows an attacker to access various services on behalf of developers, which can be a significant vulnerability in the context of currently used insecure software development practices.

For example, research [7] has shown that a significant number of WeChat (over 36 thousand detections) and Baidu (112 detections) microservices and applications have non-obvious vulnerabilities related to development technologies and features of the chosen architecture. Most often, this is due to the so-called hard auto-identification used by developers. Undoubtedly, this causes quite significant security issues, including privacy violations for both developers and users. A study [7] confirmed the conclusion that any knowledgeable attacker on the network can make it impossible for an application to work, even if they do not have their own account on the parent platform. Also, such an offender will potentially have the opportunity to engage in phishing, gain access to confidential information, and send out mailings to platform users.

Thus, the analysis shows that the nature of threats in cyberspace is heterogeneous, multifactorial, flexible, and rapidly changing. Traditional algorithms for detecting all threats at each access point and in each user application, including in the cloud environment, and neutralizing them are not effective. To research and develop new approaches to building data protection systems and analyze their effectiveness, there is a need to develop new approaches to modeling such systems, taking into account the features of integrated cloud environments.

4. Research of methods for modeling a data protection system based on DES principles for hybrid cloud ecosystems

Usually, when modeling protection systems, it is proposed to use continuous modeling methods. The main features of which are as follows:

- The system is considered to be constantly changing in time;
- Time is the main parameter and often the criterion;
- Models continuously track the state of the system in the time dimension.

It is suggested to change the viewing angle. Continuous monitoring or modeling of a complex hybrid ecosystem is a complex and resource-intensive task. It is proposed to investigate the possibility of applying a discrete approach to modeling systems with continuous information processes, in particular, to consider the principles of DES modeling.

Usually, Discrete-Event Simulation (hereinafter referred to as DES) considers a system as a sequence of events in time. It is assumed that events occur at certain points in time, separated from each other. The basic features of DES modeling can be identified:

- An event occurs at a specific time point. As defined above, DES is used to model systems, processes, and flows that can change at identifiable points in time. At the same time, DES will not track the state of flows, systems, processes between these moments of time, i.e. continuously.
- Focus on events. That is, DES focuses on events that change the state of the system.
- Skipping time intervals between events. DES does not investigate the state of the system between the moments of time when the event occurred, which reduces the complexity and the amount of resources required for modeling processes. During the modeling process, DES moves from one event to the next, the next.
- Widespread use of queueing theory. This is not a mandatory feature of DES, but in cases where the processes of receiving and servicing requests are significant in the system, the use of queue theory is more than reasonable. Unlike continuous modeling, the main features of DES are as follows:
- No changes between events
- The main parameter is not time, but an event
- The state of the system between moments of time does not matter.

When modeling data protection in complex heterogeneous systems, including hybrid cloud ecosystems, we assume that:

- From the point of view of functional suitability, the state of protection systems is significant at the time of a cyber incident, attack, unauthorized access attempt, etc.
- Outside of the moments of attack and intrusion attempts, the state of the protected system is unimportant.

At the same time, such a system can be considered as discrete, and the event that will affect the change in the system state is the moment of attack, and there is a finite set of such moments. Figure 1 shows the DES process of a data protection system in hybrid cloud ecosystems as a discrete-event system.

The properties of data protection systems in hybrid cloud ecosystems identified by the authors allow us to attribute them to discrete-event systems, which are most widely used in the theory of automata with their variants, such as various graphs and the mathematical apparatus of Petri nets. The latter has an intuitive graphical representation, has been widely used in applied problems of modeling asynchronous and parallel dynamic systems, and has algorithmic developments.

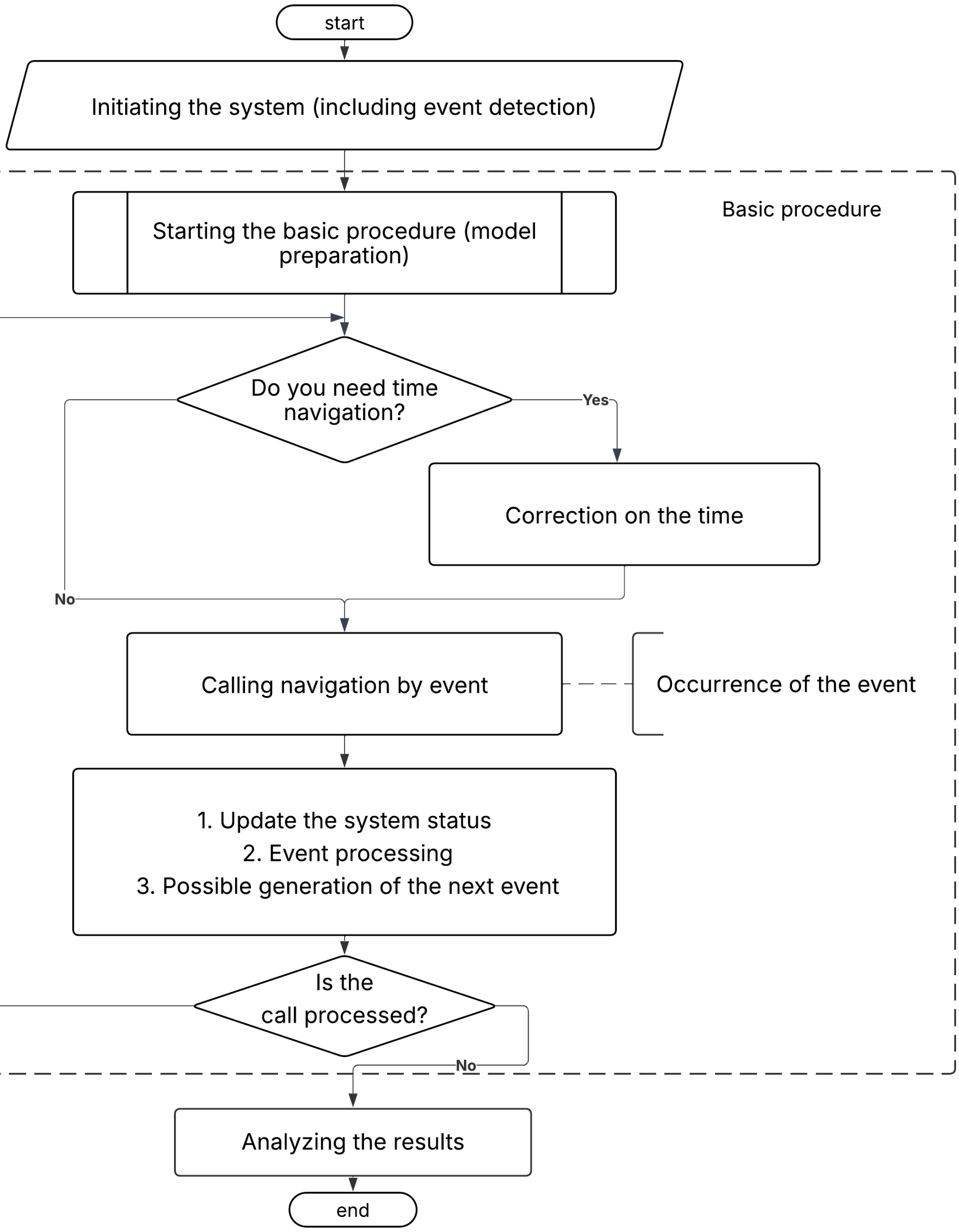
start
Initiating the system (including event detection)
Basic procedure
Starting the basic procedure (model preparation)
Do you need time navigation?
Yes
Correction on the time
Yes
No
Calling navigation by event
Occurrence of the event
1. Update the system status
2. Event processing
3. Possible generation of the next event
Is the call processed?
No
Analyzing the results
end

Figure 1: DES process of a data protection system as a discrete-event system.

In the classical representation [12], a Petri net has four components:

$$SP = (P, T, I, O) \tag{1}$$

where:
$P = \{p_1, p_2, p_3, ..., p_n\}$ is a finite set of positions, $n >= 0$ is the power of the set P;
$T = \{t_1, t_2, t_3, ..., t_m\}$ is a finite set of transitions, $m >= 0$ is the power of the set T;
$P \cap T = \varnothing$ - the sets P and T do not intersect.
$p_i, i = 1, 2, ..., n$ - any element of P;
$t_j, j = 1, 2, ..., m$ - any element of T;
I - is an input function that represents the transition of t_j to the set of positions $I(t_j)$;
O - is an output function that represents the transition of t_j to the set of positions $O(t_j)$.

Thus, the structure of a Petri net is defined by its positions, transitions, input and output functions, and is usually represented as a graph. Such a graph is a bipartite, oriented multigraph, in which one part is formed by positions and an arc by transitions. Many arcs connect positions and transitions. In such a graph, there are two types of nodes: a position (denoted by a circle) and a transition (denoted by a bar). Arcs are oriented and connect positions and transitions. An arc that has a direction from position pi to transition tj determines the position that is the output of the transition.

In the further development of distributed modeling methods, the models and structure became more complex, high-level networks, nested, functional, hierarchical, etc. appeared. A characteristic feature was often the modular representation of diverse distributed systems, with the allocation of a separate autonomous module that functioned independently, had a certain internal state (whether dynamic or stable) and a certain set of inputs and outputs. The inputs/outputs were used to interact with the external environment.

To solve the problem of modeling protection systems, it is necessary that such a functional module can be located separately from the basic system.

We will impose certain restrictions: we will assume that external influences from the environment are directed exclusively to the set of inputs, and the result can be considered exclusively in the set of outputs. This approach is inherent in functional networks, which have a more regular structure in the presence of these restrictions.

$$SPF = (P, T, I, O, PI, PO) \tag{2}$$

where:
$P = \{p_1, p_2, p_3, ..., p_n\}$ is a finite set of positions, $n >= 0$ is the power of the set P;
$T = \{t_1, t_2, t_3, ..., t_m\}$ is a finite set of transitions, $m >= 0$ is the power of the set T;
$P \cap T = \varnothing$ - the sets P and T do not intersect.
$p_i, i = 1, 2, ..., n$ - any element of P;
$t_j, j = 1, 2, ..., m$ - any element of T;
I - is an input function that represents the transition of t_j to the set of positions $I(t_j)$;
O - is an output function that represents the transition of t_j to the set of positions $O(t_j)$.
$PI \subseteq P$ - is the set of input positions of the Petri net;
$PO \subseteq P$ - is the set of output positions of the Petri net.
$PI \cap PO = \varnothing$ - the sets PI and PO do not intersect.

With this approach, another set PV is formed - the set of internal positions that are not included in the sets PI and PO.

Given that the defense systems under consideration will receive heterogeneous threats at the input positions and will require different times for the defense system to react, our model needs to include time delays for threat processing and determine the moment when the DES process is initiated. Let's get the model:

$$SPFD = (P, T, I, O, PI, PO, DT, M) \tag{3}$$

where:
$P = \{p_1, p_2, p_3, ..., p_n\}$ is a finite set of positions, $n >= 0$ is the power of the set P;
$T = \{t_1, t_2, t_3, ..., t_m\}$ is a finite set of transitions, $m >= 0$ is the power of the set T;
$P \cap T = \varnothing$ - the sets P and T do not intersect.
$p_i, i = 1, 2, ..., n$ - any element of P;
$t_j, j = 1, 2, ..., m$ - any element of T;
I - is an input function that represents the transition of t_j to the set of positions $I(t_j)$;
O - is an output function that represents the transition of t_j to the set of positions $O(t_j)$.
PI - is the set of input positions of the Petri net;
PO - is the set of output positions of the Petri net.
DT - time delay for threat processing;
M - is a position label that characterizes a certain state of the model.

It is also possible to increase the efficiency and accuracy of the analysis and the corresponding model built on the basis of hierarchical Petri nets. In this case, the same Petri nets can act as constituent elements. In the task of ensuring data protection in hybrid cloud environments, each nested element can identify different types of threats and/or perform direct protection actions. The mathematical model in this case will be as follows:

$$SPI = (P, T, I, O, M_0, A, AI, AO) \tag{4}$$

where:
$P = \{p_1, p_2, p_3, ..., p_n\}$ is a finite set of positions, $n >= 0$ is the power of the set P;
$T = \{t_1, t_2, t_3, ..., t_m\}$ is a finite set of transitions, $m >= 0$ is the power of the set T;
$P \cap T = \varnothing$ - the sets P and T do not intersect.
$p_i, i = 1, 2, ..., n$ - any element of P;
$t_j, j = 1, 2, ..., m$ - any element of T;
I - is an input function that represents the transition of t_j to the set of positions $I(t_j)$;
O - is an output function that represents the transition of t_j to the set of positions $O(t_j)$.
M_0 - is the set of initial labeling of the network;
A - is the set of nested Petri nets in p_i;
AI - is the set of input positions of nested Petri nets;
AO - is the set of output positions of nested Petri nets.

The developed model, among other things, can allow analyzing the dynamics of the system being modeled.

5. Discussion

Is it enough to focus on the approaches proposed in this article for the problem under consideration? Certainly not. The research will continue. It would be advisable to consider other approaches for modeling complex security systems in cloud heterogeneous ecosystems that will allow first to effectively model and then find a platform for building such systems, taking into account the rapid changes in IT technologies, the growth of processed data, the development of quantum technologies, etc. Sustainable search for appropriate solutions will contribute to the development of more effective data protection technologies in complex systems and hybrid cloud ecosystems, which will generally contribute to the creation of a more secure cyberspace.

6. Conclusions

The article reveals the complex and multifactorial nature of threats and vulnerabilities of information large systems and hybrid cloud ecosystems in cyberspace. The following properties of such systems are important to take into account: heterogeneity, hierarchy, multifactoriality, flexibility, and high speed of complex changes. Traditional approaches to building defense systems and corresponding algorithms are not effective enough and usually do not take into account a whole group of threats associated with the architectures of the developed defense systems.

To research and develop new approaches to building data protection systems in hybrid cloud ecosystems and analyze their effectiveness, it is proposed to use DES technologies to model protection systems, taking into account the characteristics of hybrid complex systems.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] I. B. Tregubenko, V. Y. Kotetunov, Problems of information security of hybrid integration ecosystems based on cloud technologies, in: Proceedings of the III International Scientific and Practical

Conference (Chernihiv, February 27, 2025), 780, Scientific and Educational Innovation Center for Social Transformations, Chernihiv, 2025, pp. 727–729. doi:10.54929/conf_reicst_27_02_25.

[2] O. C. Okoro, et al., Optimization of maintenance task interval of aircraft systems, International Journal of Computer Network and Information Security 14 (2022) 77–89. doi:10.5815/ijcnis.2022.02.07.

[3] M. Dahlmanns, C. Sander, R. Decker, K. Wehrle, Secrets revealed in container images: An internet-wide study on occurrence and impact, in: Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security, ASIA CCS '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 797–811. URL: https://doi.org/10.1145/3579856.3590329. doi:10.1145/3579856.3590329.

[4] J. S. Al-Azzeh, M. A. Hadidi, R. S. Odarchenko, S. Gnatyuk, Z. Shevchuk, Z. Hu, Analysis of self-similar traffic models in computer networks, International Review on Modelling and Simulations 10 (2017) 328–336. doi:10.15866/iremos.v10i5.12009.

[5] R. Feng, Z. Yan, S. Peng, Y. Zhang, Automated detection of password leakage from public github repositories, in: Proceedings of the 44th International Conference on Software Engineering, ICSE '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 175–186. URL: https://doi.org/10.1145/3510003.3510150. doi:10.1145/3510003.3510150.

[6] Y. Huang, Y. Li, W. Wu, J. Zhang, M. R. Lyu, Your code secret belongs to me: Neural code completion tools can memorize hard-coded credentials, Proceedings of the ACM on Software Engineering 1 (2024) 2515–2537. URL: http://dx.doi.org/10.1145/3660818. doi:10.1145/3660818.

[7] S. Baskaran, L. Zhao, M. Mannan, A. Youssef, Measuring the leakage and exploitability of authentication secrets in super-apps: The wechat case, in: Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses, RAID '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 727–743. URL: https://doi.org/10.1145/3607199.3607236. doi:10.1145/3607199.3607236.

[8] V. S. Sinha, D. Saha, P. Dhoolia, R. Padhye, S. Mani, Detecting and mitigating secret-key leaks in source code repositories, in: Proceedings of the 12th Working Conference on Mining Software Repositories, MSR '15, IEEE Press, 2015, p. 396–400.

[9] C. Farinella, A. Ahmed, C. Watterson, Git Leaks: Boosting Detection Effectiveness Through Endpoint Visibility , in: 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 701–709. URL: https://doi.ieeecomputersociety.org/10.1109/TrustCom53373.2021.00103. doi:10.1109/TrustCom53373.2021.00103.

[10] N. Kuzmenko, et al., Airplane flight phase identification using maximum posterior probability method, in: IEEE 3rd International Conference on System Analysis and Intelligent Computing (SAIC), 2022, pp. 1–5. doi:10.1109/SAIC57818.2022.9922913.

[11] A. Saha, T. Denning, V. Srikumar, S. K. Kasera, Secrets in source code: Reducing false positives using machine learning, in: 2020 International Conference on COMmunication Systems NETworkS (COMSNETS), 2020, pp. 168–175. doi:10.1109/COMSNETS48256.2020.9027350.

[12] J. L. Peterson, Petri Net Theory and the Modeling of Systems, Prentice Hall PTR, USA, 1981. URL: https://dl.acm.org/doi/10.5555/539513.