APPLICATION OF THE STRIDE FRAMEWORK IN A MICROSERVICE APPLICATION FOR ELECTRONIC BUSINESS

DOI: 10.5937/JEMC2502100M

UDC: 004.056:004.056.5
Original Scientific Paper

Tamara MILIC[1], **Zeljko STOJANOV**[2], **Igor VECSTEJN**[3]

[1]University of Novi Sad, Technical faculty "Mihajlo Pupin" Zrenjanin, 23000 Zrenjanin, Đure Đakovića bb, Republic of Serbia
Corresponding Author, Email: tamara.milic@tfzr.rs
ORCID-iD (https://orcid.org/0009-0004-7273-4801)
[2]University of Novi Sad, Technical faculty "Mihajlo Pupin" Zrenjanin, 23000 Zrenjanin, Đure Đakovića bb, Republic of Serbia
ORCID-iD (https://orcid.org/0000-0001-6930-5337)
[3]University of Novi Sad, Technical faculty "Mihajlo Pupin" Zrenjanin, 23000 Zrenjanin, Đure Đakovića bb, Republic of Serbia
ORCID-iD (https://orcid.org/0009-0009-9940-4411)

Paper received: 10.11.2025.; Paper accepted: 07.12.2025.

In this paper, we identify security threats in a microservice-based application using the STRIDE framework. The study addresses the question of how the architecture of STRIDE procedure, which is aligned with Data Flow Diagram (DFD) trust boundaries, can support early threat identification in microservice architecture. STRIDE methodology is used for categorizing security threats based on six criteria: identity spoofing, data falsification, denial of action or responsibility, information disclosure, denial of service, and privilege escalation. In addition, STRIDE enables assessment of the severity and status of the identified threats. The research relies on the creation of microservice architectural model created in draw.io and DFD of application that was constructed with Microsoft Threat Modeling Tool. Applied to the case study system, the procedure identified 52 threats that were dominated by Spoofing (29%) and Denial of Service (25%), followed by Elevation of Privilege (21%), Repudiation (17%), and Information Disclosure and Tampering (4% each). The paper presents a simple STRIDE procedure that can be reused for other microservice applications. It includes four steps: prepare an architectural diagram, create a DFD with trust boundaries, run STRIDE analysis in a certain tool, and summarize results in a matrix of threats per microservice to support prioritization of mitigation measures.

Key words: Microservice architecture; Application security; Vulnerability; STRIDE; Microsoft Threat Modeling Tool.

INTRODUCTION

Microservice architecture enables independent scaling, faster introduction of changes and organization of teams so that each team develops and maintains a specific service in the application, while at the same time increasing the attack surface, because it contains a greater number of components, Application Programming Interface (API), third-party libraries and cloud services (Hannousse & Yahiouche, 2021), as well as heterogeneous technologies and configurations that open up new threat vectors in practice (Yarygina & Bagge, 2018). For these reasons, attackers often target interservice interactions, identities, and dependencies (Tapia et al., 2020), so security must be considered already in the Security by Design phase with systematic threat modeling (Chondamrongkul et al., 2021; McGraw, 2004; Zdun et al., 2023).

Threat modeling and architectural risk analysis are proactive security engineering practices designed to identify, prioritize, and mitigate risks early in the software development life cycle (McGraw, 2004; Oluwaferanmi, 2025). In microservices, threats can come from internal actors (i.e., internal attacks) or from outside (i.e., external attacks). To properly secure microservices-based systems, all threats, regardless of their origin, must be detected and

ISSN 2217-8147 (Online)

prevented using available mitigation techniques or proposing innovative solutions (Hannousse & Yahiouche, 2021). Eliminating security threats in the early stages of system development can significantly reduce the overall cost by eliminating a large number of potential vulnerabilities. That is why it is extremely important to monitor the connection of relevant classes with specific architectural components and flows (Chondamrongkul et al., 2021). Because it is easier to address security vulnerabilities early in development, and security requirements are already considered in the architectural design, an architectural specification is a valuable resource for detailed design, implementation, and validation (Xu & Pauli, 2006).

In this work, STRIDE (Spoofing (S), Tampering (T), Repudiation (R), Information Disclosure (ID), Denial of Service (DoS), Elevation of Privilege (EoP)) is adopted as a framework that serves for the systematic identification and categorization of threats against a microservice application that is naturally aligned with the DFD and thus enables consistent mapping of threats to typed elements of the architecture (Hernan et al., 2006). Such a link with the DFD makes it easier to observe threats exactly where they arise, i.e., on trust boundaries, and to propose measures at the right points in the architecture (Rouland et al., 2021) that also include typical microservice patterns such as API gateways, broker channels, etc. (Das et al., 2024; Khan et al., 2017). The application of STRIDE in practice relies on the creation of a precise DFD model with clearly defined and marked trust boundaries and a systematic enumeration of threats over each DFD element in the model with the help of an appropriate Microsoft Threat Modeling Tool (MTMT) (Bygdas et al., 2021; NIST, 2016; Microsoft, 2022).

However, today's works containing the STRIDE framework in the field of microservice architectures are mostly concerned with general guidelines, comparison of tools in use, or existing theoretical frameworks. It is very rare to present a concrete, step-by-step procedure that begins with a representation of the architecture of a real microservice application and ends with a clearly structured threat matrix for each microservice. For this reason, there is still a lack of practical architecture-based examples showing how STRIDE can be systematically applied to a real microservices application and how the resulting threats can be summarized, in a way to facilitate protection planning.

This paper aims to create a DFD model with adequately set trust boundaries, to carry out identification, classification, and assessment of the level of severity of security threats using the STRIDE framework in an application that functions according to the principle of microservice architecture. The main research objective of this paper is to investigate how a stepwise STRIDE procedure, which is aligned with DFD trust boundaries, can support early identification and prioritization of security threats in the microservice architecture. To achieve this aim, this paper presents: (1) an architecture-oriented threat modeling procedure that connects architectural model and DFD with the results of STRIDE analysis, and (2) STRIDE threat matrix for each microservice that is intended to support the planning of mitigation measures and the reduction of security threats. The system used in this case study is based on an open-source e-commerce application (GoogleCloudPlatform, n.d.), which was chosen because it represents a publicly available microservice system consisting of nine coupled services and realistic patterns of interservice communication. Therefore, it is a representative and reproducible example of a modern microservice application, where the confidentiality of proprietary data in the system is not violated.

After the introductory section, the work is organized as follows: in Section 2, the methodology is presented, which consists of an architectural diagram, the creation of a DFD model, which in this case represents the security model of the system, the presentation of the model in the Microsoft Threat Modeling Tool, the definition of trust boundaries and the rules for the application of STRIDE by DFD elements. Section 3 is oriented to the discussion of the results obtained on the basis of the previously modeled DFD security model, through the STRIDE methodology. Section 4 provides a conclusion and suggestions for future research.

METHODOLOGY

This section describes an architecture-oriented threat modeling procedure for a microservice application, which consists of the following steps: (1) Preparation of an architectural view of the system, (2) Creation of a DFD model with clearly marked trust boundaries, (3) STRIDE threat categorization of identified threats according to the DFD model using the Microsoft Threat Modeling

Tool, (4) Tabular presentation of identified threats for each microservice in the application.

Research design

This paper follows a single case study design in the sense of (Yin, 2014), focusing on an open-source microservice e-commerce application as a representative example of a microservice system. The goal of this study is to examine how the architecture-driven STRIDE framework that is aligned with DFD trust boundaries can be used to identify and prioritize security threats in a system.

This paper is based on a qualitative approach using the STRIDE methodology for the identification and analysis of security threats in an application based on microservice architecture. By using the STRIDE framework to detect and categorize threats, it is possible to minimize the risk of system security threats, but not to eliminate them completely. Applied research is reflected in the practical use of the STRIDE on the target application, because it enables systematic identification of Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege threats exactly on the elements (components) where they arise (Hernan et al., 2006; Shostack, 2014).

STRIDE was selected over other threat modeling approaches because it is naturally aligned with DFD elements and it provides a structured categorization of threats (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) that can be mapped to components of architecture and trust boundaries. In this design, STRIDE is used as the primary framework for threat identification at the architecture level, while the case study design allows us to analyze in depth how this approach behaves in microservice modelling on a real example.

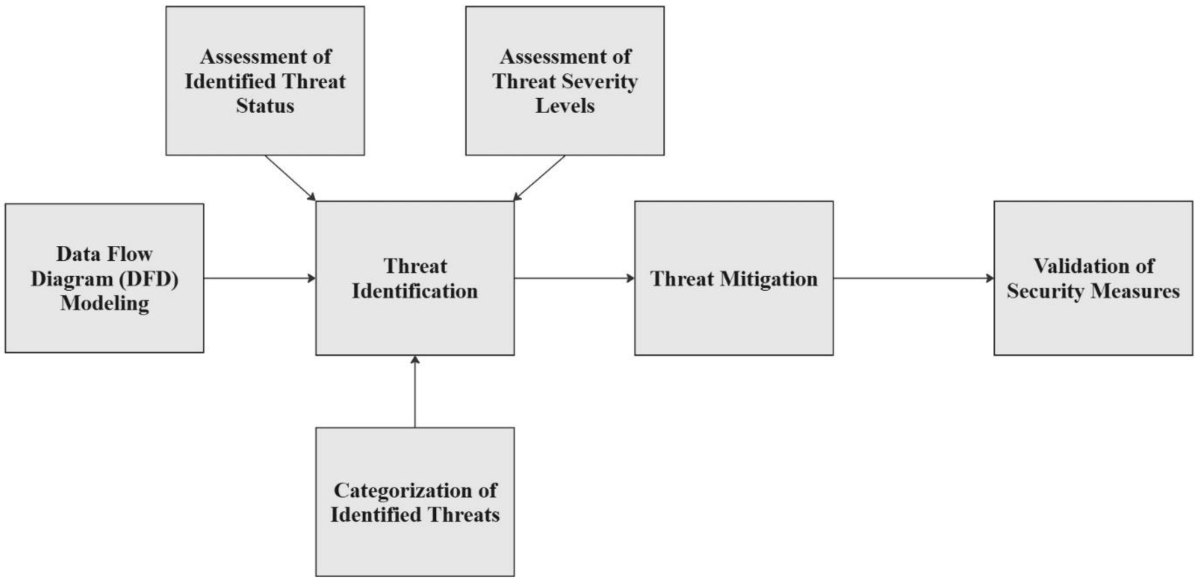
Assessment of Identified Threat Status
Assessment of Threat Severity Levels
Data Flow Diagram (DFD) Modeling
Threat Identification
Threat Mitigation
Validation of Security Measures
Categorization of Identified Threats

Figure 1: The STRIDE methodology functioning process

As shown in Figure 1, the first and basic step in the modeling and analysis process is to construct the DFD. Based on the DFD model, insight is gained into specific system components and trust boundaries, which are extremely important for identifying security threats. Properly defined trust boundaries allow the tool to identify system security threats and to classify each identified threat into one or more STRIDE categories (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege). Security threats are identified for each external entity, process, data flow, and data store. For every threat, the tool also records a severity level and a mitigation status, which are later used in the analysis to understand which STRIDE categories and microservices are most exposed and which threats should be prioritized for treatment.

Visual representation of the application architecture

The application is a microservice-based e-commerce platform. Users of the studied application can easily view the items available at a given time, store them in the cart, and make online purchases. Figure 2 shows a microservice architecture model for the application created using the draw.io tool (GoogleCloudPlatform, n.d.).

Based on Figure 2, a total of nine microservices can be noticed: frontend, recommendation, product catalog, cart, checkout, shipping, payment, currency, and email. Users access the application through the frontend service. The frontend service thus represents the entry point into the system. Checkout service connects payment, currency conversion, shipping, and email options. Payment enables a simple and fast way of purchasing, currency conversion facilitates the payment process itself, the purchase confirmation with the user's personal data received via e-mail offers the assurance of a successful purchase, and the shipping process allows the user to track the shipment and secure collection. The cart service stores user information by storing it in the Redis cache. In this way, it is possible to quickly access consumer data. The recommendation service offers users personalized benefits based on the product catalog. The ad service is responsible for displaying advertising content within the application itself. It can be concluded that the application presented through the draw.io tool offers a good visual overview of the system, which makes it easier to spot security threats. A modeled diagram using the draw.io tool represents the first step in the analysis and identification of threats in applications based on microservices architecture. The presented diagram serves as the basis for the implementation of the model in the Microsoft Threat Modeling Tool. In other words, the diagram created in the draw.io tool represents an architectural representation of the system, while the diagram generated in the Microsoft Threat Modeling Tool represents a security model that enables a clearer definition of trust boundaries (Microsoft, 2016; Microsoft, 2022) and the identification of potential system threats based on properly set trust boundaries. Figure 3 shows the model for the application using the Microsoft Threat Modeling Tool.

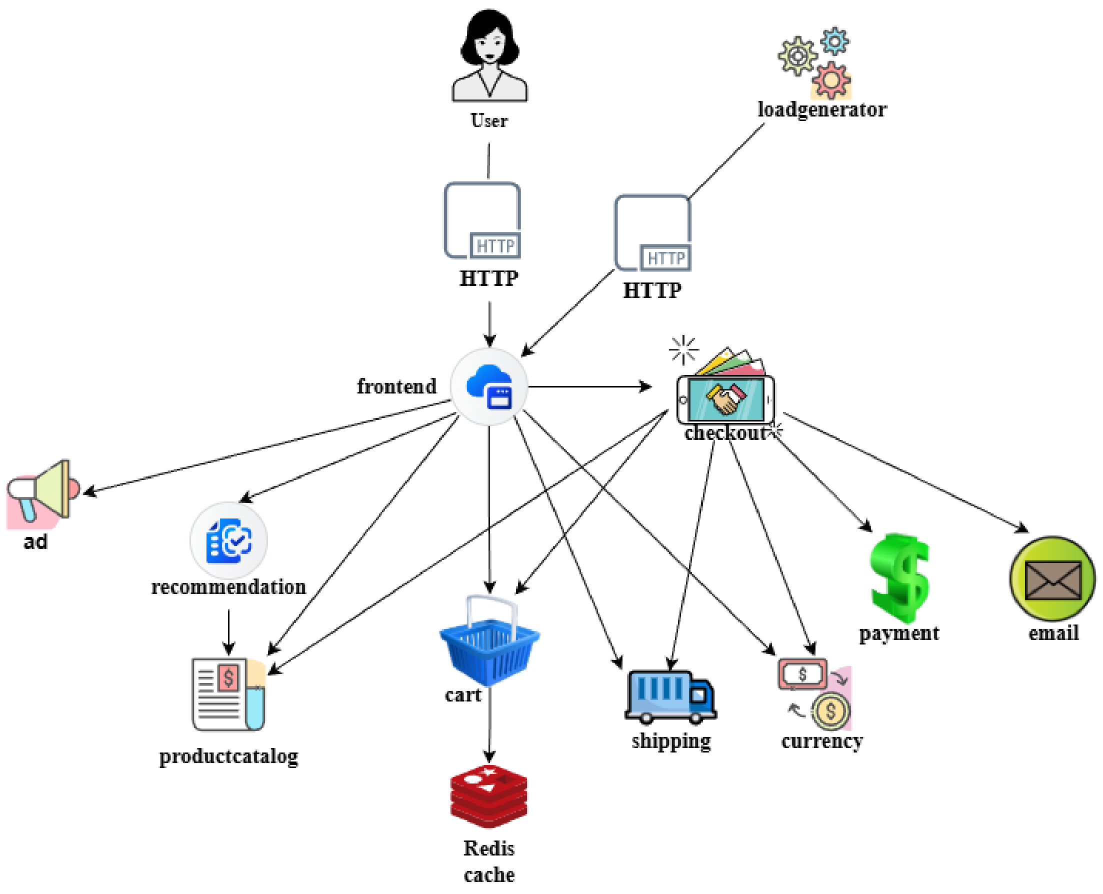

Figure 2: Microservice application architecture

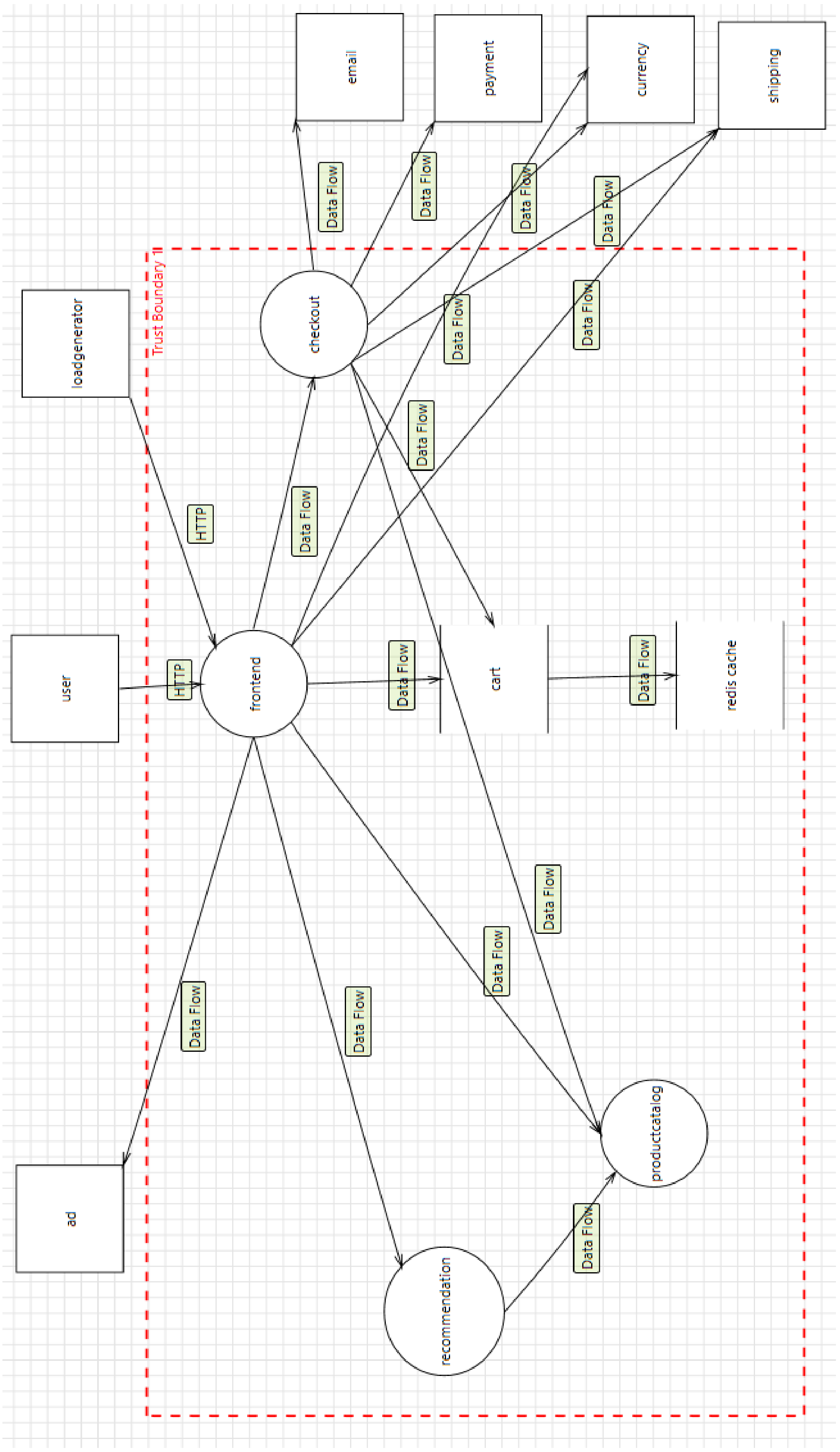

Figure 3: Microsoft Threat Modeling Tool application diagram

Figure 3 shows an example of a DFD created using the Microsoft Threat Modeling Tool, version 2014. The diagram is derived from the first model implemented using the draw.io tool. The external entities are: user, loadgenerator, payment, currency, shipping, ad, and email. There are four processes in total, namely: frontend, checkout, recommendation, and productcatalog. Data stores include: cart and Redis cache. In the end, there are a total of 17 data streams. The trust boundary, in addition to the specific data streams it cuts through, also contains processes (frontend, recommendation, productcatalog, checkout) and data stores (Redis cache). In defining the trust boundaries, we relied on the guidelines of the MTMT tool documentation. In this case, all internal microservices and the database are within a single trust boundary because they run in the same sandbox. External users, third-party payment systems, and other clients accessing the Internet are outside this boundary. Any call from their side to internal microservices is treated as potentially unreliable. On the DFD, these trust boundaries are clearly shown around the internal set of microservices and on the interfaces to external actors so that all flows that cross the trust boundary can be systematically analyzed through the STRIDE method. The diagram created in this way provides a solid foundation for identifying and further analyzing potential threats in the microservice architecture.

RESULTS

All quantitative results that are reported in this section are derived list of threats that has been automatically generated by MTMT based on DFD model of the application. Analysis reflects potential model-based threats rather than vulnerabilities that are confirmed from empirical penetration testing.

Based on created DFD, the Microsoft Threat Modeling Tool identified 52 security threats in the application. The tool automatically classified all identified threats into six categories according to STRIDE methodology criteria: identity spoofing, data falsification, denial of action or responsibility, information disclosure, denial of service, and privilege escalation. The obtained data were statistically processed in Excel and presented as a percentage through the diagram given in Figure 4.

Figure 4 shows the 52 identified security threats by the Microsoft Threat Modeling Tool for the application. Spoofing is the most prevalent category with 29% and 15 threats. This indicates a high risk of impersonating users or devices within the system, for example, using their username or password (Jawad et al., 2025). Spoofing appears as a dominant category primarily due to the characteristics of the architecture of the observed application. The application contains multiple external entry points, i.e., external frontend and publicly available services, as well as a large number of calls between services that cross trust boundaries that rely on forwarded identities or tokens, which leads to MTMT generating a larger number of Spoofing threats related to possible user or microservice impersonation.

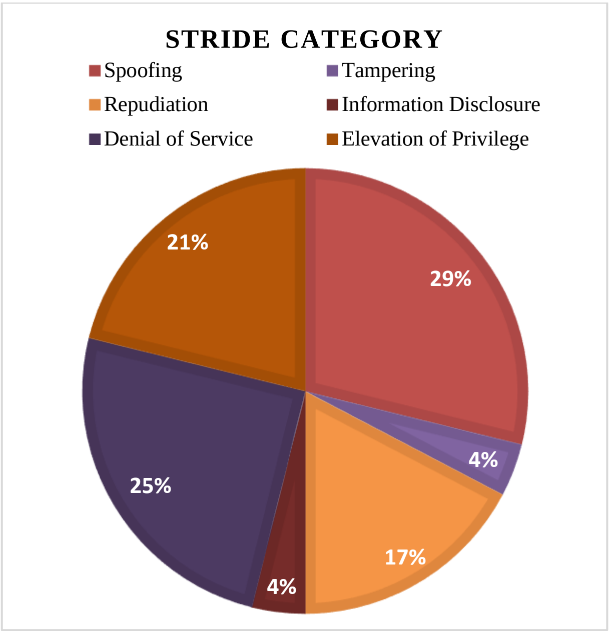

Figure 4: Distribution of the 52 identified security threats across STRIDE categories in the analyzed microservice application

After that, the next most frequent category is Denial of Service with 25% and 13 detected threats, which refers to denial of service attacks that deny service to valid users (for example, by making a web server temporarily unavailable or unusable) (Microsoft, 2022). The tool identified 11 threats for the Elevation of Privilege category, which accounts for 21% of the total number expressed as a percentage. This means that there is a risk of an attacker gaining powers greater than those originally granted, which would mean that the user must not independently elevate their privileges to a higher level (Abi-Antoun & Barnes, 2010; Khan, 2017). The next category in terms of the number of threats that can affect this application is Repudiation, with a 17% risk of this type of threat, which is a total of 9 detected threats. Repudiation represents a category of threats according to STRIDE, where there is a denial of action or a denial of inaction by the user.

The term user means human users, services, devices, and any entity that participates in an action (e.g., sends or receives a message, calls a function) (Lubrić et al., 2019). The last threats that should not be ignored are Information Disclosure and Tampering, with 4% representation each, which makes 2 threats out of the total number of identified threats. Information disclosure represents a category where there is a risk of exposing information to unauthorized persons, which occurs when an unwanted component gains unauthorized access to information, thereby violating the confidentiality goals of the system (Idensohn & Flowerday, 2025; Rouland et al., 2021). Tampering, on the other hand, means the risk of unauthorized data or code modification (Gavrić et al., 2025). Process tampering involves changing bits in a running process. Similarly, tampering with data flow involves changing bits "on the wire", that is, during communication between two active processes.

In order to recognize threats, but also eliminate them as soon as possible, it is necessary to recognize which microservice is subject to which type of threat according to the STRIDE categorization, as well as whether there is a microservice that is not subject to any threat. That is why the tools identified threats were distributed according to STRIDE categories. Table 1 shows all nine microservices that prevail in a given application, as well as STRIDE categories to which each microservice is subject.

Table 1: STRIDE threat exposure matrix per microservice in the analyzed microservice application

Microservice Types ↓	**STRIDE Categories →**	**S**	**T**	**R**	**ID**	**DoS**	**EoP**
Frontend Service		✔	✔	✔	✔	✔	✔
Product Catalog Service							✔
Cart Service		✔				✔	
Checkout Service						✔	✔
Payment Service		✔		✔		✔	
Shipping Service		✔		✔		✔	
Currency Service		✔		✔		✔	
Recommendation Service							✔
Email Service		✔		✔		✔	

Based on Table 1, it can be concluded that no microservice is completely safe from the emergence of security threats, which indicates the risk to the functioning of the application as a whole if any service remains inadequately protected. Frontend Service is the most exposed to security threats, because the tool recognizes all categories of threats for the specified service. Payment Service, Shipping Service, Currency Service, and Recommendation Service are not as vulnerable as Frontend Service, but they also have a large number of threats in the domain of Spoofing, Tampering, and Denial of Service. For the Cart Service, security threats were identified that fall into the Spoofing and Denial of Service categories according to the STRIDE table, and for the Checkout Service, threats that fall into the Denial of Service and Elevation of Privilege categories were identified. The last but not least vulnerable to the risk of security threats are the Product Catalog Service and the Recommendation Service, which are affected by threats in the Elevation of Privilege domain by categorized from the STRIDE table. The general conclusion is that the presented application is not completely safe at any level, because no service is without identified security threats. It is important to ensure the security of each microservice, because if one microservice is not adequately protected, the entire application may be interrupted as a consequence.

DISCUSSION

The resulting threats show the identity, authorization, and communication between services that are very important in microservice systems. Spoofing and Denial of Service threats occupy the largest section, while Elevation of Privilege and Repudiation also have significant access. This type of result is in line with works that indicate that microservice architecture increases the surface that can be attacked due to a greater number of endpoints, API interfaces, and distribution services, while authentication and availability are critical points of such systems (Hannousse & Yahiouche, 2021; Yarygina & Bagge, 2018; Tapia et al., 2020; Zdun et al., 2023).

When comparing our results with STRIDE analyzes that are in a different domain, a similar pattern can be observed, i.e., threats related to identity, access control, and availability are often dominant compared to classic threats. However, the specific contribution of this paper is that it presents what a step-by-step architectural representation of the STRIDE procedure looks like, which is applied based on a real microservice application, and how the results can be summarized in the exposure matrix per microservice. In this way, the gap between general STRIDE guidelines and specific tendencies in the security of microservice architectures is bridged because the literature mostly remains at the general level of patterns, comparison of tools, etc., while this paper presents a DFD that reaches a structured threat matrix for each service that contributes to the early identification of critical attack surfaces.

Although the analysis in this paper is based on the STRIDE approach, the obtained threats can be viewed in relation to other known approaches. For example, the identified threats are Spoofing, Elevation of Privilege, and Information Disclosure, which largely match the OWASP categories that include issues related to authentication, access control, and system data confidentiality. Also, the proposed approach can be combined with frameworks such as DREAD or PASTA, in which case STRIDE serves to systematically generate threats at the architecture level, while DREAD or similar models can be used to rank threats and prioritize security measures.

IMPLICATIONS AND LIMITATIONS

The obtained results have implications for researchers, employees in the industry, and educators. For the research community, this paper presented the procedure of how to model the system at the architectural level, DFD with trust boundaries, and STRIDE analysis. This approach can be applied to other microservice systems as well. Also, there is an option to replicate the study in another domain, such as healthcare or the Internet of Things.

For employees in the industry, the important message is that uncertainty in microservice systems must be viewed as an issue related to system architecture. Based on the dominance of Spoofing and Denial of Service threats, it can be concluded that strong mechanisms related to the management of identities and access rights, such as API gateway, OAuth2, as well as the systematic introduction of measures to preserve availability at public endpoints, such as rate limit, are necessary. The threat matrix per microservice can serve teams as an architecture planning tool where each team will see which STRIDE categories are most critical for their service.

In the field of education, this study can serve as a realistic example in the teaching of software engineering and software security. Based on the use of an open-source microservice application, students are allowed to reconstruct the diagram, DFD, and STRIDE analysis by themselves, and then propose some additional protection measures or supplement the architecture by themselves through tasks. In such a way, STRIDE concepts would be connected with a concrete example from practice.

This paper has several limitations that were taken into account when analyzing the results. They are: (1) Analysis based on one accessible microservice e-commerce system. Although this system is representative and can be easily reanalyzed, other domains and software architectures may have a different threat case, so the results can not be generalized to all microservice system architectures. (2) Threats are generated automatically in the MTMT tool based on the architecture model and DFD. Therefore, the results are presented as a set of potential, model-based threats, rather than actual confirmed vulnerabilities observed through penetration testing or monitoring of systems. The obtained results depend on the accuracy and level of the models used, as well as on the set of rules that the tool applies during the threat generation process. (3) Trust boundaries in DFD are modeled at a general level. One internal trust boundary encompasses the entire microservice cluster and database, while external users and services are represented as separate actors outside the specified boundary. This level of abstraction is sufficient to illustrate the procedure and identify the main permeability areas, but in larger environments, a more detailed elaboration of trust boundaries and a more detailed distinction of different types of security zones is required.

CONCLUSION

This paper presents a systematic application of the STRIDE framework to the created DFD model of a microservice application that can identify key points at the architectural level at an early stage. The procedure that includes creating an architectural view, then DFD with clearly marked trust boundaries, systematic STRIDE identification and categorization of threats, and display of identified threats in the form of a table for each microservice in the application, proved to be practical for early prioritization without changing the business logic. This reduces the risk even before implementation and testing, and each recommendation remains traceable to the corresponding DFD element and specific data flow.

Based on the analysis of the list of threats generated by the Microsoft Threat Modeling Tool, 52 threats were obtained, with the largest share of Spoofing (29%; 15 threats) and Denial of Service (25%; 13 threats), followed by Elevation of Privilege (21%; 11 threats), Repudiation (17%; 9 threats), while Information Disclosure and Tampering recorded 4% each (2 threats each). This profile indicates that identity, authorization, and inter-service communication are the dominant attack surfaces in the analyzed architecture, and that measures should be focused precisely on these layers.

Results derived directly from the profiles of identified threats based on the distribution by DFD elements point to a set of measures that target the most common exposure points, especially flows across trust boundaries. What needs to be ensured is reliable inter-service authentication and traffic integrity with consistent checking and limiting the use of tokens, applying the principle of least privilege, and clear access rules on services and data stores, increasing resilience and availability to protect against overload, where it is necessary to standardize secret management and data encryption with regular credential checking.

When it comes to directions for further research, it is planned to introduce a quantitative risk assessment with the help of the DREAD framework. For each identified threat, the parameters Damage, Reproducibility, Exploitability, Affected Users, and Discoverability will be evaluated on a calibrated scale from 0 to 10, with clearly defined ranges. Based on the results obtained in this way, it is possible to estimate the share of risk level of each threat and the prioritization of mitigations. Results will be verified through targeted tests.

REFERENCES

Abi-Antoun, M., & Barnes, J. M. (2010). *STRIDE-based security model in Acme (CMU-ISR-10-106)* [*Technical report*]. Carnegie Mellon University, Institute for Software Research

Bygdas, E., Jaatun, L. A., Antonsen, S. B., Ringen, A., & Eiring, E. (2021). Evaluating threat modeling tools: Microsoft TMT versus OWASP Threat Dragon. *In 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)* (pp. 1–7). IEEE

Chondamrongkul, N., Sun, J., & Warren, I. (2021). Formal security analysis for software architecture design: An expressive framework to emerging architectural styles. *Science of Computer Programming, 206*, 102631. https://doi.org/10.1016/j.scico.2021.102631

Das, P., Al Asif, M. R., Jahan, S., Ahmed, K., Bui, F. M., & Khondoker, R. (2024). STRIDE-based cybersecurity threat modeling, risk assessment and treatment of an in-vehicle infotainment system. *Vehicles, 6(3)*, 1140–1163. https://doi.org/10.3390/vehicles6030054

Gavrić, N., Shalaginov, A., Andrushevich, A., Rumsch, A., & Paice, A. (2025). Enhancing security in international data spaces: A STRIDE framework approach. *Technologies, 13(1)*, 8. https://doi.org/10.3390/technologies13010008

GoogleCloudPlatform. (n.d.). *Microservices-demo (Online Boutique) [Source code repository]*. GitHub. https://github.com/GoogleCloudPlatform/microservices-demo

Hannousse, A., & Yahiouche, S. (2021). Securing microservices and microservice architectures: A systematic mapping study. *Computer Science Review*, *41*, 100415. https://doi.org/10.1016/j.cosrev.2021.100415

Hernan, S., Lambert, S., Ostwald, T., & Shostack, A. (2006). Uncover security design flaws using the STRIDE approach *[Technical article]. MSDN Magazine*, 68–75

Idensohn, C. J., & Flowerday, S. (2025). Financial insider threats: A cybersecurity STRIDE analysis. *Issues in Information Systems, 26(1)*. https://doi.org/10.48009/1_iis_108

Jawad, A., Jaskolka, J., Matrawy, A., & Ibnkahla, M. (2025). strideSEA: A STRIDE-centric Security

Evaluation Approach. *arXiv preprint*. https://doi.org/10.48550/arXiv.2503.19030

Khan, R., McLaughlin, K., Laverty, D., & Sezer, S. (2017, September). STRIDE-based threat modeling for cyber-physical systems. *In 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)* (pp. 1–6). IEEE. https://doi.org/10.1109/ISGTEurope.2017.8260283

Khan, S. A. (2017). A STRIDE model-based threat modelling using unified and-or fuzzy operator for computer network security. *International Journal of Computing and Network Technology, 5(1)*, 13–20

Luburić, N., Sladić, G., & Milosavljević, B. (2019). Examining repudiation threats using a framework for teaching security design analysis. *In International Conference on Information Society and Technology (ICIST 2019)*.

McGraw, G. (2004). Software security. *IEEE Security & Privacy, 2(2)*, 80–83. https://doi.org/10.1109/MSECP.2004.1281254

Microsoft. (2016). *Threat Modeling Tool 2016 - Getting started guide [Technical documentation]*. Microsoft. https://download.microsoft.com/download/4/F/D/4FDDEA98-4ABD-47A7-AA0E-815CE8660A76/Threat%20Modeling%20Tool%202016%20Getting%20Started%20Guide.docx

Microsoft. (2022). *Microsoft Threat Modeling Tool - Threats [Technical documentation]*. Microsoft Learn. https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool

NIST. (2016). *Guide to Data-Centric System Threat Modeling (SP 800-154, Initial Public Draft) [NIST Special Publication]*. National Institute of Standards and Technology

Oluwaferanmi, A. (2025). *Role of threat modeling and architectural risk analysis in proactively identifying and mitigating software vulnerabilities. [Manuscript posted on Research Gate]*. Link: https://www.researchgate.net/publication/393579757_Role_of_Threat_Modeling_and_Architectural_Risk_Analysis_in_Proactively_Identifying_and_Mitigating_Software_Vulnerabilities

Rouland, Q., Hamid, B., & Jaskolka, J. (2021). Specification, detection, and treatment of STRIDE threats for software components: Modeling, formal methods, and tool support. *Journal of Systems Architecture, 117*, 102073. https://doi.org/10.1016/j.sysarc.2021.102073

Shostack, A. (2014). Threat modeling: Designing for security. Wiley

Tapia, F., Mora, M. Á., Fuertes, W., Aules, H., Flores, E., & Toulkeridis, T. (2020). From monolithic systems to microservices: A comparative study of performance. *Applied Sciences, 10(17)*, 5797. https://doi.org/10.3390/app10175797

Xu, D., & Pauli, J. (2006). Threat-driven design and analysis of secure software architectures. *Journal of Information Assurance and Security, 1(3)*, 171-180.

Yarygina, T., & Bagge, A. H. (2018). Overcoming security challenges in microservice architectures. *In 2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)* (pp. 11–20). IEEE. https://doi.org/10.1109/SOSE.2018.00011

Yin, R. K. (2014). Case study research: Design and methods (5th ed.). Sage.

Zdun, U., Queval, P. J., Simhandl, G., Scandariato, R., Chakravarty, S., Jelić, M., & Jovanović, A. (2023). Microservice security metrics for secure communication, identity management, and observability. *ACM Transactions on Software Engineering and Methodology, 32(1)*, 1–34. https://doi.org/10.1145/3532183

PRIMENA *STRIDE* OKVIRA U MIKROSERVISNOJ APLIKACIJI NAMENJENOJ ELEKTRONSKOM POSLOVANJU

U ovom radu je izvršena identifikacija bezbednosnih pretnji u aplikaciji, zasnovanoj na mikroservisnoj arhitekturi primenom STRIDE okvira. Upotreba STRIDE metodologije je bila ključna za kategorizaciju bezbednosnih pretnji na osnovu šest kriterijuma: lažiranje identiteta, falsifikovanje podataka, poricanje radnje ili odgovornosti, otkrivanje informacija, uskraćivanje usluge i povećanje privilegija, ali i za procenu štetnosti i procenu statusa identifikovanih pretnji. Pored toga, istraživanje se oslanja na kreiranje dijagrama toka podataka za pomenutu aplikaciju korišćenjem Microsoft Threat Modeling Tool-a, kao i na prethodno modelovanje softverskih arhitektura u alatu draw.io. Dijagram putem draw.io alata je kreiran na osnovu prerade postojećeg primera dijagrama dostupnog na GitHub-u. Ovakav način analiziranja aplikacija doprinosi smanjenom riziku od nastanka bezbednosnih pretnji.

Ključne reči: Mikroservisna arhitektura; Bezbednost aplikacija; Ranjivost; STRIDE; Microsoft Threat Modeling Tool.