http://doi.org/10.35784/iapgos.6482

received: 09.01.2025 | revised: 05.03.2025 | accepted: 15.03.2025 | available online: 31.03.2025

CRITICAL CYBERSECURITY ASPECTS FOR IMPROVING ENTERPRISE DIGITAL INFRASTRUCTURE PROTECTION

Roman Kvyetnyy[1], Volodymyr Kotsiubynskyi[1], Serhii Husak[1], Yaroslav Movchan[1], Nataliia Dobrovolska[2], Sholpan Zhumagulova[3], Assel Aitkazina[3]

[1]Vinnitsia National Technical University, Vinnytsia, Ukraine, [2]Vinnytsia Institute of Trade and Economics of Kyiv National University of Trade and Economics, Kyiv, Ukraine, [3]Al-Farabi Kazakh National University City, Almaty, Kazakhstan

Abstract. *This article is related to the security landscape that evolves to protect the digital infrastructure of organizations. As the number of interconnected systems, cloud computing, and IoT devices increases, so do today's security risks. Hence, effective protection measures are needed. This study is meant to identify common server security risks, analyze protection methods, enhance alerting and monitoring, implement fraud prevention, handle code vulnerabilities, and provide recommendations. To prove the effectiveness of some of these proposed security measures, in-depth practical testing and simulation are done, proving that proactive security strategy is key in today's digital environment. The article addresses the broad segment of best practices, modern technologies, and strategies to protect enterprise digital infrastructure in light of the most recent cybersecurity challenges.*

Keywords: cybersecurity, Zero Trust Model, intrusion prevention, Data Privacy Policies, server security, protection methods

KRYTYCZNE ASPEKTY CYBERBEZPIECZEŃSTWA W CELU POPRAWY OCHRONY INFRASTRUKTURY CYFROWEJ PRZEDSIĘBIORSTWA

Streszczenie. *Artykuł jest związany z rozwojem bezpieczeństwa w celu ochrony cyfrowej infrastruktury organizacji. Wraz ze wzrostem liczby wzajemnie połączonych systemów, przetwarzania w chmurze i urządzeń IoT, rosną również dzisiejsze zagrożenia bezpieczeństwa. W związku z tym potrzebne są skuteczne środki ochrony. Niniejsze badanie ma na celu zidentyfikowanie typowych zagrożeń bezpieczeństwa serwerów, analizę metod ochrony, ulepszenie ostrzegania i monitorowania, wdrożenie zapobiegania oszustwom, obsługę luk w kodzie i przedstawienie zaleceń. Aby udowodnić skuteczność niektórych z proponowanych środków bezpieczeństwa, przeprowadzono dogłębne testy praktyczne i symulacje, udowadniając, że proaktywna strategia bezpieczeństwa jest kluczowa w dzisiejszym środowisku cyfrowym. Artykuł dotyczy szerokiego segmentu najlepszych praktyk, nowoczesnych technologii i strategii ochrony infrastruktury cyfrowej przedsiębiorstw w świetle najnowszych wyzwań związanych z cyberbezpieczeństwem.*

Słowa kluczowe: cyberbezpieczeństwo, model zerowego zaufania, zapobieganie włamaniom, polityka prywatności danych, bezpieczeństwo serwerów, metody ochrony

Introduction

Cybersecurity poses a big challenge to organizations in the protection of enterprise digital infrastructure. Security risks are rising together with developing connected systems, cloud computing, and providing new fraud methods.

One of the critical paradigms driving the security strategy is the Zero Trust Model, when nobody is automatically trusted on the network. Instead, it requires strict verification of all the users and devices accessing the resources in order to reduce the risk of unauthorized access and possible breaches.

In addition, data privacy policies such as the European GDPR set strict guidelines for the handling and privacy of data, further shaping the security landscape. Organizations have to ensure compliance to not only defend against outside threats but also avoid penalties and damage to the organization's reputation.

This article highlights the most recent security challenges popular services face and provides practical recommendations to implement strong security measures. Organizations could strengthen protection from the threats in the digital environment by understanding these issues and utilizing proactive strategies using our recommendations.

1. The aim and objectives of the study

Identify Top Common Server Vulnerabilities: the research will focus on the top popular server vulnerabilities, like unauthorized access, malware, malicious code attacks, sensitive data leaking, etc. It also will investigate how they affect the security of an organization's IT infrastructure.

Fraud Prevention. The study offers suggestions on how IT security departments can diminish fraudulent attacks by investing in additional fraud prevention technologies like SSRF/CSRF prevention, firewall rules, intrusion prevention systems (IPS), digital protocols and multi-factor authentication using Thrid-Party integrations (e.g. Google OAuth, Apple, Facebook) [3].

Sensitive data protection. To be in compliance with the world's general data protection standards, use different encryption or masking methods for user’s PII data.

Improving Alerting and Monitoring: the research will also suggest ways to establish a complete alerting and monitoring system to detect and report security incidents in real-time.

Effective Management of Code Vulnerability: best practices in code security management will be discussed, including code review and secure coding, and tools like automated security regression tests to detect and remediate code vulnerabilities.

Deliver Holistic Suggestions: the research will deliver some practical suggestions and best practices to assist organizations in enhancing overall server security compliance, management, and operations including technical solutions and operational processes.

2. Detailed review of the protection methods

In today's digital landscape, implementing robust protection methods is paramount to safeguarding sensitive data and preventing unauthorized access. The following recommendations and suggestions serve as key modern practices for improving the cybersecurity state and mitigating potential threats.

2.1. Fraud prevention

SSRF and CSRF prevention. Implement input validation to ensure that user-supplied data does not contain malicious content. Validate URLs in user input to prevent SSRF attacks that exploit server-side requests. Implement CSRF tokens that are tied to user sessions and are validated on each request. These tokens should be unique per session and request, making it difficult for attackers to forge requests and bypass CSRF protections. Also, you can introduce URL/Domains Whitelisting that can access your service [2].

Use Web Application Firewall (WAF). Use WAF to filter incoming traffic and block known attack patterns. For example, if you use AWS for hosting your service, configure the AWS Cloud Front to limit the amount of external requests from the same IP and detect and block SQL injection attempts. If you use Azure or Google they have their own firewalls you can use.

This work is licensed under a Creative Commons Attribution 4.0 International License.
Utwór dostępny jest na licencji Creative Commons Uznanie autorstwa 4.0 Międzynarodowe.

Implement intrusion prevention systems by adding Security Attestation or Trust Mechanism. Add additional attestation tokens to your client applications to be verified on the server side and prevent direct access from REST clients and fraud scripts [4]. Also use input sanitization to prevent Remote Code Execution. Sanitize user input in web forms to block malicious scripts and prevent XSS vulnerabilities [1]. For testing purposes introduce Penetration Testing (Pen Testing). Add penetration tests to your services to identify vulnerabilities. For example, simulate a SQL injection or DDoS attack during a penetration test to find potential weaknesses and manage them in the future.

Use digital protocols to authenticate the identity of external systems before establishing connections. Verify SSL/TLS protocols to ensure secure communication channels.

Authentication/Authorization. Ensure secure authentication and authorization mechanisms are in place to protect your endpoints and control access to sensitive data. Use strong authentication methods such as Basic or JWT authorization to verify user identities and enforce access controls. In addition, use extra security layers like Multi-Factor Authentication.

2.2. Sensitive data protection

PII data encryption. Using popular encryption methods like RSA, BLOWFISH, AES for encrypting Personally Identifiable Information (PII) data before storing is crucial for maintaining data privacy and security. Encryption ensures that even if unauthorized access occurs, the data remains unreadable without the decryption key.

PII data masking. Use masks for securing PII data in application logs. For example the part of the phone number, license plate, IBAN can be replaced with asterisks (+38067****859, AB***FG, UA123*************546). Be sure to leave some symbols at the beginning and ending of the string to simplify investigation and debugging [5].

2.3. Monitoring and alerting

Introduce Intrusion Detection Systems (IDS). Implement IDS systems to detect and respond to potential intrusions or malicious activities in real-time. IDS solutions with AI elements such as Snort, Suricata, or commercial offerings like Cisco Firepower provide alerts and insights into network threats.

Analyze User and Entity Behavior (UBA). Use UBA solutions for analyzing traffic patterns, user behavior, and entity activities to detect anomalies and potential security incidents. UBA tools like Splunk User Behavior Analytics (UBA), Exabeam, or Rapid7 InsightIDR can help identify suspicious activities and insider threats [6, 9].

Use Monitoring Tools. Utilize monitoring tools like Prometheus, Splunk, and Grafana for comprehensive visibility into your network, applications, and infrastructure. These tools offer customizable dashboards, log analysis, and alerting capabilities to proactively monitor and respond to security events. Example of some useful queries in Prometheus (parameters in red are configurable).

```
a.  High heap memory usage:
    (sum(memory_used_bytes{application="{applicationName}
    ", area="heap"}) by ( application) * 100) /
    (sum(memory_max_bytes{application="{applicationName}"
    , area="heap"}) by (application)) > 80 #80 percentage
b.  High system CPU usage:
sum(system_cpu_usage{application="{applicationName}"}) by
    (cluster) * 100 > 80  #80 percentage
c.  High HTTP error rate:
sum by (application)
    (rate(http_server_requests_seconds_count{application=
    "{applicationName}"},status=~"5.."}[5m])) > 0.1 #10
    percentage
d.  High HTTP error rate:
    ((sum by (method, uri, service, cluster)
    (increase(http_server_requests_seconds_count{applicat
    ion="{applicationName}"}[30s]))) / ((sum by (method,
    uri, service)
    (increase(http_server_requests_seconds_count{applicat
    ion="{applicationName}"}[30s] offset 1m) )))) > 30
    #30 percentage
```

Use Alert Notification Providers. Integrate with alert notification providers such as Opsgenie, PagerDuty, or other similar services to receive real-time alerts and notifications about security incidents. These providers offer escalation policies, on-call schedules, and integrations with monitoring tools for efficient incident response [7].

2.4. Automated code security tools

Include automated security tools like OWASP, Trivy Scan, CodeQL, and SonarQube into your development pipeline to scan for vulnerabilities, perform static code analysis, and identify security issues early in the development process [4].

3. Fraud attacks: causes, actions, and system reactions

The value of security theories is only as good as the degree to which they work in practice. Based on research, this behavior modeling simulates different types of fraud cases focusing on the ploys and methods that the attackers use to get into systems for unauthorized access, data theft, or disruption. The proposed security-minded changes methodically implemented to protect against these attacks will not only serve for defense in a developed space but also keep us battle-tested and ready to deal with any upcoming real-world challenges [8].

Bringing both theoretical best practices and real attack simulations, this combined method will assist you in constructing a solid and resilient server security posture to protect your precious data assets from potential cyber threats.

Table 1 (part 1 of 2). System reactions for simulated fraud actions

No	Fraud Attack	Possible Protection Measures	Fraud Actions	Protected System Reaction
1	SSRF (Server-Side Request Forgery)	Input validation for URLs URL/Domain whitelisting	Exploit vulnerabilities in applications to execute unauthorized requests on external servers. Steal data or launch further attacks from compromised servers.	Block malicious requests attempting to exploit SSRF vulnerabilities. Investigate source and update validation rules.
2	CSRF (Cross-Site Request Forgery)	CSRF tokens tied to user sessions Secure authentication (Basic/JWT)	Trick users into performing unauthorized actions in web applications through forged requests. Steal data, modify user accounts, or perform unauthorized actions	Block requests with invalid CSRF tokens. Investigate source and update authentication measures.
3	SQL Injection	Input validation and sanitization Secure database access control Regular database security audits	Inject malicious SQL code into database queries to steal or manipulate data. Gain unauthorized access to sensitive information.	Identify and block malicious queries. Restore data from backups if compromised. Patch database vulnerabilities.

Table 1 (part 2 of 2). System reactions for simulated fraud actions

No	Fraud Attack	Possible Protection Measures	Fraud Actions	Protected System Reaction
4	XSS (Cross-Site Scripting)	Input validation and sanitization Secure coding practices Web Application Firewalls (WAFs)	Inject malicious scripts into websites to steal user data or redirect them to fraudulent sites. Steal cookies, session tokens, or other sensitive information from users.	Block requests containing malicious scripts. Investigate source and update sanitization rules.
5	Log Injection	Input validation for log messages Secure logging practices Use libraries/frameworks with built-in sanitization	Inject malicious code into log messages to exploit vulnerabilities in logging systems or gain unauthorized access. Disrupt logging functionality or potentially escalate privileges	Sanitize user input before logging. Investigate source of suspicious log messages. Patch logging system vulnerabilities.
6	Denial-of-Service (DoS) Attacks	Rate limiting on network traffic Geo-blocking Web Application Firewalls (WAFs)	Overwhelm servers with traffic, making them unavailable to legitimate users. Disrupt business operations and cause financial losses.	Utilize DDoS mitigation services. Scale server resources to handle increased traffic. Analyze logs to identify attack source.
7	Account Takeover (ATO)	Strong password policies and multi-factor authentication (MFA) User activity monitoring Session timeouts	Attackers gain unauthorized access to legitimate user accounts. Steal data, perform unauthorized actions, or launch further attacks	Investigate suspicious account activity. Reset compromised account passwords. Enhance access controls and monitoring.
8	Code Security (OWASP, Trivy Scan, CodeQL, SonarQube)	Fix identified security vulnerabilities in the code. Update development practices to prevent similar vulnerabilities in the future.	Exploit vulnerabilities in application code to inject malicious code or manipulate data. Gain unauthorized access to the server or steal sensitive information.	Can't exploit vulnerabilities in application code
9	Penetration Testing (Pen Testing)	Periodic penetration testing by qualified security professionals	Simulate various attack vectors to identify vulnerabilities in your server defenses. Gain potential access to unauthorized data or functionality.	Patch identified vulnerabilities and retest to ensure effectiveness. Prioritize remediation efforts based on the severity of vulnerabilities.
10	Unauthorized API Access	Security Attestation or Trust Mechanism API key management	Exploit weak API authentication or impersonate legitimate clients to gain unauthorized access. Steal data or disrupt operations through API calls	Throw unauthorized exception from API calls
11	Data Breach	Encryption of PII data at rest and in transit Data masking for non-production environments Regular data backups	Hackers gain unauthorized access to servers and steal PII data.	Limit access to PII data

To implement real-world infrastructure, organizations must prioritize a multifaceted approach that combines proactive security measures, regular assessments, and robust incident response strategies. This will help create a solid foundation for effective protection against ever-changing cyber threats. By continuously adapting to new challenges and evolving security risks, organizations can ensure their systems remain resilient and secure over time.

4. Practical research and observations results

While it is truly impossible to protect against all types of cyber threats, the authors suggest maximize enterprise IT systems defense by adopting this comprehensive approach.

As the part of this research, the authors suggested the following approach on how the fraud attacks and intrusion prevention should be practically handled, based on the data analytics done by the system. The diagram below (Fig 1.) reflects the data collection process via the usage of Logging and Monitoring Services, with the further processing by Audit Service. The data collected by the mentioned above services, are processed by the Analytics Service that initiates the usage of respective vulnerabilities prevention practices and analysis techniques.

The system that operates according to the flow diagram above consists of the following distinct services below:

1. User Management Service: Handles user registration, authentication, and account management.
2. OTP Generation Service: Generates secure one-time passwords.
3. OTP Delivery Service: Sends OTPs via SMS, email, or other channels.
4. Request Validation Service: Validates incoming requests and applies security checks.
5. Session Management Service: Manages user sessions and token lifecycle.
6. Logging and Monitoring Service: Tracks service logs, errors, and operational metrics.
7. Audit Service: Records user actions and critical events for compliance and auditing.
8. Notification Service: Sends notifications related to OTP expiry, usage alerts, etc.
9. Analytics Service: Provides insights into OTP usage trends and performance metrics.
10. Admin Dashboard Service: Offers an interface for admins to monitor and manage the system.

System Details:

- Number of Microservices: 10,
- Average HTTP Requests per Second (per service): ~180,
- Total Average HTTP Requests per Second: ~1,800,
- Observation Time: 24 hours,
- AWS Instance Type: Each service runs on a separate t3.medium instance,
- Java Version: 17 (consistent across all services),
- Framework: Spring Boot, version 3.3.3.RELEASE (used for all microservices),
- Database: PostgreSQL (each service may have its own schema, depending on its responsibility).

Additional Statistics:

- Peak Requests per Second (per service): ~250,
- Total Peak Requests per Second: ~2,500,
- Average CPU Utilization (per instance): ~50%,
- Average Memory Usage (per instance): ~2.5 GB,
- Total CPU Utilization: ~500% across all instances.

A combination of used methods:

- Using Firewall rules and SSRF/CSRF protection,
- Input Sanity and Trust Mechanism,
- Prenetration testing,

for input service configuration:

- Service Responsibility: One time password authentication,
- Average HTTP requests per second: ~180,
- AWS Instance: t3.medium,
- Java 17,
- Spring Boot Framework, version 3.3.3.RELEASE,
- Database: PostgresSQL.

The real testing was performed with a different combinations of the practices and results below demonstrate the substantial increase in system security comparatively to the systems using limited amount of defense practices.

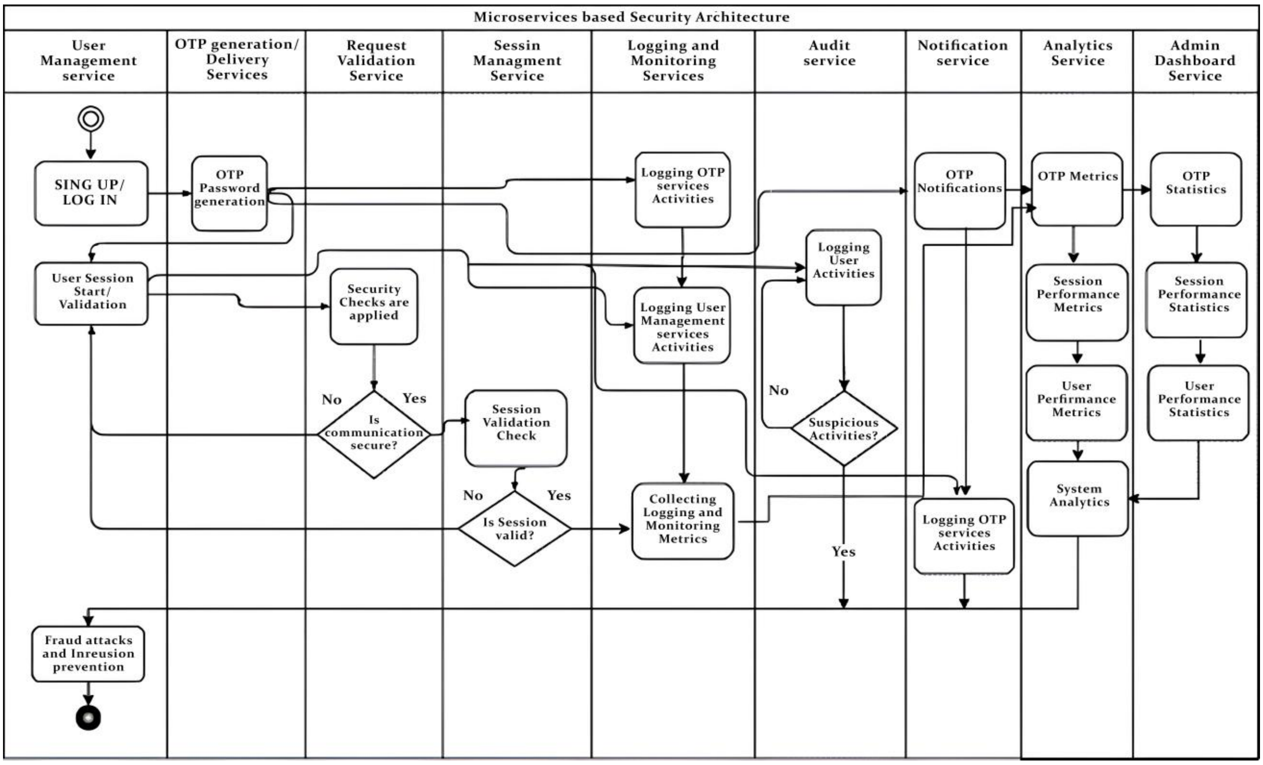
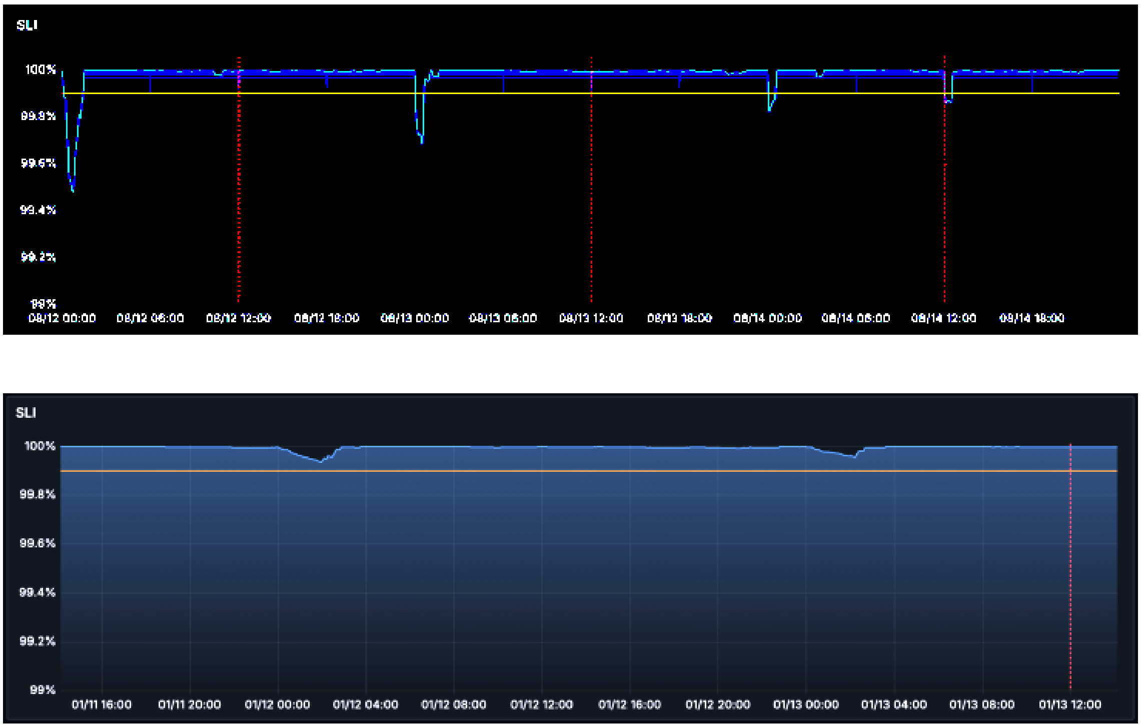

Microservices based Security Architecture
User Management service
OTP generation/ Delivery Services
Request Validation Service
Sessin Managment Service
Logging and Monitoring Services
Audit service
Notification service
Analytics Service
Admin Dashboard Service
SING UP/ LOG IN
OTP Password generation
Logging OTP services Activities
OTP Notifications
OTP Metrics
OTP Statistics
Logging User Activities
User Session Start/ Validation
Security Checks are applied
Logging User Management services Activities
Session Performance Metrics
Session Performance Statistics
No
Yes
Is communication secure?
Session Validation Check
No
Suspicious Activities?
User Perfirmance Metrics
User Performance Statistics
No
Yes
Is Session valid?
Collecting Logging and Monitoring Metrics
Logging OTP services Activities
Yes
System Analytics
Fraud attacks and Inreusion prevention

Fig 1. UML swimlane diagram of the main data-collection/processing flows

Before:



After:

Fig. 2. Performance difference between services before and after applying

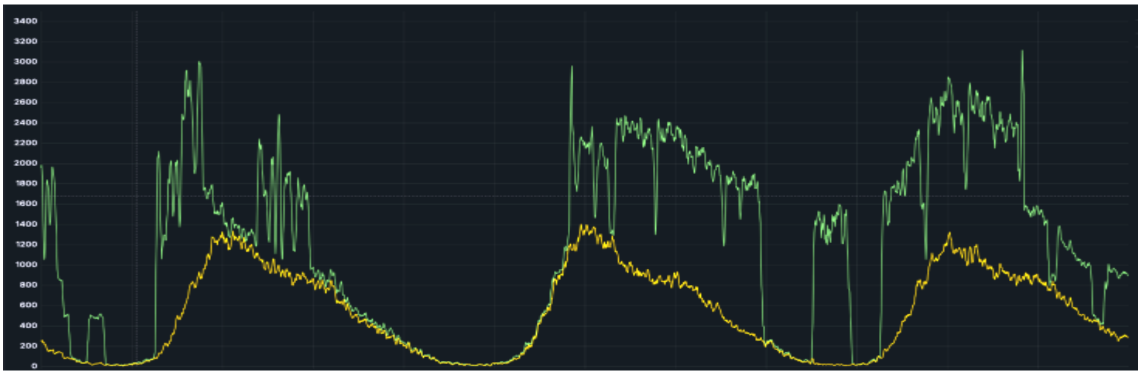


Fig. 3. Significant decrease in the number of requests

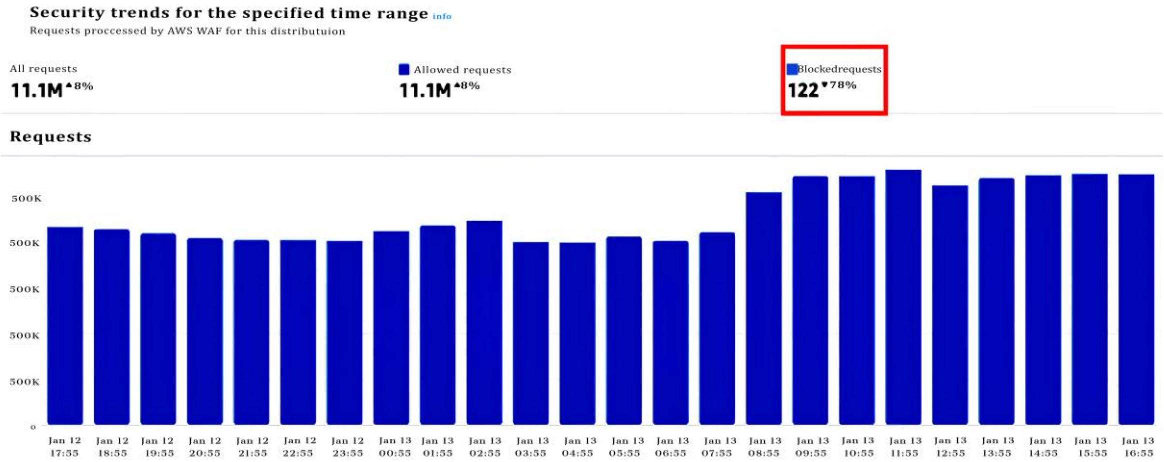


Fig. 4. Prevented fraud attacks

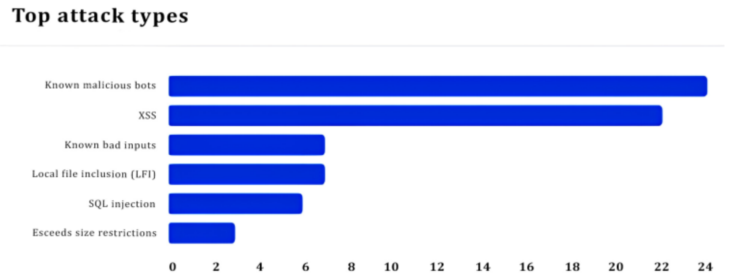


Fig. 5. Result of Cloud Front Firewall blocked 122

SLI increase

On the graphs (Fig. 2) you could see the performance difference between services before and after applying the proposed approach, we observed an increase in the uptime of our microservices.

Reduced the number of fraud requests

After enabling Input Sanity and Trust Mechanism, we observed a significant decrease in the number of requests (Fig. 3). The green line represents the request count before enabling these measures, while the yellow line shows the count after activation.

Prevented fraud attacks

Cloud Front Firewall blocked 122 fraud requests by the following reasons (Fig. 5).

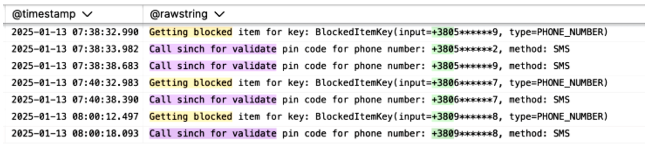
Masked PII data

Here we can see masked PII data that protects user data from unauthorized access.

Scanned Code Vulnerabilities

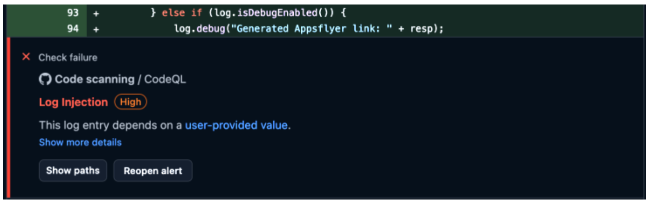
Example of a report that identifies log injection in the code An example of a report that identifies log injection in the code.

An example of the recommended changes to protect the service from log injection (Fig. 8).



```
@timestamp              @rawstring
2025-01-13 07:38:32.990 Getting blocked item for key: BlockedItemKey(input=+3805******9, type=PHONE_NUMBER)
2025-01-13 07:38:33.982 Call sinch for validate pin code for phone number: +3805******2, method: SMS
2025-01-13 07:38:38.683 Call sinch for validate pin code for phone number: +3805******9, method: SMS
2025-01-13 07:40:32.983 Getting blocked item for key: BlockedItemKey(input=+3806******7, type=PHONE_NUMBER)
2025-01-13 07:40:38.390 Call sinch for validate pin code for phone number: +3806******7, method: SMS
2025-01-13 08:00:12.497 Getting blocked item for key: BlockedItemKey(input=+3809******8, type=PHONE_NUMBER)
2025-01-13 08:00:18.093 Call sinch for validate pin code for phone number: +3809******8, method: SMS
```

Fig. 6. Masked PII data



```
+        } else if (log.isDebugEnabled()) {
+            log.debug("Generated Appsflyer link: " + resp);
```

✕ Check failure

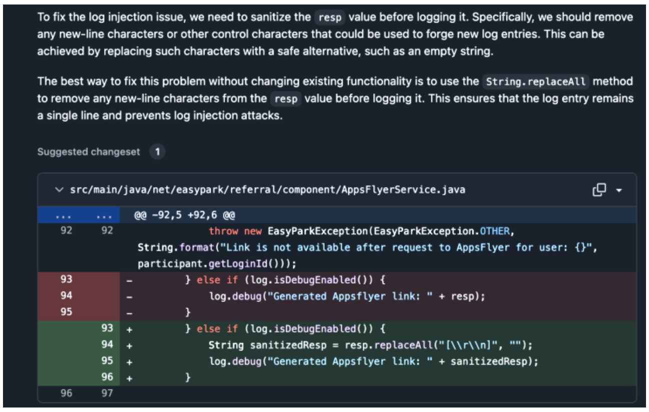
Code scanning / CodeQL

Log Injection High

This log entry depends on a user-provided value.

Show more details

Fig. 7. Example of a report that identifies log injection in the code



To fix the log injection issue, we need to sanitize the `resp` value before logging it. Specifically, we should remove any new-line characters or other control characters that could be used to forge new log entries. This can be achieved by replacing such characters with a safe alternative, such as an empty string.

The best way to fix this problem without changing existing functionality is to use the `String.replaceAll` method to remove any new-line characters from the `resp` value before logging it. This ensures that the log entry remains a single line and prevents log injection attacks.

Suggested changeset 1

```
src/main/java/net/easypark/referral/component/AppsFlyerService.java
@@ -92,5 +92,6 @@
                throw new EasyParkException(EasyParkException.OTHER,
String.format("Link is not available after request to AppsFlyer for user: {}",
participant.getLoginId()));
-           } else if (log.isDebugEnabled()) {
-               log.debug("Generated Appsflyer link: " + resp);
-           }
+           } else if (log.isDebugEnabled()) {
+               String sanitizedResp = resp.replaceAll("[\\r\\n]", "");
+               log.debug("Generated Appsflyer link: " + sanitizedResp);
+           }
```

Fig. 8. An example of the recommended changes

5. Conclusions

This research demonstrates the efficiency of a comprehensive cybersecurity strategy that integrates multi-advanced security techniques. The combination of the provided methods forms a multilayered defense framework encompassing, sensitive data protection, fraud prevention tactics, and monitoring with alert systems. Scientific and technical experiments have proven that this approach enhances an organization’s ability to identify, prevent, and respond to cyber threats greatly.

The implemented strategy was validated through rigorous testing and real-world simulations. Enterprises can achieve robust resilience by using modern technologies and adopting our best practices. This is an active multi-faceted approach that ensures the protection of digital enterprise infrastructure from various cyber risks of today’s world and saves operation stability as well as confidentiality of information.

References

[1] Amoo O. O. et al.: GDPR's impact on cybersecurity: A review focusing on USA and European practices. International Journal of Science and Research Archive 11(01), 2024, 1338–1347.

[2] Bisikalo O. et al.: Parameterization of the Stochastic Model for Evaluating Variable Small Data in the Shannon Entropy Basis. Entropy 25(2), 2023, 184.

[3] Bravo C., Toska D.: The Art of Social Engineering: Uncover the Secrets Behind the Human Dynamics in Cybersecurity. Packt Publishing, Birmingham 2023.

[4] Kelley D., Moyle E.: Practical Cybersecurity Architecture - Second Edition: A Guide to Creating and Implementing Robust Designs for Cybersecurity Architects. Packt Publishing, Birmingham 2023.

[5] Mack S. D.: The DevSecOps Playbook: Deliver Continuous Security at Speed. Wiley 2023.

[6] National Institute of Standards and Technology (NIST). Zero Trust Architecture: An Introduction, 2023 [https://www.nist.gov/cybersecurity/zero-trust].

[7] NIST National Cybersecurity Center of Excellence. Implementing a Zero Trust Architecture. NIST Cybersecurity Practice Guide [https://www.nccoe.nist.gov/sites/default/files/2024-07/zta-nist-sp-1800-35-preliminary-draft-4.pdf].

[8] Open Web Application Security Project (OWASP). (n.d.). Top Ten Web Application Security Risks [https://owasp.org/www-project-top-ten/].

[9] Security Magazine. Best Practices for Server Security, 2023 [https://www.securitymagazine.com/articles/93274-best-practices-for-server-security].

Prof. Roman Kvyetnyy
e-mail: rkvetny@sprava.net

Professor of Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University. Scientific interests include modeling of complex systems and decision-making under conditions of uncertainty (probabilistic and interval methods), modern methods of data processing. The main direction of scientific activity is development of methods and tools for mathematical modeling and information processing in computerized systems of automation and control.

https://orcid.org/0000-0002-9192-9258

PhD. Volodymyr Kotsiubynskyi
e-mail: vkotsyubinsky@gmail.com

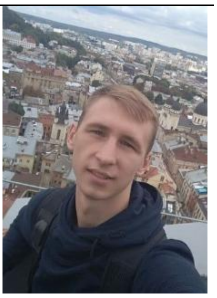
Ph.D., senior lecturer of Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University.
The main direction of scientific activity is mathematical modelling, information technologies, modelling of business processes and management of software projects.

https://orcid.org/0000-0001-6759-5078

M.Sc. Serhii Husak
e-mail: gusaksergiy710@gmail.com

Master of Science at the Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University. Interests include security systems and applications for sending notifications to users and blocking attempts at fraudulent activity.

https://orcid.org/0009-0007-4680-5487



M.Sc. Yaroslav Movchan
e-mail: yaroslavmovchan24@gmail.com

Master of Science at the Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University. Interests include modern methods of data processing, such as filtering security data, identifying extraneous objects in secure areas, and methods for improving security systems.

https://orcid.org/0009-0003-5338-4140

Ph.D. Nataliia Dobrovolska
e-mail: natali0212@ukr.net

Vinnytsia Institute of Trade and Economics of Kyiv National University of Trade and Economics, Ukraine
Interests include modern methods of data processing, such as filtering security data, identifying extraneous objects in secure areas, and methods for improving security systems in economy.

https://orcid.org/0000-0003-3444-1245

M.Sc. Sholpan Zhumagulova
e-mail: sh.zhumagulovakz@gmail.com

M.Sc., post-graduate student of Al-Farabi Kazakh National University, Almaty, Kazakhstan
Interests include modern methods of data processing, identifying extraneous objects in secure areas, and methods for improving security systems in telemedicine and bioinformatics.

https://orcid.org/0009-0006-3696-0021

M.Sc. Assel Aitkazina
e-mail: Aitkazina.aseel@gmail.com

M.Sc., post-graduate student of Al-Farabi Kazakh National University, Almaty, Kazakhstan.
Interests include modern methods of data processing, identifying extraneous objects in secure areas, and methods for improving security systems in telemedicine and bioinformatics.

https://orcid.org/0009-0005-0100-9490