A robust approach for modern digital logistics in supermarket chains

Vítor Santos
MSc. Cibersecurity
DETI, University of Aveiro
Aveiro, Portugal
vitor.mtsantos@ua.pt, NMEC 107186

João Luís
MSc. Cibersecurity
DETI, University of Aveiro
Aveiro, Portugal
jnluis@ua.pt, NMEC 107403

Ricardo Quintaneiro
MSc. Cibersecurity
DETI, University of Aveiro
Aveiro, Portugal
ricardoquintaneiro@ua.pt, NMEC 110056

***Abstract*—This paper presents a modern approach to managing and securing logistics digitally within modern supermarket chains. Our solution addresses the limitations of traditional centralized architectures by incorporating a decentralized framework that enhances both operational reliability and security. Each of the features presented is designed to minimize downtime and protect against both cyber and operational threats. To ensure data confidentiality, integrity and authenticity, the system integrates secure protocols and comprehensive monitoring tools, and proactively identifies and mitigates vulnerabilities by leveraging a secure development lifecycle, ensuring compliance with industry standards and resilience against cyber attacks. The proposed solution not only optimizes supply chain logistics but also enhances scalability and risk management for supermarket chains. This work aims to set a new standard in digital logistics, combining robust and secure software with operational efficiency to address the complex and evolving needs of modern retail supply chains.**

***Index Terms*—reliability, robustness, security, logistics, retail**

I. Introduction

When it comes to supermarket chains, having efficient and secure logistics is critical for the success of their retail operation as it allows maintaining continuity of operation, meet consumer demand and finally maximize profits. Supermarket chains often rely on sophisticated logistical software to manage these events. A logistical system that is well-functioning ensures the seamless movement of products from suppliers to warehouses, and eventually to stores, optimizes inventory management and minimizes waste due to spoilage or overstocking.

However, traditional systems often face significant challenges and suffer from reliability issues such as system crashes, inaccurate inventory tracking, from poor scalability and security risks, among others. This can lead to disruptions of the supply chain and to significant financial losses, not only from commercial operations but also from hurting the image of the brand and the perception of its customers.

This paper introduces a robust, decentralized logistics management solution tailored for supermarket chains, designed to address the aforementioned critical challenges of reliability, scalability and security. Our system architecture emphasizes redundancy which means scalability through distributed computing, minimizing single points of failure and ensuring high availability even under heavy traffic or system stress.

Our motivation behind our proposal is simple. Traditional systems in supermarket chains are typically centralized, which poses several risks and challenges:

- Single points of failure can bring down the entire network, resulting in widespread operational downtime. This could manifest as inventory mismatches, delayed order processing or even system crashes during peak usage times, such as holidays or promotional sales.
- Security vulnerabilities in these systems expose them to potential major cyberattacks, including data breaches, tampering with orders or even complete system sabotage.
- Existing systems often struggle with scalability, unable to handle the increased load as supermarket chains grow, adding more store locations and warehouses to the network.

In addition, logistical software must accurately track and manage perishable goods to prevent waste and ensure legal compliance. Many current systems fail to track expiration dates accurately, leading to spoilage, customer dissatisfaction and potential legal ramifications for selling expired goods. Furthermore, the lack of real-time error detection and handling exacerbates inventory management issues, with miscounted or misplaced items going unnoticed until they create larger problems.

Our decentralized supermarket logistics system addresses the limitations of traditional centralized architectures by introducing several key features designed to improve operational reliability and security such as automated backup and recovery, dynamic load balancing, performance monitoring and error detection, end-to-end encryption for data transfers and policies to enforce security audits. These features coupled with real-time inventory monitoring and tracking of goods ensure that stock levels are accurately maintained and that perishable items are managed efficiently throughout the supply chain.

To achieve this it incorporates a secure development lifecycle. Each component of the system undergoes rigorous testing, including vulnerability assessments and penetration testing, meaning that potential attack vectors are identified and mitigated before they are integrated into the system. Additionally, role-based access control and state of the art encryption are integrated to protect sensitive data such as

order details, inventory levels and supplier contracts from unauthorized access, ensuring a secure and robust operational environment.

The solution will also adhere to data protection standards such as GDPR for consumer privacy. Integrity is maintained by using hash-based validation, ensuring that no tampering occurs during communications. Accountability and non-repudiation are enforced through digital signatures and logging mechanisms that provide a verifiable audit trail, ensuring that all transactions and system interactions can be traced and verified if needed, supporting both regulatory compliance and operational transparency.

By implementing these advanced features in such a way, our proposed system provides a resilient and secure platform for managing supermarket logistics, reducing operational risks and ensuring continuous product availability.

II. Secure Life Cycle Description

This section outlines the secure development life cycle that will be implemented in our solution to ensure compliance with security requirements.

A. Education and Awareness - Provide Training

Security is a shared responsibility across the entire organization. It is crucial that all employees, regardless of their role, possess a foundational understanding of how to apply appropriate security measures within the scope of their work. Effective training equips the team with the knowledge needed to incorporate security into software development, ensuring that the product meets both quality and security standards while addressing business requirements and delivering value to users. While not everyone may be a security specialist, it is essential to grasp the attacker's perspective and objectives to take proactive steps that prevent potential threats.

This training should be done in both the technical and conceptual aspects, through workshops and presentations, focusing on different security scope concepts such as risk assessment, use of patterns and good practices (e.g. OWASP Cheat Sheet Series [1]), and also non-functional requirements. Also, all developers, testers, and program managers must complete at least one security training class each year.

In our system in particular, reliability and availability are very important aspects since a proactive approach to these concepts allows us to reduce the risk of system failures or computer attacks, especially during the holiday season or operational peaks, where the economic impact on a company of this kind would be greatest.

B. Project Inception - Use approved tools

Establishing a list of approved tools is not only a secure practice, but also required for ISO/IEC 27001 compliance, an international standard for information security management.

Tools will be selected for the approved tool list using the following criteria:

1) The tool's author must have a positive reputation in security.
2) The tool must not possess any known high or critical severity vulnerabilities, as classified by the CVSS v4.0 [2].
3) The tool must be currently supported by its author.
4) The tool must have a compatible license with developed software.
5) The tool undergoes regular security testing.

Ensuring tools are valid for approval will involve using research and news resources like Snyk [3], NVD [4], the CVE program [5], CSO online [6], Dark Reading [7], repository host sites and other tool author resources to verify regular security checking and compliance.

C. Analysis and Requirements - Perform threat modeling

Before identifying and assessing the potential threats and risks on their own, business functionality and structure must be minimally outlined.

1) Actors: During the stakeholder interview process, actors will be outlined, not just limited to human system users, but also external company systems that should be integrated. Sales system and identity provision systems are some of the expected external components.

The product owner should take special note of logistical roles in the company, as well as existing processes and systems.

2) Use scenarios: Use scenarios should describe the observed and expected workflows actors will perform with the system. Besides interviewing company managers, the product owner should also interview workforce employees in the company's operational logistics level to not only understand workflow details, but also concerns and problems with current systems and approaches.

Use scenarios will be documented as sequences of steps and illustrated with flowcharts.

3) Use cases: Use scenarios are analyzed to derive use cases. Some scenarios will be complex, reflecting the many variables contributing and influencing modern logistic chains, requiring some scenarios to be decomposed and bundled as alternate flows to formulate use cases.

Use cases will be described and cataloged using the format displayed on Table I.

TABLE I
USE CASE FORMAT

Use case name	
Use case number	
System	
Stakeholders/actors	
Use case goal	
Primary actor	
Preconditions	
Basic flow	
Alternative flows	

Use case relationships will also be illustrated using UML Use Case diagrams [8] for a concise overview of functionality.

4) Assets: Assets are defined as an object of value in a system that needs protection. Assets will be identified and prioritized with stakeholder interviews and research on strategic assets in the supermarket industry. The highest priority assets are expected to be mission-critical and the most sensitive to disruptions and attacks.

These high-level assets will drive threat prioritization as their compromise would cascade into a near-total communications breakdown on the logistics chain.

5) Architecture overview: In the context of threat modeling, a technical in-depth low-level architecture report is not necessary as threats on this level are implementation dependent and detected during development. This is why the architecture overview will be modeled using a variant of Level 2 DFD (Data Flow Diagram) with some explicit architecture elements (databases, caches, message brokers, applications, etc...) [9]. Level 2 DFDs provide adequate balance between simplicity and detail, exposing enough information to understand attack surfaces and security properties.

6) Threat Identification: In practice, threat identification is done via a meeting with all team members. Microsoft SDL recommends this meeting to last 2 hours: the first hour to reach a common understanding of system functionality and use scenarios, the second hour to identify threats, mitigations and risks [9].

Based on OWASP recommendations, the STRIDE methodology will be used to enumerate a range of basic potential threats [10].

Advanced threat identification requires offensive thinking, i.e. thinking like an attacker. There's no single definitive methodology for this [9], so all team members are invited to an exercise in imagining scenarios where security assumptions are challenged and security controls fail.

At this stage, threat identification will require technical knowledge of common system weaknesses, vulnerabilities and attacks.

D. Architectural and Detailed Design - Establish design requirements

Thinking about design and architecture in a secure way from the start of the system's development makes it possible to reduce the risk of future problems, which would be more expensive to solve than if they were thought about and mitigated at an early stage. Following this ideas, we can use Saltzer and Schroeder's [11] design principles to guide us in this process.

Bearing in mind that part of the system's target users are people with no technical knowledge, presenting a user-friendly interface with good usability, i.e. above 68 on the System Usability Scale (SUS) [12] and easy access to real-time data and analytics should be an aspect to consider at this stage.

It is also essential to implement an access policy, so that only those who need it are allowed to access or modify the data (also known as least privilege principle). An Role-Based Access Control (RBAC) model can be used for this, granting separation of privileges, in which the training programs mentioned previously help to understand the risks associated with each role.

A system of this kind must have an elastic infrastructure, i.e. the ability to scale both horizontally and vertically, since the integration of new shops, warehouses or suppliers must not affect the system's overall performance. Both this infrastructure and authentication should apply the open design principle, i.e. avoiding obscurity and applying transparent security protocols.

The system should enforce complete mediation by validating each access request dynamically. Instead of assuming that prior access authorization suffices, the system checks permissions each time a user accesses or modifies data, ensuring that any change in user privileges or security policies is applied immediately.

In architectural terms, to achieve a decentralized architecture, a cloud solution can be used which, as well as offering more redundancy, leverages flexibility and scalability. By distributing resources across multiple regions or availability zones within the cloud, the system avoids relying on a single point of failure, which also aligns with the principle of least common mechanism by minimizing shared dependencies among different parts of the system.

Both the architecture and other decisions should be well documented from the start of the project, so that both future participants and stakeholders understand all the decisions made up to a certain point.

E. Implementation and Testing - Perform dynamic analysis security (DAST)

The implementation phase of the supermarket logistics system will focus on adhering to the secure development practices outlined in the earlier stages. All code will be developed within approved IDEs, applying secure coding practices. The system will be containerized using Docker and orchestrated with Kubernetes to ensure consistent and scalable deployments across different environments. Secure development practices will be enforced through Git repositories hosted on GitHub or GitLab, with Jenkins automating the continuous integration/continuous deployment (CI/CD) pipelines. During the CI/CD process, security tools like OWASP ZAP and Nikto will be utilized to detect vulnerabilities in the web applications.

As the system is implemented, comprehensive DAST scans will be performed as part of the integration and system testing stages. Integration testing will validate the interactions between modules, ensuring that the system functions cohesively, while DAST will assess the system's behavior under various conditions, allowing us to identify vulnerabilities that might arise due to external interactions. Performance tests will simulate high-traffic scenarios to evaluate the system's ability to handle operational peaks, while DAST will continue to detect potential security flaws in real time as the system processes high volumes of requests.

To ensure comprehensive security and monitoring, Security Information and Event Management (SIEM) tools like Wazuh will be integrated. Wazuh will facilitate real-time monitoring

of security events and ensure that any anomalies or potential threats are quickly identified and addressed. Additionally, Prometheus will be employed for system performance monitoring and alerting, enabling the team to detect performance bottlenecks or failures during operational peaks.

Following implementation, DAST results will be reviewed and prioritized for mitigation, ensuring that all critical vulnerabilities are addressed before the next phase. Finally, a formal User Acceptance Testing (UAT) phase will be conducted, where actual end-users, such as local inventory managers and administrators, interact with the system to ensure it meets their needs, functions as expected and delivers a user-friendly experience.

F. Release, Deployment, and Support - Establish a Standard Incident Response process Process

The team will develop and maintain a comprehensive Incident Response Plan (IRP) that includes the roles, responsibilities, and communication protocols for all team members involved in incident response. Regular training sessions and exercises will be conducted to familiarize the incident response team with simulated attack scenarios, ensuring prompt and effective responses. Additionally, monitoring tools such as Wazuh Security Information and Event Management (SIEM) and Prometheus will be configured to provide real-time alerts for any unusual activity, enabling early detection of potential security threats.

In the event of an incident, detection and analysis will be conducted to assess the nature and scope of the incident. Alerts from Wazuh and Prometheus will be reviewed to determine the type, severity and potential impact of the event. An assigned incident responder will document all relevant details, including timestamps, affected systems and preliminary analysis of the incident's origin. Log files, network traffic data and attack-defense trees will be examined to identify patterns and determine the most effective response approach.

Once the incident is confirmed, containment measures are implemented to isolate the affected systems and prevent the incident from spreading. The containment strategy may involve short-term containment (such as isolating affected nodes from the network) and long-term containment (such as deploying patches or reconfiguring network parameters). Access controls will be reviewed and adjusted as necessary, ensuring that only authorized personnel can access or modify affected systems. Any temporary fixes applied during containment will be documented and tested to avoid unintended consequences on overall system operations.

After containment, the next process is eradication and that involves identifying and eliminating the root cause of the incident, such as removing malware, closing vulnerable ports or applying necessary patches. During recovery, systems will be carefully restored to normal operation in a controlled manner. This may involve restoring data from backups, revalidating the integrity of affected components and gradually reintegrating them into production. Continuous monitoring will be maintained to confirm that the incident is fully resolved and that no residual vulnerabilities remain.

Following successful containment and recovery, a post-incident review will be conducted to document and analyze the incident response. This includes reviewing all actions taken, assessing the effectiveness of the response and identifying areas for improvement. The insights gained will be incorporated into the Incident Response Plan and may lead to updated security measures, enhanced monitoring rules or new training programs. Lessons learned will also be shared with relevant stakeholders to reinforce a proactive security culture across the organization.

III. Solution Description

IV. Security Requirements

V. Conclusion

References

[1] J. Manico, J. Maćkowski, K. W. Wall, and S. Z. Heigh, "Owasp cheat sheet series," accessed 25-October-2024. [Online]. Available: https://owasp.org/www-project-cheat-sheets/

[2] "Cvss v4.0 specification document," June 2024, accessed 24-October-2024. [Online]. Available: https://www.first.org/cvss/v4.0/specification-document

[3] "Snyk software supply chain security," accessed 24-October-2024. [Online]. Available: https://snyk.io/pt-BR/solutions/software-supply-chain-security/

[4] "National vulnerability database," accessed 24-October-2024. [Online]. Available: https://nvd.nist.gov/

[5] "Cve® program website homepage," accessed 24-October-2024. [Online]. Available: https://www.cve.org/

[6] "Cso online homepage," accessed 24-October-2024. [Online]. Available: https://www.csoonline.com/

[7] "Dark reading homepage," accessed 24-October-2024. [Online]. Available: https://www.darkreading.com/

[8] K. Fakhroutdinov, "Uml use case diagrams," Jan 2014, accessed 25-October-2024. [Online]. Available: https://www.uml-diagrams.org/use-case-diagrams.html

[9] "Perform secure design review and threat modeling," accessed 25-October-2024. [Online]. Available: https://www.microsoft.com/en-us/securityengineering/sdl/practices/secure-by-design

[10] L. Conklin, V. Drake, S. Strittmatter, Z. Braiterman, and A. Shostack, "Owasp threat modeling process," accessed 25-October-2024. [Online]. Available: https://owasp.org/www-community/Threat_Modeling_Process

[11] M. Mardjan, "Saltzer and schroeder's design principles," Jun 2024. [Online]. Available: https://nocomplexity.com/documents/securityarchitecture/architecture/saltzer_designprinciples.html

[12] N. Thomas, "How to use the system usability scale (sus) to evaluate the usability of your website," 2019, [Online; accessed 23-October-2024]. [Online]. Available: https://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/

[13] Microsoft, "Security training," [Online; accessed 22-October-2024]. [Online]. Available: https://www.microsoft.com/en-us/securityengineering/sdl/practices/security-training

[14] Microsoft, May 2012, [Online; accessed 22-October-2024]. [Online]. Available: https://learn.microsoft.com/en-us/previous-versions/windows/desktop/cc307407(v=msdn.10)

[15] WKRC, "Walmart charged wrong prices for days due to internal system failure, 1600 stores affected," accessed 25-October-2024. [Online]. Available: https://www.abc3340.com/news/nation-world/walmart-charged-wrong-prices-internal-system-failure-1600-stores-affected-cincinnati-consumer-alerts-hundreds-of-stores-affected-price-point-scanning-certain-items-largest-retailer-fix-issue-documents-acknowledge-breakdown-computer-net-information-detail

[16] S. Moss, "It outage at uk supermarket sainsbury's cancels "vast majority" of online orders," Mar 2024, accessed 25-October-2024. [Online]. Available: https://www.datacenterdynamics.com/en/news/it-outage-at-uk-supermarket-sainsburys-cancels-vast-majority-of-online-orders/

[17] M. Boyle, “A $50 million glitch? target takes a hit from register outage,” May 2024, accessed 25-October-2024. [Online]. Available: https://www.datacenterknowledge.com/outages/a-50-million-glitch-target-takes-a-hit-from-register-outage