OWASP TOP 5 WEB APPLICATION SECURITY RISKS (2025)

by

Ameen ul Islam

Abstract

With the rapid expansion of digital services, modern web applications face increasingly sophisticated security threats that exploit weaknesses in design, configuration, and dependency management. To provide a focused and updated understanding of the most critical risks, the Open Web Application Security Project (OWASP) released its 2025 edition of the OWASP Top 10. This seminar concentrates on the top five vulnerabilities of the 2025 update— Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection. These risks collectively represent major root causes behind real-world data breaches, unauthorized system access, dependency-based compromises, and privacy violations. As organizations increasingly depend on interconnected systems, third-party packages, and distributed architectures, understanding these vulnerabilities becomes crucial for creating secure and resilient applications that can withstand evolving threat landscapes.

This study examines each vulnerability in depth, analysing how misconfigured systems, insecure dependency chains, weak or outdated cryptographic practices, flawed access control mechanisms, and improper handling of user input can lead to severe exploitation. The abstract highlights the rising threat of supply chain attacks, such as compromised open-source libraries and malicious updates, which now rank among the most impactful issues due to the widespread use of external components. Alongside identifying failure points, the report outlines essential mitigation strategies including strict access control validation, secure configuration management, dependency integrity verification, encryption best practices, and the use of parameterized queries or runtime validation to prevent injection attacks. By exploring the OWASP Top 5 (2025) in a structured manner, this seminar underscores the importance of adopting a security-first development approach and integrating defensive controls throughout the software lifecycle to significantly reduce risk and strengthen overall application security posture.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

The OWASP Top 10 is a globally recognized standard that highlights the most critical security risks affecting modern web applications, helping developers, organizations, and security professionals understand and mitigate threats that frequently appear in real-world environments. The 2025 update reflects significant shifts in the cybersecurity landscape, emphasizing vulnerabilities that have become increasingly common due to cloud adoption, microservices architecture, widespread open-source usage, and complex application ecosystems. The Top 5 risks— Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures and Injection— represent foundational security weaknesses that attackers routinely exploit to gain unauthorized access, manipulate system behavior, steal sensitive data, or compromise entire application infrastructures. Broken Access Control continues to lead the list due to frequent lapses in enforcing user privileges, while Security Misconfiguration rises in importance as organizations struggle with mismanaged cloud settings, default credentials, and exposed services. The new inclusion of

Software Supply Chain Failures in the top ranks reflects the growing reliance on external dependencies and the sharp increase in attacks targeting package managers, libraries, and build pipelines. Cryptographic Failures persist as a major issue due to improper key management, weak algorithms, or insecure data transmission, and Injection remains a long-standing threat resulting from improper handling of untrusted input. Together, these vulnerabilities form the core challenges that must be addressed to design, build, and maintain secure applications capable of resisting modern attack techniques.

1.2 Problem Statement

As web applications continue to grow in complexity and scale, they increasingly rely on interconnected services, third-party components, and distributed architectures that introduce new security challenges. Despite the availability of secure development guidelines, many applications still suffer from fundamental weaknesses such as Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection. These vulnerabilities lead to unauthorized access, data breaches, integrity loss, and large-scale exploitation, affecting both organizations and end-users. The core problem is that developers, administrators, and system architects often lack awareness, proper configuration practices, secure dependency management, and robust input handling mechanisms necessary to defend modern systems. As a result, applications remain exposed to preventable threats that attackers can easily exploit. Therefore, there is a critical need to examine these top vulnerabilities in depth, understand how they arise, and identify practical mitigation strategies that can be integrated into the development and deployment lifecycle to ensure secure, reliable, and resilient web applications.

1.3 Need of the Study

The rapid evolution of web technologies, combined with the widespread adoption of cloud platforms, microservices, APIs, and third-party libraries, has significantly increased the exposure of modern applications to sophisticated security risks. While these advancements provide scalability and operational efficiency, they also introduce complex attack surfaces that organizations must continuously protect. Despite improvements in development frameworks, automated testing tools, and security scanning solutions, many applications still exhibit recurring and preventable vulnerabilities. Issues such as Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection continue to dominate web security reports and breach analyses year after year. Their persistence indicates that technical solutions alone are insufficient; underlying gaps in secure development practices, architectural design decisions, and operational oversight remain unresolved.

Studying these vulnerabilities is critical because they represent the most consistently exploited weaknesses across diverse industries, including finance, healthcare, e-commerce, and government systems. Real-world incidents show that attackers often target predictable flaws caused by improper configuration, weak authorization logic, unsafe handling of user input, and insecure dependency usage. These weaknesses regularly lead to severe consequences such as large-scale data breaches, unauthorized access to sensitive information, financial losses, service outages, compliance violations, and long-term reputational damage. Understanding the root causes and exploitation patterns of these vulnerabilities enables organizations to identify where their systems are most fragile and which areas require prioritized defensive measures.

Moreover, the increased reliance on third-party components, open-source ecosystems, and automated CI/CD pipelines has created new avenues for attacks that traditional security models struggle to address. Software supply chain attacks, in particular, demonstrate how vulnerabilities can be introduced indirectly through trusted external packages or compromised build processes. Meanwhile, misconfigurations in cloud environments, insecure encryption practices, and inconsistent implementation of access controls highlight the need for stronger governance and security awareness within development and operations teams. By closely examining the OWASP Top 5 risks, this study contributes to building a deeper understanding of these emerging challenges and the strategies needed to counter them.

Ultimately, the need for this study lies in strengthening the security posture of modern web applications by bridging the gap between theoretical vulnerability classifications and their practical implications. A focused exploration of the OWASP Top 5 enables developers, administrators, and security professionals to adopt robust secure coding habits, improve configuration management, and integrate security into every phase of the software lifecycle. This ensures that applications are not only functionally reliable but also resilient against evolving cyber threats. Through this study, teams gain the knowledge required to align with global security standards, reduce the likelihood of exploitation, and foster a more security-conscious development environment.

1.4 Practical Implications

Understanding the major vulnerabilities highlighted in the OWASP Top 5 has direct practical relevance for organizations, developers, and security teams working

on modern web applications. Weaknesses such as Broken Access Control and Security Misconfiguration often arise from everyday development and deployment decisions, meaning that even small oversights can expose critical functionalities or sensitive data. For example, improperly configured cloud storage, unrestricted API endpoints, missing role validations, or default server settings can unintentionally open pathways for attackers. By studying how these issues occur, teams are better equipped to implement systematic access control checks, follow secure configuration baselines, enforce least-privilege principles, and conduct thorough reviews before releasing applications into production. This knowledge also helps reduce the attack surface, strengthen compliance with legal and industry standards, and improve the overall trustworthiness of digital services.

Similarly, the practical impact of Software Supply Chain Failures, Cryptographic Failures, and Injection vulnerabilities is increasingly evident in real-world incidents. Organizations frequently rely on open-source components, package managers, and automated build pipelines, which, if not monitored, can introduce malicious dependencies or outdated libraries. Understanding these risks encourages the adoption of practices such as dependency integrity verification, regular patching, and secure CI/CD workflows. Cryptographic Failures highlight the need for proper key management, use of strong encryption, and secure data transmission protocols to prevent confidentiality breaches. Injection vulnerabilities, despite being well-known, continue to compromise systems through unsafe handling of user input; studying them reinforces the importance of parameterized queries, validation, and sanitization routines. By recognizing how these threats manifest in real operational environments, teams can integrate practical security controls at every stage of development, reducing the likelihood of breaches and strengthening long-term resilience.

1.5 Study Objectives

The primary objective of this study is to examine the most critical security risks affecting modern web applications as identified in the OWASP Top 5. By focusing on vulnerabilities, this study aims to understand how these weaknesses arise, how they are exploited, and what measures can effectively prevent them. The goal is to enhance awareness, strengthen secure development practices, and support the creation of robust, resilient applications capable of defending against evolving cyber threats.

- To identify and understand the most critical web application security risks highlighted in the OWASP Top 5.
- To explore the root causes behind these weaknesses in terms of design flaws, configuration issues, dependency management, and coding practices.
- To evaluate effective mitigation techniques, secure development practices, and security controls that can reduce or eliminate these risks.
- To promote awareness among developers, administrators, and organizations about the importance of secure architecture, code quality, and proper configuration.
- To provide actionable insights that support the development of secure, reliable, and resilient web applications aligned with industry standards.

1.6 Scope of the Study

The scope of this study is centered on analysing the top five security risks identified in the OWASP 2025 update, namely Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection. These five categories were chosen because they represent the most frequently exploited and most impactful vulnerabilities affecting modern web applications across industries. The study focuses on examining their definitions, technical characteristics, underlying causes, and how they manifest in real operational environments. By limiting the analysis to the top five risks, the study maintains a clear and focused direction, allowing for a deeper exploration of the vulnerabilities that consistently contribute to major security incidents.

Within this defined scope, the study assesses both the offensive and defensive aspects of each vulnerability. This includes understanding common exploitation methods observed in real-world attacks, examining the weaknesses in application design or configuration that make these vulnerabilities possible, and exploring the practical mitigation strategies that organizations can adopt. The assessment covers secure coding practices, configuration standards, dependency management approaches, encryption best practices, and continuous monitoring techniques that help reduce exposure to these risks. The study also incorporates insights from academic research and industry reports to provide a balanced and evidence-based analysis of how these vulnerabilities arise and how they can be effectively addressed.

At the same time, the scope of the study intentionally excludes certain areas to maintain clarity and relevance. It does not delve into low-level cryptographic algorithms, detailed penetration testing procedures, or the broader OWASP Top 10 categories beyond the selected five. The study does not attempt to simulate real attacks or

conduct vulnerability scanning on live systems; instead, it emphasizes conceptual understanding supported by research-driven examples. This scoped approach ensures that developers, students, and security professionals can gain practical, actionable insights without being overwhelmed by overly technical or peripheral content. By clearly defining its boundaries, the study provides a structured and meaningful examination of the OWASP Top 5 and their importance in strengthening web application security.

1.7 Report Organization

This report is organized into four chapters, including the present chapter (Chapter 1), which introduces the research topic and establishes its overall context. Chapter 1 outlines the background and relevance of studying the OWASP Top 5 Web Application Security Risks, discusses the problem statement, objectives, need for the study, and clearly defines the scope and limitations of the work. This chapter also provides an overview of how the report is structured to guide the reader through the subsequent analysis.

Chapter 2 presents the Literature Review, which examines existing research related to the five selected vulnerabilities. It summarizes prior studies, identifies key findings from the academic community, and highlights research gaps that motivate the present work. Chapter 3 provides a detailed discussion of the OWASP Top 5 risks, including their characteristics, exploitation patterns, implementation aspects, visual workflow representations, and comparative analysis. The chapter concludes with the findings and discussion derived from the study. Finally, Chapter 4 presents the overall conclusions and outlines future directions for extending the study, offering insights for further research and practical improvements in web application security.

Chapter 2

Reviews of the Related Works

2.1 Introduction

The study of web application security has gained significant attention in academic research, industry reports, and global cybersecurity frameworks due to the rising number of attacks targeting online services. Numerous researchers and security organizations have analyzed common vulnerability patterns, threat behaviors, and mitigation strategies to strengthen application defenses. The OWASP Top 10, a widely recognized reference standard, has been the subject of extensive literature, with multiple studies evaluating its effectiveness, industry adoption, and relevance in addressing emerging risks. Recent works have particularly focused on challenges related to access control implementation, misconfiguration issues in cloud environments, the escalating threat of software supply chain compromises, and persistent cryptographic and injection flaws. This chapter reviews relevant research, technical papers, case studies, and security analyses that contribute to understanding these vulnerabilities. By exploring prior findings, methodologies, and recommendations,

this chapter establishes a foundation for the present study and highlights the gaps that this research aims to address.

2.2 Related Works

Research on Broken Access Control has been explored extensively due to its persistent presence in real-world applications. Anas et al. conducted a detailed survey on detection and prevention mechanisms, highlighting how improper privilege enforcement continues to be a leading cause of unauthorized access in web systems. Their study evaluates static analysis tools, dynamic testing techniques, and hybrid approaches, concluding that no single method is sufficient for comprehensive protection. The paper emphasizes the need for improved automated detection, stronger access-control policies, and continuous monitoring. This work provides a strong foundation for understanding both the technical and practical challenges associated with preventing broken access control vulnerabilities.

Studies on Security Misconfiguration focus on how configuration errors in servers, cloud platforms, APIs, and application settings often lead to severe security breaches. Loureiro et al. examine common misconfiguration patterns and classify them into a structured taxonomy, showing how default credentials, exposed administrative interfaces, improper permission settings, and misconfigured cloud services are frequently exploited. Their analysis stresses the importance of secure-by-default configurations, periodic audits, and automated configuration scanning. This paper demonstrates that most misconfigurations stem from operational oversight rather than technical limitations, making awareness and proper configuration management essential.

The growing threat of Software Supply Chain Failures has been a major focus of modern cybersecurity research. Ohm et al. present one of the most comprehensive

analyses of supply chain attacks, compiling real-world incidents involving malicious package uploads, dependency hijacking, and compromised build systems. Their work demonstrates how attackers target popular open-source ecosystems like npm, PyPI, and Maven to distribute backdoored libraries. The paper also highlights the weaknesses in current package-verification mechanisms and proposes the adoption of stronger integrity controls such as software bills of materials (SBOMs), dependency signing, and secured CI/CD workflows. This research underscores the complexity and impact of supply chain compromises in modern development environments.

Research on Cryptographic Failures continues to emphasize the critical role of proper key management and secure encryption practices. Gautam et al. provide a comprehensive study on key management systems, detailing common weaknesses such as hardcoded keys, improper storage, weak algorithm choices, and inadequate key-rotation policies. Their findings show that even well-designed encryption systems can fail if key lifecycle practices are neglected. The study recommends using hardware-backed key storage, enforcing strict key rotation, and adopting modern cryptographic standards. This paper highlights that cryptographic failures are often organizational rather than purely technical, and require disciplined security governance.

Finally, Injection vulnerabilities remain one of the most extensively studied security issues due to their long history and continued exploitation. Ross analyzes multi-source data collection and machine learning–based detection methods for identifying SQL injection attacks. By evaluating behavioral patterns in HTTP requests, payload structures, and anomaly-based indicators, the study demonstrates how machine learning can enhance traditional detection mechanisms. The research also highlights limitations such as false positives and the need for continuous model updates. This work shows that while injection prevention requires strong coding practices like parameterized queries, advanced detection systems provide an additional defense layer.

2.3 Summary

This chapter reviewed a broad range of research contributions addressing the major vulnerabilities outlined in the OWASP Top 5, providing a contextual foundation for the present study. Prior studies on Broken Access Control consistently highlighted the persistent difficulty of designing and enforcing robust authorization mechanisms in dynamic web environments. Researchers have shown that even well-structured systems often fail to validate permissions consistently, particularly across API endpoints, multi-tenant platforms, and microservices. These weaknesses allow attackers to exploit gaps through techniques such as parameter tampering, forced browsing, and privilege escalation. The literature also emphasizes the limitations of current automated detection tools, which struggle to capture contextual authorization flaws. Research on Security Misconfiguration further demonstrates that configuration-related issues remain among the most common root causes of web application breaches. Studies point to the increasing complexity of cloud, containerized, and serverless environments, where misconfigured access policies, exposed administrative services, and insecure default settings frequently occur due to human error or oversight rather than technological limitations. This body of work underscores the urgent need for automated configuration auditing and secure defaults across modern deployment environments.

Research on Software Supply Chain Failures provided compelling evidence of a rapidly growing area of concern, driven by the widespread use of open-source dependencies and automated CI/CD pipelines. Studies reveal how attackers compromise package repositories, manipulate update processes, or introduce malicious components into build pipelines to gain widespread, downstream access. These findings highlight the fragility of modern software ecosystems and the need for stronger dependency vetting, signature verification, and SBOM-based transparency. In the

domain of Cryptographic Failures, prior research reinforced that most real-world encryption breakdowns stem not from the cryptographic algorithms themselves, but from improper key management, insecure protocol configurations, and developer misuse. Misconfigured TLS deployments, hardcoded keys, and reliance on outdated algorithms are recurring issues identified across multiple empirical studies. Finally, research into Injection vulnerabilities shows that despite decades of awareness, injection attacks remain highly prevalent due to inconsistent input validation and unsafe query construction practices. Recent studies highlight the importance of combining secure coding standards with machine learning–based anomaly detection and runtime monitoring to counter increasingly sophisticated injection techniques. Collectively, the literature reviewed in this chapter offers a comprehensive understanding of the technical challenges, threat patterns, and mitigation strategies that underpin the analysis and implementation discussions presented in the subsequent chapters of this study.

Chapter 3

OWASP Top 5 Web Application Security Risks

3.1 Introduction

The security landscape surrounding modern web applications continues to expand in complexity as organizations increasingly adopt cloud-native infrastructures, API-driven communication, and integrated third-party services. These advancements provide significant functional and operational benefits but also introduce new and often subtle attack surfaces. As applications become more interconnected, even small weaknesses in configuration, access handling, or external dependencies can lead to systemic vulnerabilities. Attackers continue to refine their techniques, leveraging automation, reconnaissance tools, and emerging exploitation methods to target any component within the web application ecosystem.

In this environment, identifying and prioritising the most critical vulnerabilities is essential. The OWASP Top 5 Web Application Security Risks provides a focused and data-driven classification of the weaknesses most frequently observed in real-world application breaches. These include Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection. Together, these categories represent issues that span design flaws, development oversights, operational errors, and ecosystem-level threats. Their prevalence across different industries highlights how fundamental these failures are to the overall breakdown of application security.

A detailed examination of these five categories shows that each vulnerability arises under different circumstances yet shares common roots in insufficient security practices. Broken Access Control and Injection vulnerabilities often emerge during development due to inadequate validation or insecure coding patterns. Security Misconfiguration and Cryptographic Failures typically originate during deployment or system maintenance, where improper settings or weak encryption mechanisms remain unnoticed. Software Supply Chain Failures have gained prominence due to the modern dependency-driven development model, illustrating how compromise at any point in the software supply chain can propagate to multiple downstream systems.

Together, these vulnerabilities form the foundation for understanding the broader security posture of a web application. Presenting them in a comparative manner helps establish a clear perspective on their relative characteristics, causes, and impacts. The following table summarises the key aspects of the OWASP Top 5 risks, providing an accessible overview before the chapter progresses into methodological analysis, exploitation workflows, and practical implementation strategies.

Risk	**Primary Cause**	**Exploit Complexity**	**Potential Impact**
Broken Access Control	Missing or weak authorization checks, insecure role validation, exposed endpoints.	Low–Medium	High (privilege escalation, data exposure).
Security Misconfiguration	Default credentials, exposed admin panels, incorrect cloud permissions, unnecessary services enabled.	Low	High (data leaks, service compromise).
Software Supply Chain Failures	Compromised dependencies, malicious packages, insecure build pipelines.	Medium	Very High (wide-scale downstream compromise).
Cryptographic Failures	Weak or outdated algorithms, poor key management, hardcoded or reused keys, misconfigured TLS.	Low–Medium	High (confidentiality loss, credential theft).
Injection	Unsafe handling of untrusted input, concatenated queries, lack of parameterization.	Low	Very High (DB compromise, remote code execution).

Table 3.1: Comparative Overview of OWASP Top 5 Risks

3.2 Methodology

The methodology for this study is based on a structured, research-driven approach designed to analyse the OWASP Top 5 Web Application Security Risks from multiple perspectives. The first phase focused on reviewing the official OWASP 2025 update, which forms the authoritative basis for identifying and defining the five most critical vulnerabilities present in modern web applications. OWASP documentation was used to understand the foundational descriptions, causes, potential attack patterns, and recommended mitigation strategies associated with each vulnerability category. This phase ensured that the study remained aligned with globally recognised standards and reflected the most current threat landscape.

The second phase involved a comprehensive literature review drawing from peer-reviewed research papers, technical analyses, and contemporary cybersecurity studies. One scholarly paper was selected for each vulnerability to support balanced and evidence-based coverage. Selection criteria included publication credibility, relevance to the OWASP category, methodological validity, and applicability to real-world systems. The works of Anas et al. (Broken Access Control), Loureiro et al. (Security Misconfiguration), Ohm et al. (Software Supply Chain Failures), Gautam et al. (Cryptographic Failures), and Ross (Injection) provided practical and research-oriented insights into how these vulnerabilities manifest and are exploited.A general exploitation workflow representing how attackers identify and exploit weaknesses was incorporated here to support this phase of the study.

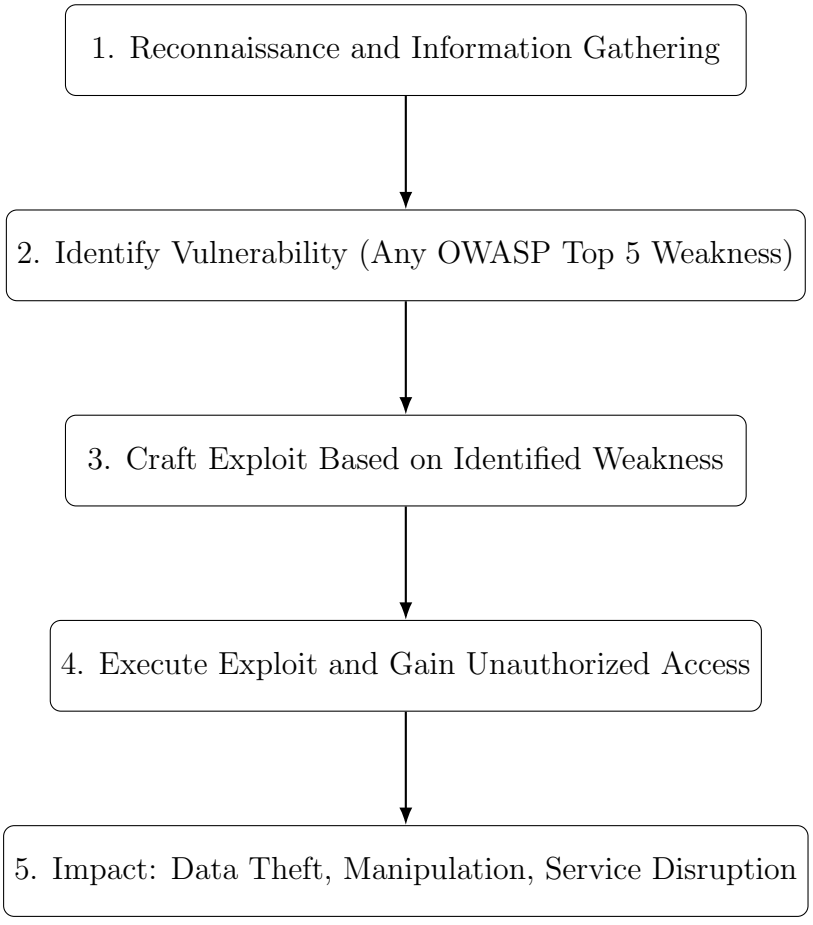

FIGURE 3.1: General Exploitation Workflow for OWASP Top 5 Vulnerabilities

The third phase focused on synthesising information from OWASP documentation and the reviewed literature to identify recurring themes and patterns across the five vulnerabilities. This included examining shared root causes, similarities in exploitation paths, dependency between configuration and code-level weaknesses, and the impact of organisational practices on vulnerability prevalence. This comparative synthesis provided a deeper understanding of how the different vulnerabilities relate to confidentiality, integrity, and availability within the security triad, and helped shape a more unified analytical perspective for the study.

The final phase evaluated the mitigation strategies proposed across both OWASP guidelines and academic research. Each mitigation technique was assessed for practicality, implementation difficulty, scalability, and long-term effectiveness in real-world operational environments. This phase also analysed whether vulnerabilities persist primarily due to technical oversights, human error, insecure development processes, or inadequate monitoring. Through this multi-layered approach, the methodology ensures a comprehensive and reliable examination of the OWASP Top 5 security risks, enabling a well-rounded understanding of how these vulnerabilities emerge, how they are exploited, and how they can be effectively addressed.

3.3 Implementation

The implementation of the OWASP Top 5 Web Application Security Risks in this study focuses on demonstrating how these vulnerabilities appear in real-world environments and how organizations can practically integrate security measures to prevent them. Rather than performing experimental testing or hands-on exploitation, this section relies on established case studies, empirical research, and widely

recommended industry practices. The objective is to translate the theoretical definitions of each vulnerability into real implementation contexts, emphasising how these weaknesses manifest during system development, deployment, and maintenance.

To contextualize the overall defensive approach, a general prevention workflow has been included to illustrate how secure development and operational practices can collectively mitigate the OWASP Top 5 vulnerabilities. This workflow visually represents the recommended sequence of preventive actions—from secure design to continuous monitoring—and supports the implementation strategies discussed in the following paragraphs.

The implementation of Broken Access Control is demonstrated through patterns such as insecure role validation, predictable identifiers, missing authorization checks, and insufficient privilege separation. Common real-world cases include unauthorized access achieved by manipulating URL parameters or accessing API endpoints that lack server-side enforcement of user roles. Implementation strategies toward mitigating these issues include establishing robust role-based or attribute-based access controls, validating authorization on the server for every request, and ensuring proper session and identity management across the application.

Security Misconfiguration is addressed by examining the operational oversights that frequently expose systems to attacks. Examples include default or weak credentials left unchanged, unnecessary services running in production, directory listing enabled on web servers, and misconfigured cloud storage buckets leading to public data exposure. Implementation countermeasures involve using configuration hardening guides, automated security configuration scanning, environment-specific configuration files, and strict disabling of debug and developer tools in production. CI/CD pipelines can also incorporate configuration validation steps to proactively detect unsafe deployments.

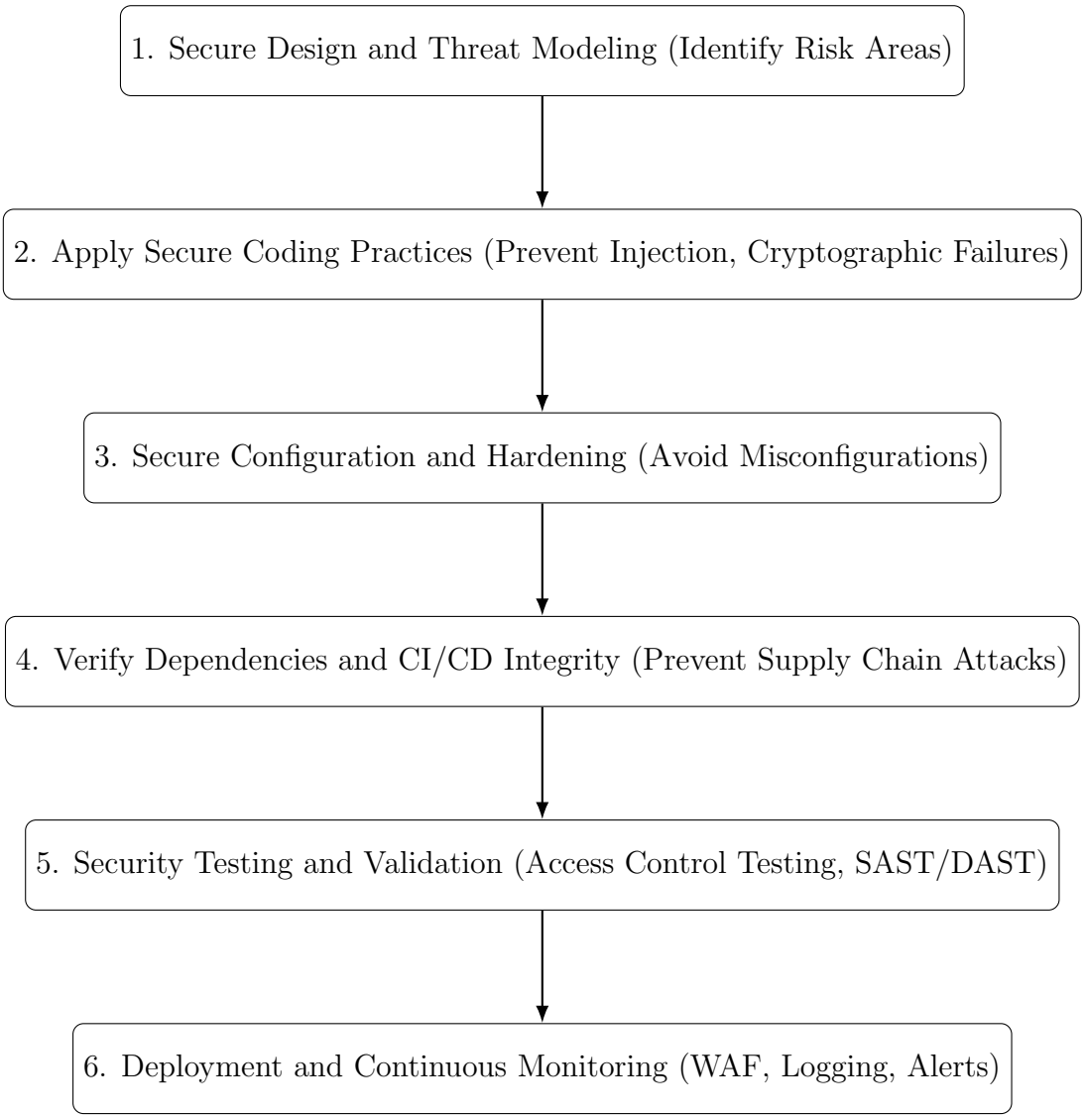

FIGURE 3.2: General Prevention Workflow for Mitigating OWASP Top 5 Risks

For Software Supply Chain Failures, implementation focuses on managing dependencies and maintaining the integrity of external software components. Realistic attack scenarios include malicious packages published to public repositories, typosquatting libraries, or compromised CI/CD environments that inject malicious code into build artifacts. Preventive implementation practices include using dependency-locking files, verifying the authenticity of libraries through signatures or checksums, maintaining a Software Bill of Materials (SBOM), and integrating automated software

composition analysis (SCA) tools into development pipelines.

The implementation of Cryptographic Failures emphasizes secure encryption usage and proper key lifecycle management. Common implementation flaws include using outdated cryptographic algorithms, transmitting sensitive data without encryption, storing keys directly in source code, or improperly configuring SSL/TLS certificates. Secure implementation practices involve adopting modern cryptographic algorithms such as AES-256 and TLS 1.3, enforcing encrypted communication channels, using dedicated key vaults or hardware security modules (HSMs), and automating certificate renewal and rotation processes.

Injection vulnerabilities are implemented through common programming mistakes where untrusted user input is processed without proper validation. Examples include SQL injection via concatenated queries, command injection through system-level functions, and script injection in web forms or API endpoints. Mitigation strategies include the consistent use of parameterized queries, input validation and sanitization techniques, adopting ORM frameworks, enabling content security policies, and adding protective layers such as Web Application Firewalls (WAFs). Research-backed approaches involving anomaly detection and machine learning can provide additional defensive depth in high-risk environments.

Taken together, these implementation strategies demonstrate how the OWASP Top 5 vulnerabilities can be effectively mitigated through a combination of secure design principles, disciplined development practices, strict configuration management, and continuous monitoring. This section highlights how theoretical vulnerabilities translate into actionable security engineering requirements and how organizations can incorporate practical controls at multiple stages of the software lifecycle.

3.4 Advantages & Disadvantages

Advantages

The following section outlines the key advantages associated with understanding and applying the OWASP Top 5 in web application security. These points highlight the overall benefits that organizations and development teams can gain from adopting this framework.

1. **Clear Prioritization of Critical Risks**

   The OWASP Top 5 highlights the most severe and common vulnerabilities, helping organizations focus their security efforts where they will have the greatest impact.

2. **Industry-Recognized Framework**

   OWASP is widely accepted across cybersecurity and software engineering communities, providing a universal language for discussing and managing web security risks.

3. **Supports Secure Development Practices**

   The categorization aligns with secure SDLC activities such as code reviews, architecture analysis, and threat modeling, making integration into development workflows straightforward.

4. **Educational Value**

   It simplifies complex security topics into understandable categories, making it useful for training developers, students, and security professionals.

5. **Versatile Across Technologies**

   The OWASP Top 5 can be applied to traditional web apps, APIs, cloud-native services, and microservices, offering broad relevance.

Disadvantages

This section briefly presents the main disadvantages linked to the OWASP Top 5. These points help provide a balanced perspective by acknowledging the limitations and challenges that may arise during its adoption.

1. **Broad and Generalized Categories**

   The vulnerabilities are grouped at a high level, occasionally lacking granular detail needed for highly specific security scenarios.

2. **Slow Update Cycle**

   Since OWASP updates the list every few years, rapidly emerging threats may not be represented immediately.

3. **Requires Interpretation by Experts**

   Applying OWASP guidelines effectively often requires experienced developers or security analysts to contextualize the recommendations for their environments.

4. **Not a Complete Security Checklist**

   The OWASP Top 5 identifies major risks but does not provide all possible vulnerabilities or step-by-step corrective actions.

5. **Implementation Cost and Effort**

   Fully addressing these vulnerabilities may require organizational changes, training, and specialized tools that smaller teams may find challenging.

3.5 Applications

The section below lists several applications of the OWASP Top 5 framework. These points demonstrate how the principles can be utilized in practical, technical, and organizational contexts.

- **Secure Software Development Lifecycle (SDLC):** Used to define coding standards, checklists, and review protocols throughout the development phases from design to deployment.
- **Penetration Testing and Security Audits:** Provides a baseline for vulnerability assessments, test cases, and attack simulations performed by security teams.
- **Risk Assessment and Prioritization:** Helps organizations classify and prioritize vulnerabilities based on their prevalence and severity.
- **Developer and Cybersecurity Training:** Forms the core content in educational programs, workshops, and professional training modules to improve security awareness.
- **DevSecOps and CI/CD Integration:** Guides automated checks such as dependency scanning, configuration audits, and static code analysis within pipelines.
- **Cloud and API Security Reviews:** Applied in evaluating cloud misconfigurations, insecure access controls, and API exposure commonly found in modern architectures.

- **Policy and Compliance Frameworks:** Many organizations use OWASP guidelines to form internal security policies or complement external standards such as ISO 27001, SOC2, and PCI-DSS.

3.6 Findings and Discussion

The findings of this study indicate that the OWASP Top 5 Web Application Security Risks continue to represent the most prevalent and damaging categories of vulnerabilities observed across modern web applications. Broken Access Control, Security Misconfiguration, and Injection remain especially widespread, primarily because they emerge from common development mistakes, inconsistent enforcement of validation checks, and operational oversights that accumulate over time. These vulnerabilities frequently appear in production systems not because they are difficult to understand, but because they require disciplined implementation practices and continuous verification—processes that many development teams struggle to maintain. The literature reviewed in earlier chapters reinforces that even well-established application architectures can suffer from these weaknesses if authorization logic is not consistently applied, configurations are not regularly audited, or input handling is not rigorously controlled.

The findings also stress the rising significance of Software Supply Chain Failures and Cryptographic Failures in the current threat landscape. As organizations increasingly rely on external libraries, container images, APIs, and CI/CD automation, the risk of compromise through third-party dependencies continues to grow. Studies demonstrate how attackers exploit this reliance by poisoning software packages, compromising build pipelines, or injecting malicious components into trusted supply chains. Similarly, cryptographic failures persist not due to inherent flaws in

modern algorithms but because of incorrect implementation practices, such as weak key management, outdated protocols, misconfigured certificates, or hardcoded secrets within application code. These implementation errors weaken otherwise strong cryptographic protections and expose sensitive data to unauthorized access.

Across all five vulnerabilities, the findings highlight a common theme: although effective mitigation techniques are well-documented and technically straightforward, their real-world application often falls short due to gaps in secure development practices, inadequate testing, and insufficient organizational awareness. Many organizations lack structured processes for continuous security assessment, rely on outdated configurations, or prioritize rapid deployment over secure design and validation. This discussion suggests that addressing the OWASP Top 5 requires more than isolated fixes—it demands a coordinated effort that integrates secure coding principles, automated configuration management, supply chain validation, and continuous monitoring into the entire software development lifecycle. Strengthening these foundational practices can significantly reduce the likelihood of exploitation and enhance the overall security posture of modern web applications.

Chapter 4

Conclusions and Future Scope

The study of the OWASP Top 5 Web Application Security Risks highlights the persistent and critical vulnerabilities that continue to impact modern web applications regardless of industry, scale, or technological maturity. Through a detailed review of OWASP documentation, academic research, and practical implementation studies, it becomes evident that weaknesses such as Broken Access Control, Security Misconfiguration, Software Supply Chain Failures, Cryptographic Failures, and Injection remain widespread due to recurring deficiencies in secure design, validation processes, configuration management, and dependency handling. These vulnerabilities are not merely isolated technical defects but are symptomatic of broader structural issues in software development practices, such as inconsistent enforcement of security policies, incomplete understanding of secure coding principles, rushed deployment cycles, and inadequate monitoring. Although established mitigation strategies exist for each of the OWASP Top 5 categories, the study shows that their real-world effectiveness relies heavily on consistent adoption, disciplined implementation, and ongoing evaluation throughout the software development lifecycle. Ultimately, this research underscores the need for a proactive security culture—one that embeds

threat modeling, configuration auditing, dependency verification, and robust encryption practices as fundamental components of responsible software engineering.

This study further demonstrates that the evolving nature of web technologies continues to introduce new layers of complexity, making traditional security models insufficient when applied in isolation. The increasing reliance on third-party ecosystems, microservices, multi-cloud deployments, and automated CI/CD pipelines creates additional challenges that demand more adaptive and holistic security strategies. As demonstrated in the literature, even well-intentioned development teams often struggle with visibility into their own supply chains, misconfigured cloud environments, or improper implementation of cryptographic controls. In this context, the OWASP Top 5 serves as a practical baseline that helps organizations prioritize high-impact vulnerabilities and approach web application security with clarity and structure. The findings from this study make it clear that maintaining security is not a one-time task but an ongoing responsibility that requires coordinated effort across design, development, deployment, and maintenance phases.

Looking ahead, there is considerable scope for expanding this work by integrating advanced and emerging security methodologies into both research and practice. Future studies may focus on leveraging artificial intelligence and machine learning to detect authorization anomalies, misconfigurations, and injection attempts with greater precision and reduced manual effort. The rapid rise of software supply chain attacks highlights the growing need for stronger verification frameworks, including automated SBOM (Software Bill of Materials) generation, dependency signature validation, trusted build environments, and enhanced CI/CD integrity scanning. Likewise, emerging security paradigms such as zero-trust architecture, confidential computing, hardware-backed key management, and distributed identity systems offer promising avenues for safeguarding applications at both infrastructure and application layers.

Further enhancement of OWASP's classification framework through more frequent updates, deeper categorization, and integration with cloud-native security standards would also support better industry alignment.

In addition, future scope exists in strengthening the human and organizational aspects of security. Developer training programs, awareness initiatives, and security-focused curricula can significantly reduce the recurrence of preventable vulnerabilities. Organizations may also benefit from expanding secure coding guidelines, implementing DevSecOps pipelines, conducting regular threat modeling workshops, and enforcing continuous compliance checks. Long-term improvements will require not only technological innovation but also a sustained commitment to fostering a security-first mindset across all levels of the development lifecycle. With these advancements, organizations will be better equipped to anticipate, detect, and mitigate the vulnerabilities outlined in the OWASP Top 5, thereby improving the resilience and trustworthiness of modern web applications.

Bibliography

[1] Anas, Ahmed; Elgamal, Salwa; Youssef, Basheer. "Survey on detecting and preventing web application broken access control attacks." International Journal of Electrical and Computer Engineering, 14(1), 772–781, 2024.

[2] Ohm, Marc; Plate, Henrik; Sykosch, Arnold; Meier, Michael. "Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks." arXiv preprint, 2020.

[3] Gautam, A.K.; et al. "A comprehensive study on key management in cryptographic systems." SN Applied Sciences, 3, 2021.

[4] Sivaramakrishna, D.; et al. "A novel hybrid cryptographic framework for secure data storage in cloud computing." Future Generation Computer Systems, 135, 2023.

[5] Ross, Kevin. "Multi-source data analysis and evaluation of machine learning techniques for SQL injection detection." Proceedings of ACM SIGKDD Workshop on Security and Privacy Analytics, 2018.