Journal of Media Horizons
ISSN: 2710-4060 2710-4052 Volume 7, Issue 4, 2026

MODERN WEB APPLICATION SECURITY: A COMPREHENSIVE ANALYSIS OF VULNERABILITIES, EXPLOIT CHAINS, AND PREVENTION STRATEGIES

Muhammad Ahsan Hayat[*1], Engr Maha Zaka[2], Maryam Shaikh[3], Arsalan Haider[4], Muhammad Arshad Patoli[5], Areeba Noor[6], Muhammad Anas Aslam[7]

[*1,4]*Senior Lecturer, Department Computer Science, Iqra University, Karachi, Pakistan*
[2]*Lecturer, Department Computer Science, Iqra University, North Campus, Karachi*
[3]*Lecturer, Department of Computer Science, Iqra University, Karachi, Pakistan*
[5]*Director Libraries Sindh, Directorate General of Public Libraries Sindh, Culture Tourism Antiquities and Archives Department Government of Sindh, Karachi, Pakistan*
[6,7]*Department of Criminology, University of Karachi, Karachi, Pakistan*

[*1]Muhammad.ahsan@iqra.edu.pk, [2]maha.zaka@iqra.edu.pk, [3]maryam.shaikh@iqra.edu.pk, [4]arsalanhaider102@gmail.com, [5]arshadpatoli1979@gmail.com, [6]areeban08@gmail.com, [7]anasjat63@gmail.com

[*1]https://orcid.org/0009-0001-5063-7603

DOI: https://doi.org/10.5281/zenodo.20024270

Keywords
Web Application Security, OWASP Top 10, DevSecOps, API Security, Credential Theft, Software Supply Chain.

Article History
Received: 11 March 2026
Accepted: 21 April 2026
Published: 30 April 2026

Corresponding Author: *
Muhammad Ahsan Hayat

Abstract

Web applications which use cloud-native architectures in their distributed operational environments have their attack surface area increased. The research study establishes a complete system of web application security weaknesses which matches the DOA Top 10 list (2025) and the API Security Top 10 list (2023) while including additional threats from supply-chain operations and cloud-native environments. The empirical research shows that 77% of web application security breaches occur through stolen credentials, while 21% of breaches happen due to brute-force attacks and 13% of breaches occur through direct vulnerability exploitation according to Verizon DBIR 2024. The OWASP findings show that 3.73% of assessed applications display Broken Access Control while 3.00% show Security Misconfiguration and 3.80% exhibit Cryptographic Failures which demonstrates ongoing fundamental security vulnerabilities. The research develops multi-phase exploitation models which begin with actual security breaches that occurred between 2018 and 2026 to show how attackers begin their intrusion attempts which lead to extensive data loss incidents. The evaluation of tools shows that SAST DAST and runtime systems demonstrate different detection abilities because both false positives and integration difficulties act as major drawbacks. The research establishes a defense framework which combines secure SDLC methods with DevSecOps pipeline security and runtime security assessment. The results show that identity theft and system misconfiguration errors represent the main causes of contemporary web security breaches which require organizations to implement complete security frameworks.

INTRODUCTION

Background and motivation

The modern web application has evolved from a monolith into a constellation of services: single-page applications, microservices, SaaS identity, service meshes, serverless functions, and third-party scripts. Each integration increases the attack surface, creating new boundaries of trust that are not necessarily represented in code logic but are instead represented in configuration and policy. This is also represented in OWASP Top 10:2025, where "Security Misconfigurations" is elevated in priority, "Software Supply Chain Failures" is introduced as its own category, and SSRF is subsumed under "Broken Access Control," indicating the presence of these boundaries and composition risk. [8]

The threat model is not just represented by vulnerability discovery but also by exploitation economics. The Verizon DBIR indicates that even when "Basic Web Application Attacks" is not represented as a primary breach model, credential abuse is a common cause: "77 percent of data breaches in this pattern involve the use of stolen credentials, whereas brute force attacks and vulnerability exploitation are also present but not as common." [9] This puts authentication/session management, rate limiting, and automation protection alongside secure coding as first-order concerns.

Scope and definitions

The paper specifically deals with modern web application vulnerabilities. The vulnerabilities discussed in this paper include vulnerabilities in web applications, APIs, infrastructure components such as reverse proxies and gateways, identity components, third-party components and dependencies, and cloud native deployment components that impact web application security. The paper discusses vulnerabilities at different levels: design and logic-level vulnerabilities (not CVE addressable), implementation-level vulnerabilities (usually CVE addressable), and operational configuration-level vulnerabilities (not always CVE addressable but usually the key link in the chain for many incidents). The OWASP Top 10 2025 is the primary high-level categorization for risks. [1]

Research Objectives

This study aims to:

O1: Develop a unified taxonomy of modern web application vulnerabilities across application, API, supply chain, and cloud-native layers

O2: Quantitatively analyze vulnerability prevalence and breach patterns using recent datasets (2021–2026)

O3: Model multi-stage exploit chains to understand real-world attack progression

O4: Map vulnerabilities to effective prevention and mitigation strategies

O5: Evaluate modern security tools in terms of detection coverage and limitations

O6: Provide actionable recommendations for secure SDLC and DevSecOps environments

Novel Contributions

This work provides the following key contributions:

- A multi-layer taxonomy integrating OWASP, API security, cloud-native, and supply-chain risks
- A combined empirical + exploit-chain perspective, linking vulnerability prevalence with real breach behavior
- A cross-domain mapping of vulnerabilities → exploit chains → prevention strategies
- A tool-centric evaluation highlighting real-world limitations of AppSec tooling
- A defense-in-depth architecture tailored for modern microservices and cloud-native systems

Literature Review

Recent studies emphasize the growing complexity of web application security due to cloud-native architectures and software supply chain dependencies. Research shows that modern vulnerabilities are increasingly configuration-driven rather than purely code-based.

- Studies on Log4Shell (2021) highlight systemic risks in widely used libraries

- Supply-chain attacks such as Codecov (2021) and xz backdoor (2024) demonstrate trust boundary exploitation
- API security research (2022–2025) identifies Broken Object Level Authorization (BOLA) as a dominant risk
- Empirical evaluations of SAST/DAST tools reveal high false positive rates (20–40%) and incomplete coverage
- Cloud-native security research stresses runtime monitoring and zero-trust models

Overall, literature indicates a shift from single vulnerabilities → complex exploit ecosystems, validating the need for integrated approaches like this study.

Methodology

Evidence collection and selection

The sources were given the following priority:

1) Primary and official sources: OWASP Top 10 (2025), OWASP API Security Top 10 (2023), NIST Secure Software Development Framework (SP 800-218), NIST Cloud Native DevSecOps (SP 800-204C), NIST API protection (SP 800-228), W3C CSP Level 3 specification, NIST National Vulnerability Database CVE entries, CISA exploitation alerts/advisories, and vendor security advisories. [10]

The methodology follows a structured multi-stage workflow. Data collection begins with authoritative sources which include OWASP, NIST, and Verizon DBIR reports. The collected data undergoes validation procedures which confirm its reliability and relevant to the research. The research team developed a complete vulnerability taxonomy based on their findings. Statistical trend analysis detects patterns which show how common different items are. The team uses exploit chain modeling to study how vulnerabilities spread through system boundaries. The evaluation process tests modern security tools which helps researchers understand their detection capabilities and their operational boundaries. The team mapped prevention strategies to known vulnerabilities while synthesizing their findings to create practical insights.

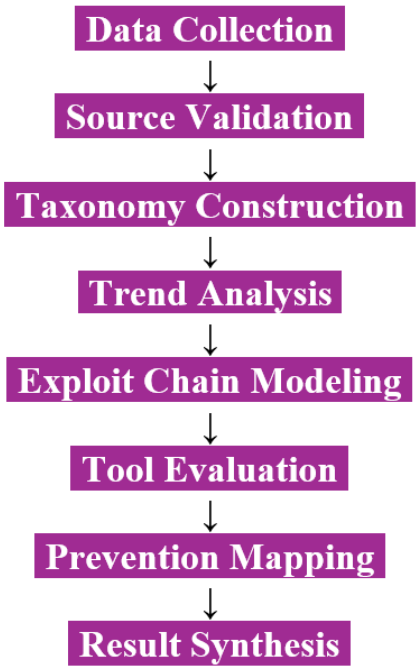

Chart 1: Methodology Workflow

2) Peer-reviewed literature for incident analysis and supply chain/tool evaluation:

- Systematic analysis of the Capital One data breach incident;
- Supply chain surveys;
- Evaluation of SAST tools. [11]

3) Good quality industrial reports in the absence of primary sources or to support interpretational understanding.

Taxonomy construction

The taxonomy was constructed by:

- Using OWASP Top 10:2025 categories as the baseline, including category-specific notes (e.g., SSRF rolled into Broken Access Control; prevalence indicators for selected categories). [8]
- Expanding for API-specific risks using OWASP API Security Top 10 (2023). [12]
- Adding supply-chain and cloud-native layers aligned with SBOM guidance (NTIA minimum lements), SLSA provenance levels, and NIST cloud-native DevSecOps and API protection guidance. [13]

Trends and timeline methodology

Trends were supported by:

- OWASP Top 10 2025's data-informed methodology and prevalence statements for selected categories (e.g., 3.73% for Broken Access Control; 3.00% Security Misconfiguration; 3.80% Cryptographic Failures). [8]
- Verizon DBIR 2024 and 2025 snapshots for breach-pattern action prevalence, focusing on credential abuse vs vulnerability exploitation. [14]
- A curated timeline of widely exploited, web-relevant CVEs from 2018–2025 drawn from NVD records and corroborated exploitation advisories from CISA where available. [15]

Tool evaluation methodology

Tools were assessed qualitatively on: detection coverage, precision/false positives (as described in peer-reviewed comparisons where available), CI/CD integration, scalability, deployment model, and suitability for modern architectures (SPAs/APIs/microservices). Tool capabilities were derived from official documentation and maintained project repositories; claims beyond documentation were explicitly treated as analytical interpretation. [16]

Results

The results directly correspond to the defined research objectives by presenting taxonomy development (O1), statistical analysis (O2), exploit chain modeling (O3), and mitigation mapping (O4).

Taxonomy of modern web application vulnerabilities

OWASP Top 10:2025 is the most current consensus list of critical web application security risks. It reports that Broken Access Control remains #1, with contributed data indicating an average of 3.73% of tested applications having at least one of the category's mapped CWEs; Security Misconfiguration is #2 (3.00%); Cryptographic Failures reports 3.80% and often leads to sensitive data exposure or compromise; and Injection remains heavily tested and spans issues from high-frequency/low-impact XSS to low-frequency/high-impact SQLi. [8]

OWASP also clarifies that its 2025 edition is "data-informed" and uses both contributed testing data and community survey input because some high-risk categories are difficult to test at scale; it further notes that it leverages CVE mappings (via NVD/CWE linkages) for exploitability and impact scoring. [8]

Table 1 provides a consolidated taxonomy across OWASP Top 10 (2025), API-specific risks, supply chain, and cloud-native layers, with representative examples and illustrative CVEs where applicable.

Table 1 — Vulnerability taxonomy with examples and CVE references (2018–2025)

Category (modern web context)	Typical weakness pattern	Illustrative CVEs / incidents (examples)	Primary notes / mapping
Broken access control and trust-boundary failures	Missing object/function authorisation, IDOR/BOLA, SSRF, tenant isolation breakdown	Capital One breach chain includes SSRF + cloud credential misuse (case study below) [17]	OWASP Top 10:2025 #1; explicitly folds SSRF into this category [8]
Security misconfiguration	Exposed admin interfaces, insecure defaults, permissive CORS, debug endpoints, weak headers	**CVE-2020-5902** (BIG-IP TMUI RCE on management interface) [18]	OWASP Top 10:2025 #2; prevalence statement 3.00% [8]
Software supply chain failures	Compromised dependencies, build pipeline compromise, malicious releases, compromised signing/provenance	**CVE-2024-3094** (xz/liblzma malicious code in release tarballs); Codecov Bash Uploader incident (2021) [19]	OWASP Top 10:2025 #3; framed as broader than "vulnerable/outdated components" [1]
Cryptographic failures	Weak/absent encryption, poor key management, insecure randomness, sensitive data exposure	Often context-specific; risk amplified by misconfig and design gaps rather than a single CVE [8]	OWASP Top 10:2025 #4; prevalence statement 3.80% [8]
Injection (including deserialisation/interpretation)	SQL/NoSQL injection, OGNL/template injection, command injection, JNDI injection	**CVE-2023-34362** (MOVEit Transfer SQLi); **CVE-2022-26134** (Confluence OGNL injection RCE); **CVE-2021-44228** (Log4j2 JNDI injection) [20]	OWASP Top 10:2025 #5; includes XSS↔SQLi spectrum [8]
Insecure design and business logic flaws	Missing threat modelling, insecure workflows, abuse of intended features, broken rate limits	Web login automation and credential stuffing exemplify logic + control gaps (see DBIR chart) [2]	OWASP Top 10:2025 #6; design-level category introduced in 2021 and continues [8]
Authentication failures and session/token abuse	Weak MFA UX, session fixation, token leakage, password reset flaws	**CVE-2023-4966** (CitrixBleed session token disclosure); **CVE-2025-5777** (CitrixBleed 2 memory	OWASP Top 10:2025 #7; aligns with observed credential abuse dominance [22]

		overread leading to token theft) [21]	
Software or data integrity failures	Unsigned updates, untrusted deserialisation, CI/CD artefact integrity gaps	Codecov Bash Uploader compromise targeted CI secrets/artefact trust [23]	OWASP Top 10:2025 #8; distinct from broader supply chain failures [8]
Logging and alerting failures	Missing security logs, no actionable alerting, poor detection engineering	CISA advisories frequently emphasise detection/response capabilities during active exploitation [24]	OWASP Top 10:2025 #9 [8]
Mishandling of exceptional conditions	Fail-open behaviour, unsafe error handling, improper resource cleanup	Often manifests as business logic + defensive coding failure; difficult to measure at scale [8]	OWASP Top 10:2025 #10; new category [8]
API-specific authorisation failures (BOLA/BOPLA)	Object-level authorisation missing at endpoints; property-level leakage/mass assignment	API1:2023 Broken Object Level Authorization (risk description) [12]	OWASP API Security Top 10 (2023) is the API-focused taxonomy [25]
API security misconfiguration and abuse	Excessive data exposure, rate-limit absence, SSRF via integrations, unsafe GraphQL	NIST SP 800-228 frames API risks across lifecycle and recommends pre-runtime and runtime controls [26]	Maps to both OWASP API and OWASP Top 10:2025 categories [27]
Cloud-native platform escapes / container runtime risks	Container escape, unsafe capabilities, weak node isolation	**CVE-2019-5736** (runc container escape) [28]	Cloud-native guidance emphasises controls across build-to-runtime pipelines [29]
Public-facing infrastructure flaws adjacent to web apps	Directory traversal, exposed gateways, edge device compromise	**CVE-2019-19781** (Citrix ADC/Gateway traversal); **CVE-2021-41773/42013** (Apache httpd path traversal/incomplete fix) [30]	Frequently exploited; CISA issued detection guidance for Citrix (2020) [31]

Recent trends, statistics, and vulnerability prevalence

The strongest empirical signal across "web application attacks" in recent breach data is the predominance of credential abuse over pure vulnerability exploitation within that pattern. The Verizon 2024 DBIR states that "just over 8%" of breaches fall into "Basic Web Application Attacks," and within that pattern, attackers gain access primarily via the use of stolen credentials (77%), followed by brute force (21%) and "exploit vuln" (13%). [9]

To visualise this distribution:

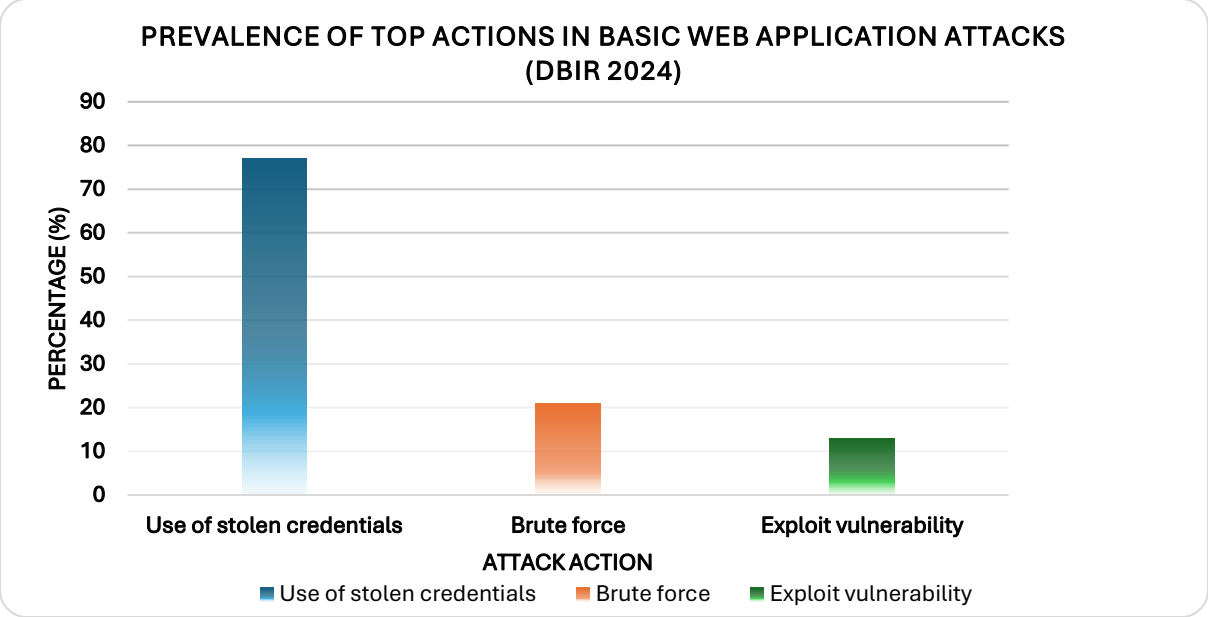

Figure 1 — Prevalence of top actions in Basic Web Application Attacks (DBIR 2024) [9]

A consistent conclusion is that authentication/session controls and anti-automation are core web security controls, not "adjacent" controls. This interpretation is strengthened by DBIR's long-horizon observation that stolen credentials appear in ~31% of breaches over the past ten years. [2]

OWASP Top 10:2025 provides complementary *application testing prevalence* indicators. It reports that Broken Access Control remains the most serious risk and notes, based on contributed data, that 3.73% of tested applications had at least one CWE instance in this category; Security Misconfiguration is reported at 3.00%; and Cryptographic Failures at 3.80%. [8] Although these percentages represent different measurement frames than DBIR breach rates, their combination suggests that (i) boundary failures are common in tested apps, and (ii) identity attacks and misconfigurations dominate realized compromises.

CVE timeline and representative examples (2018–2026)

The vulnerability ecosystem relevant to web applications includes both "application bugs" (e.g., SQLi) and "web-adjacent" components (reverse proxies, gateways, runtimes, CI/CD tooling). NIST NVD provides canonical CVE records; CISA's Known Exploited Vulnerabilities (KEV) alerts provide evidence of active exploitation for selected vulnerabilities. [32]

Year	Event Description
2018	CVE-2018-7600 (Drupal RCE)
2019	CVE-2019-19781 (Citrix ADC traversal); CVE-2019-5736 (runc escape)
2020	CVE-2020-5902 (F5 BIG-IP RCE); Capital One regulatory penalty
2021	Apache httpd CVE-2021-41773/42013; Log4Shell; Codecov compromise
2022	Spring4Shell; Confluence OGNL RCE
2023	MOVEit SQLi exploited by CL0P; CitrixBleed token leak
2024	CVE-2024-3094 xz/liblzma backdoor
2025	CVE-2025-5777 (CitrixBleed 2) added to CISA KEV
2026	Security focus on supply-chain and tool-chain hardening (YTD)

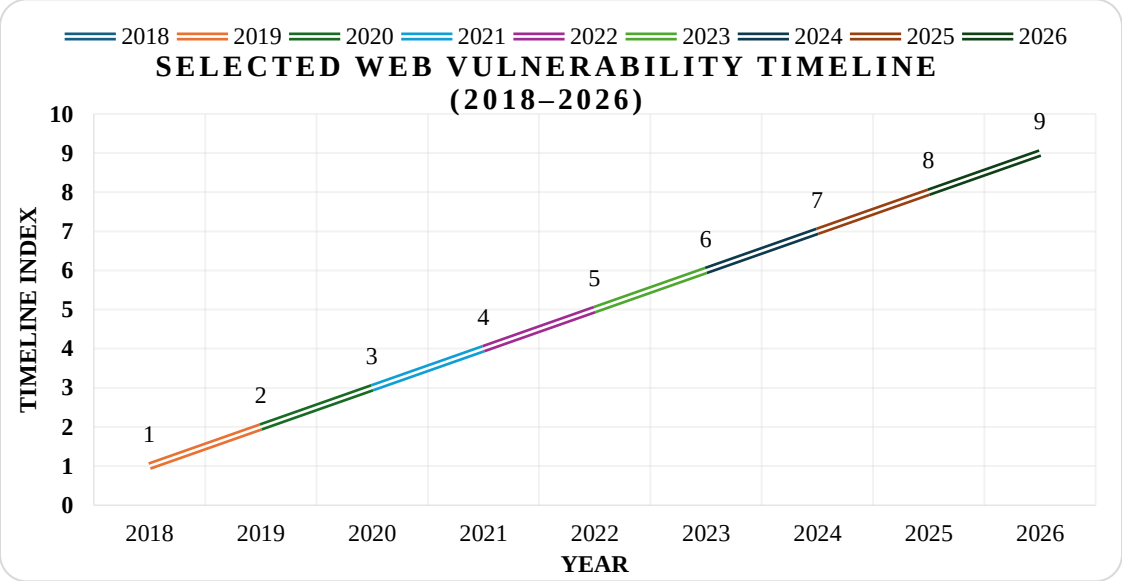

Figure 2 — Mermaid timeline of selected high-impact, web-relevant CVEs and incidents (2018–2026) [33]

The endpoint of 2026 is explicitly constrained by data completeness (the year is in progress) and is an "as-of" boundary.

Attack Techniques and Exploit Chains

In all recent incidents, attack chains rarely end with the initial web exploit. Rather, they cross trust boundaries: application → identity → cloud control plane → data stores → monitoring/response systems

.

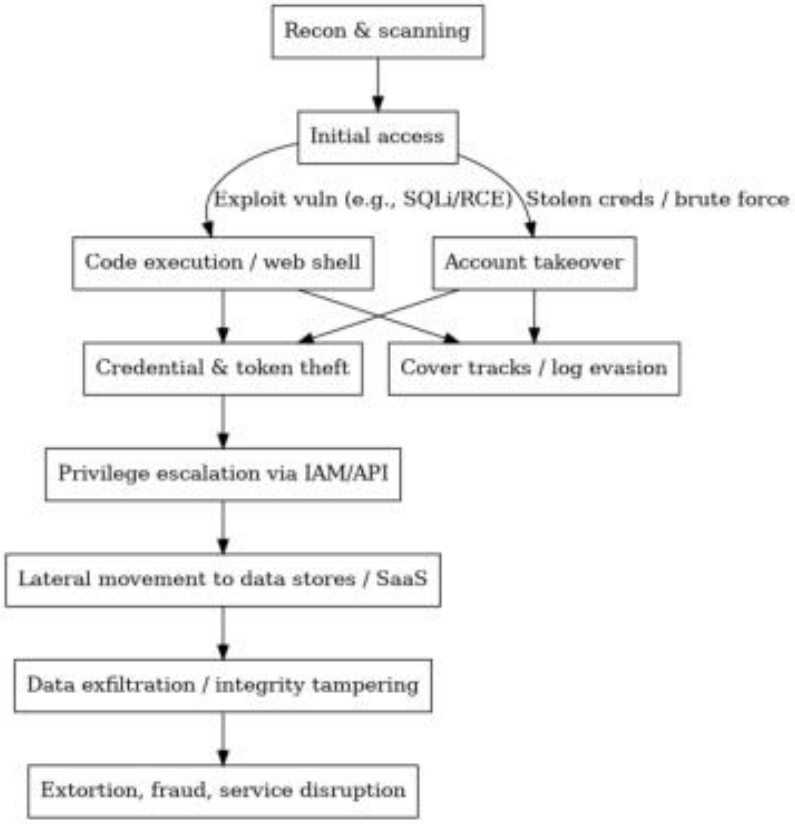

Figure 3 — Mermaid exploit-chain model (web entry to cloud/data impact) [34]

This model is also in line with DBIR findings that identify that for web application attacks, credential abuse is a major initial action type, whereas vulnerability exploitation is a critical but not exclusive action type. [14] This model is also in line with the NIST SP 800-228 that recognized that API protection needs pre-runtime and runtime controls throughout the API lifecycle since exploitation can be both development artefact-based (tokens/secrets) and channel-based. [26]

Case studies of notable incidents

MOVEit Transfer exploitation and extortion (2023)

CVE-2023-34362 is documented in the NVD as a SQL injection vulnerability in the web application for MOVEit Transfer, which allows for unauthenticated access to the database. [35] The exploitation of this vulnerability, as documented in CISA's joint advisory (#StopRansomware), exemplifies how the CL0P ransomware group took advantage of this zero-day vulnerability, where exploitation began with a SQL injection attack, leading into further compromise activities consistent with web shell exploitation and data exfiltration/extortion schemes. [36] This serves as an example of an exploit chain where a traditional vulnerability class, such as SQLi, serves as a gateway for systemic impact, highlighting the importance of both preventative measures and detection/patch response.

Codecov Bash Uploader supply-chain compromise (2021)

The official post-mortem by Codecov describes the attack vector where the threat actor attacked the Bash Uploader tool to distribute the payload to the users of the tool. [37] Codecov describes the attack vector where the threat actor accessed the tool due to a mistake in the creation of the Docker

image where the credential to the script modification tool was revealed. This is an example of a CI/CD pipeline trust issue rather than an endpoint vulnerability. [38] CISA published an alert on the Codecov detections on the supply chain compromise. This shows the relevance of the attack to national vulnerability management. [39] This attack vector demonstrates the importance of SBOMs and provenance along with CI/CD pipeline security.

Capital One cloud/web exploit chain and regulatory impact (2019–2020)

Analyses by peer review detail how the attack on Capital One occurred by a combination of a web-facing vulnerability and access to cloud metadata. The Office of the Comptroller of the Currency calculated an $80 million civil money penalty for the Capital One banks. This demonstrates that an attack such as this can have a consequence in terms of a penalty in addition to the technical impact. The impact is therefore not just confidentiality but also business and governance. The attack demonstrates the need for defense in depth for SSRF/metadata access and least privilege IAM.

Impact analysis

In the context of the contemporary web application compromise, the impact on the classical CIA triad is as follows:

- **Confidentiality:** Data exfiltration from the database/APIs, as exemplified by SQLi leading to database access. This is exemplified in the description of SQLi for MOVEit above and the CISA advisory on extortion-based exploitation. [42]
- **Integrity:** The compromise of software supply chains could allow for the injection of malicious code. This is exemplified in the NVD CVE-2024-3094 entry on the injection of malicious code into the release tarballs. [43]
- **Availability:** A number of the vulnerabilities classified as web adjacent enable system disruption through RCE or exhaustion. NVD explicitly describes RCE classifications like CVE-2020-5902 as well as path traversal/RCE classifications like CVE-2021-41773. [44]
- **Financial/Legal:** The impact on the organization from a financial/legal standpoint is significant. The OCC penalty on Capital One is a clear example of the financial/legal impact of security control deficiencies. [41]
- **Reputational/Operational:** The CISA advisories for widespread exploitation of vulnerabilities like MOVEit and CitrixBleed/CitrixBleed 2 indicate a widespread level of urgency. This is a direct impact on the operations of the organization. [45]

Prevention and mitigation strategies mapped to vulnerabilities

Effective mitigation is layered. OWASP emphasises secure coding and testing guidance (e.g., injection prevention via safe APIs and parameterisation; authentication and authorisation cheat sheets); W3C CSP Level 3 provides a browser-enforced policy layer to reduce the impact of content injection; and NIST guidance (SSDF, DevSecOps, API protection) frames organisational and pipeline-level controls. [46]

Table 2 — Mitigation techniques mapped to vulnerability classes

Vulnerability class	High-leverage mitigations	Supporting standards/guidance
Injection (SQLi, OGNL, template, command)	Parameterised queries/safe APIs; strict input handling where applicable; output encoding; least-privilege DB accounts; regression tests for sinks	OWASP SQLi & Injection Prevention cheat sheets; OWASP Top 10 Injection guidance [47]
XSS and content injection	Contextual output encoding; template auto-escaping; CSP with nonces/hashes; avoid dangerous DOM sinks	OWASP XSS Prevention cheat sheet; W3C CSP3 spec; OWASP CSP cheat sheet [48]
Broken access control / BOLA	Deny-by-default; centralised authorisation enforcement; object-level checks per request; policy tests; strong tenancy isolation	OWASP authorisation guidance; OWASP API Top 10 (API1:2023); OWASP Broken Access Control description [49]
Authentication failures / credential abuse	MFA with anti-fatigue patterns; rate limiting; bot/captcha strategies; strong session management; secure password storage	OWASP Authentication and Session Management cheat sheets; DBIR credential prevalence motivates priority [50]
Security misconfiguration	Harden defaults; minimise exposed admin surfaces; secure headers; infrastructure-as-code scanning; segmented environments	OWASP Security Misconfiguration prominence; DevSecOps pipeline guidance stresses automated checks [51]
Supply chain failures	SBOM generation and consumption; signed/provenanced builds; dependency policy gates; protected CI secrets; reproducible builds where feasible	NTIA SBOM minimum elements; SLSA provenance levels; OWASP supply chain category [52]
CI/CD compromise and artefact integrity	Least privilege for CI runners; secret scanning; isolated build environments; attestations; monitoring of pipeline changes	NIST SP 800-204C DevSecOps primitives; Codecov incident illustrates pipeline credential exposure risk [53]
API lifecycle risks	Schema/contract-first design; authN/authZ at service and end-user levels; runtime protection at gateways; continuous monitoring	NIST SP 800-228 (pre-runtime + runtime controls); OWASP API Top 10 [54]
Cloud-native and container risks	Harden container runtime; minimise privileges/capabilities; runtime detection; secure service-to-service authZ	NVD CVE-2019-5736; Falco runtime detection; NIST cloud-native DevSecOps guidance [55]

Evaluation of tools and frameworks

Frameworks such as NIST SSDF and NIST cloud-native DevSecOps guidance formalise "security as process" rather than ad hoc testing. NIST SP 800-218 (SSDF) provides practices for secure development. NIST SP 800-204C discusses DevSecOps pipelines as an automated process that includes build, test, packaging, deployment, and operations activities. The process focuses on feedback loops. [56] NIST SP 800-228 applies the life cycle model to APIs and discusses both pre-runtime and runtime protections. The document also discusses implementation trade-offs. [26]

Empirical studies on the evaluation of tools are an ongoing process. Recent systematic studies compare SAST tools based on benchmarks and describe detection trade-offs and integration challenges in CI/CD. [57] At the same time, supply chain security focuses on the diversity of

attacks and the need for a structured approach to risk assessment strategies in the pipeline. [58]

Table 3 — Tool comparison matrix (representative modern AppSec toolchain)

Tool / project	Category	Strengths (from official scope)	Common limitations (analytical)	Source (official)
Semgrep	SAST (+SCA/secrets in platform)	Fast static analysis; rule-based scanning; supports organisational policy/rules workflows [59]	Rule tuning needed to manage false positives; deeper dataflow features vary by mode and language (interpretation)	[60]
CodeQL	SAST / semantic analysis	Query-based analysis over extracted databases; supports custom queries; used for code scanning alerts [61]	Requires query expertise for maximum value; build/database extraction overhead for some stacks (interpretation)	[61]
OWASP ZAP	DAST	Open-source dynamic testing proxy; supports automation (e.g., GitHub Actions scans) [62]	Coverage for complex SPA flows can require scripting/auth handling; may be noisy without configuration (interpretation)	[63]
Burp Scanner (Burp Suite)	DAST	Automated DAST scanner designed to replicate manual tester methodology; maintained vulnerability checks list [64]	Commercial licensing for full capability; automation at scale requires orchestration (interpretation)	[65]
OWASP Dependency-Check	SCA (dependency vulnerability scan)	Scans dependencies to identify known vulnerable components; uses NVD-derived vulnerability data [66]	CPE matching and dependency resolution can produce false positives/negatives; best used with triage gates (interpretation)	[66]
OWASP Dependency-Track	Component analysis platform (SBOM-centric)	Uses SBOMs to identify and reduce software supply-chain risk [67]	SBOM quality and ingestion pipelines determine effectiveness (interpretation)	[67]

Trivy	Vulnerability + misconfiguration scanning	Scans container images, repos, filesystems, Kubernetes; detects vulnerabilities and misconfigurations [68]	Scan scope breadth can require policy tuning; database freshness and air-gapped workflows require planning (interpretation)	[69]
Syft	SBOM generation	Generates SBOMs from container images/filesystems; intended for use with scanners like Grype [70]	SBOM completeness depends on ecosystem detection; governance needed for consistent formats (interpretation)	[70]
Grype	Vulnerability scanning (incl. SBOM input)	Scans images, filesystems, and SBOMs for known vulnerabilities [71]	Accuracy depends on vulnerability DB sources and matching; requires risk-based prioritisation (interpretation)	[71]
Falco	Runtime detection (cloud-native)	Runtime security across hosts/containers/Kubernetes; rule-driven detection of abnormal behaviour [72]	Requires rule engineering and alert tuning to reduce noise; detection ≠ prevention unless integrated with response (interpretation)	[72]

Defense-in-depth reference architecture

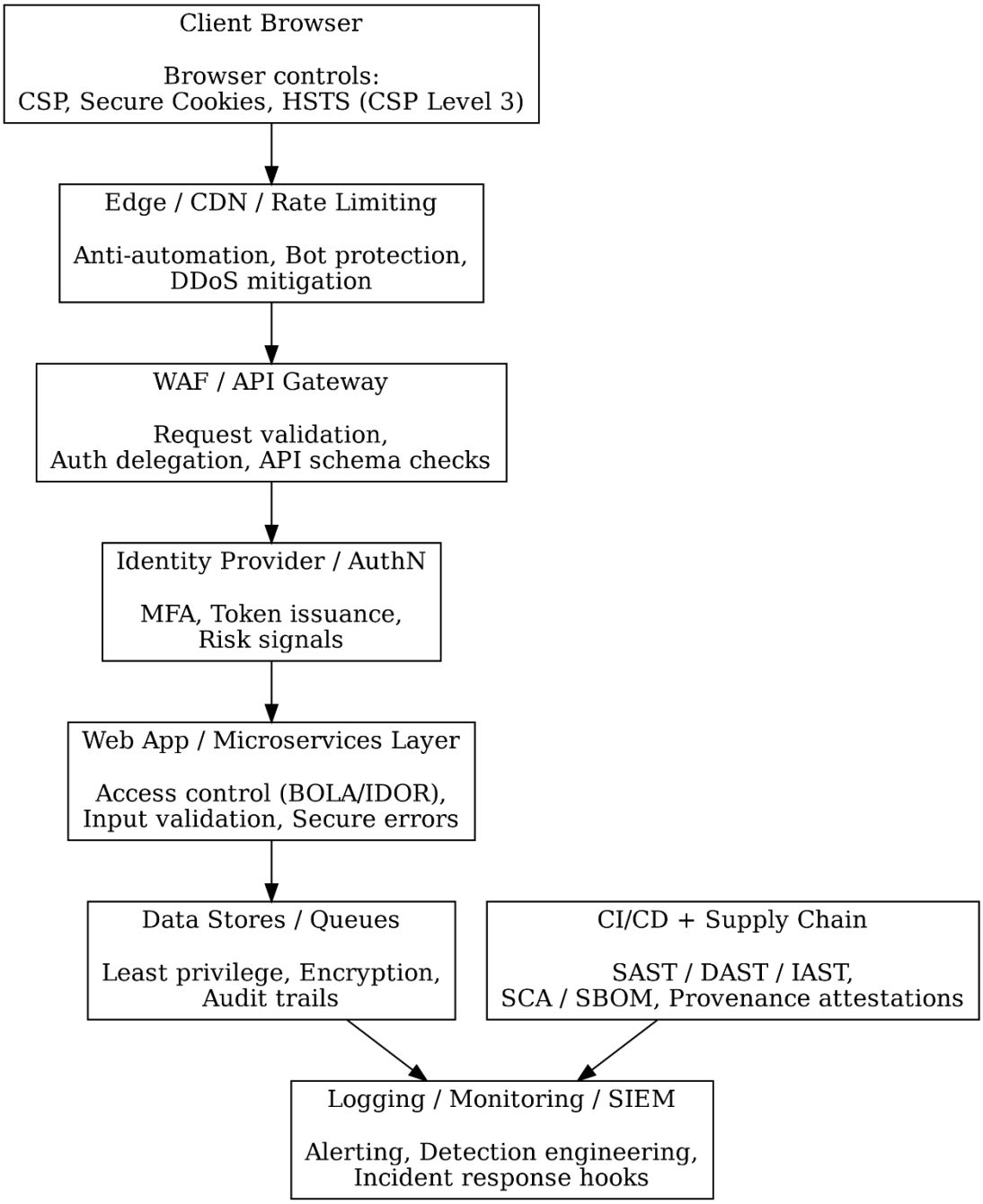

Figure 4 — Defense-in-depth for modern web applications (ASCII reference architecture) [73]

This architecture is based on NIST's focus on secure DevSecOps pipelines for cloud-native applications, its API protection lifecycle model, CSP for client-side risk mitigation, and OWASP session management guidelines for reducing account takeover risk impacts. [74]

Discussion

Synthesis of findings

Firstly, while vulnerability prevalence and breach prevalence are different metrics, both ultimately suggest that identity and boundary issues were a key factor in the results. OWASP's prevalence data suggests that access control and crypto issues manifest in a non-trivial percentage of tested applications. DBIR's data suggests that stolen credentials drive the majority of web application breaches in their "Basic Web Application Attacks" pattern. [22]

Secondly, exploit chains in modern breaches tend to be compositional. The MOVEit breach demonstrates how a single SQLi exploit can compose into a multi-step extortion attack. The Codecov breach demonstrates how a CI/CD attack can use trust relationships to distribute an attack without directly compromising production endpoints. The description of the xz/liblzma backdoor in the NVD suggests that "vulnerability" can be a backdoor intentionally added to a build artifact. [75]

Third, "shift-left" is necessary but not sufficient. Tooling (SAST, DAST, SCA, etc.) helps with detection, yet governance (SSDF, DevSecOps primitives, provenance), as well as runtime observability, must be considered for unknown unknowns and time-to-detection. Indeed, NIST's own DevSecOps publication clearly describes pipelines as end-to-end workflows: build, test, deploy, operate, mirroring our own empirical evidence of how attackers use both dev-time secrets and runtime vulnerabilities. [76]

Implications of the Study

Practical:

- Improves secure SDLC implementation
- Helps DevSecOps adoption

Industry:

- Guides tool selection & pipeline security

Policy:

- Supports compliance with NIST & OWASP

Academic:

- Provides integrated taxonomy + exploit modeling framework

Recommendations

For Developers:

Use parameterized queries + secure APIs
Implement strong authentication & MFA

For Organizations:

Adopt DevSecOps pipelines
Enforce SBOM and supply-chain security

For Security Teams:

Combine SAST + DAST + runtime detection
Prioritize identity and misconfiguration risks

Limitations

The 2018-2026 timeline is curated, not exhaustive, with a focus on the most exploited and web-relevant examples with available NVD records and CISA advisories, if applicable. Data for 2026 is necessarily limited as of March 2026. [77] The evaluation of tools is documentary and qualitative in nature, with peer-reviewed studies for some categories of tools, though a benchmark for tools, languages, applications, and architectures is still a research problem. [78]

Conclusion

Modern web application security has evolved beyond code-level security flaws to include identity systems together with your system configurations and supply-chain dependencies. Attacker tactics have changed according to empirical evidence that shows credential abuse accounts for 77 percent of security incidents while direct vulnerability exploitation only accounts for 13 percent. The unified framework from the proposed taxonomy together with the exploit chain models and mitigation mappings enables risk assessment and risk management activities. Security strategies need to combine development and deployment together with runtime defenses to effectively protect against emerging security threats.

Future work

Future research directions that would materially advance practice include:

(i) Reproducible, open benchmarks that measure SAST/DAST/IAST performance on modern architectures (SPAs, GraphQL, microservices).
(ii) More robust empirical mapping from supply-chain artefacts (SBOM + provenance + CI telemetry) to exploit likelihood and prioritization.
(iii) Integrated models that quantify "credential abuse + vulnerability exploitation" joint risk, reflecting DBIR's repeated observation that credentials remain the core breach enabler across years. [81]

REFERENCES

[1] Open Web Application Security Project[], "OWASP Top 10:2025 Introduction," 2025, accessed Mar. 12, 2026.
[2] OWASP, "OWASP Top Ten Web Application Security Risks (current: 2025)," accessed Mar. 12, 2026.
[3] OWASP, "OWASP Top 10 API Security Risks – 2023," accessed Mar. 12, 2026.
[4] National Institute of Standards and Technology, "SP 800-218: Secure Software Development Framework (SSDF)," accessed Mar. 12, 2026.
[5] NIST, "SP 800-204C: Implementation of DevSecOps for a Microservices-based Application," 2022, accessed Mar. 12, 2026.
[6] NIST, "SP 800-228: Guidelines for API Protection for Cloud-Native Systems," 2025, accessed Mar. 12, 2026.
[7] World Wide Web Consortium, "Content Security Policy Level 3," accessed Mar. 12, 2026.
[8] Blockchain-Secured Iot Framework for Smart Waste Management in Urban Environments", *CRSSS*, vol. 3, no. 3, pp. 1462–1467, Aug. 2025, doi: 10.59075/mcze1x98.
[9] OWASP Cheat Sheet Series, "Cross Site Scripting Prevention Cheat Sheet," accessed Mar. 12, 2026.
[10] VLSI Design Challenges in Nanotechnology and Future Transistor: https://doi.org/10.5281/zenodo.18980261", *AMAR*, vol. 4, no. 3, pp. 16–24, Mar. 2026, Accessed: May 04, 2026. [Online]. Available: https://amresearchjournal.com/index.php/Journal/article/view/1639
[11] OWASP Cheat Sheet Series, "Authorization Cheat Sheet," accessed Mar. 12, 2026.
[12] OWASP, "Content Security Policy Cheat Sheet," accessed Mar. 12, 2026.
[13] Verizon, "2024 Data Breach Investigations Report (DBIR)," 2024.
[14] Verizon, "2025 DBIR Manufacturing Snapshot," 2025.
[15] Cybersecurity and Infrastructure Security Agency, "#StopRansomware: CL0P Ransomware Gang Exploits MOVEit Transfer Vulnerability," 2023.
[16] NIST, "CVE-2018-7600 Detail," National Vulnerability Database, 2018.
[17] NIST, "CVE-2019-19781 Detail," National Vulnerability Database, 2019.
[18] NIST, "CVE-2020-5902 Detail," National Vulnerability Database, 2020.
[19] NIST, "CVE-2021-41773 Detail," National Vulnerability Database, 2021.
[20] NIST, "CVE-2021-42013 Detail," National Vulnerability Database, 2021.
[21] NIST, "CVE-2021-44228 Detail," National Vulnerability Database, 2021.
[22] NIST, "CVE-2022-22965 Detail," National Vulnerability Database, 2022.
[23] NIST, "CVE-2022-26134 Detail," National Vulnerability Database, 2022.
[24] NIST, "CVE-2023-34362 Detail," National Vulnerability Database, 2023.
[25] NIST, "CVE-2023-4966 Detail," National Vulnerability Database, 2023.
[26] NIST, "CVE-2024-3094 Detail," National Vulnerability Database, 2024.
[27] NIST, "CVE-2025-5777 Detail," National Vulnerability Database, 2025.
[28] CISA, "CISA Adds One Known Exploited Vulnerability to Catalog (CVE-2025-5777)," 2025.

[29] Progress Software, "MOVEit Transfer Critical Vulnerability (CVE-2023-34362) advisory," 2023.
[30] Codecov[110], "Post-Mortem / Root Cause Analysis (April 2021)," 2021.
[31] Codecov, "Bash Uploader Security Update," 2021.
[32] CISA, "Codecov releases new detections for supply chain compromise," 2021.
[33] S. Khan et al., "A Systematic Analysis of the Capital One Data Breach," ACM (full text), 2022.
[34] CYBERSECURITY CHALLENGES AND DEFENSE MECHANISMS IN AUTONOMOUS VEHICLES: PROTECTING CONNECTED AND INTELLIGENT TRANSPORTATION SYSTEMS FROM EMERGING CYBER THREATS", *Kashf J. Multidiscip. Res.*, vol. 3, no. 03, pp. 36–52, Mar. 2026, doi: 10.71146/kjmr857.
[35] National Telecommunications and Information Administration[112], "The Minimum Elements for a Software Bill of Materials (SBOM)," 2021.
[36] Open Source Security Foundation, "SLSA specification v1.0," 2023 (status noted on spec page), accessed Mar. 12, 2026.
[37] OWASP, "OWASP Dependency-Check Project," accessed Mar. 12, 2026.
[38] OWASP Developer Guide, "Dependency-Check (SCA) guidance," accessed Mar. 12, 2026.
[39] Trivy Project, "Trivy documentation: vulnerability scanning and targets," accessed Mar. 12, 2026.
[40] Muhammad Ahsan Hayat, Imran Ali Channa, Ubaidullah Khan, Nazia Alfred Fernandes, Urooj Tariq, and Khan Ikram Uddin, "BRIDGING CLASSICAL STATISTICS AND MODERN AI TOWARD INTERPRETABLE DATA-SCIENCE MODELS", *SES*, vol. 3, no. 9, pp. 525–537, Sep. 2025.
[41] Anchore, "Grype (vulnerability scanning) repository," accessed Mar. 12, 2026.
[42] OWASP ZAP Project, "ZAP Getting Started / automation guidance," accessed Mar. 12, 2026.
[43] PortSwigger, "Burp Scanner documentation," accessed Feb. 23, 2026.
[44] GitHub, "CodeQL documentation and query analysis workflow," accessed Mar. 12, 2026.
[45] The Role of HR in Managing Robotic Process Automation (RPA) Displacement Anxiety among Employees", *CRSSS*, vol. 3, no. 3, pp. 1090–1109, Aug. 2025, doi: 10.59075/f4y5dc30.
[46] Falco Project, "Falco documentation," accessed Mar. 12, 2026.
[47] Office of the Comptroller of the Currency, "OCC assesses $80 million civil money penalty against Capital One," 2020.
[48] Cloud Native Computing Foundation, "Falco project status and lifecycle," accessed Mar. 12, 2026.
[49] OWASP Cheat Sheet Series, "SQL Injection Prevention Cheat Sheet," accessed Mar. 12, 2026.
[50] BlockVerse: A Decentralized Mobile App for Identity and Access Management in the Metaverse", *AMAR*, vol. 4, no. 4, pp. 438–451, Apr. 2026, doi: 10.66021/.
[51] Sentiment Analysis and Demand Forecasting from Patient Reviews", *AMAR*, vol. 4, no. 4, pp. 452–459, Apr. 2026, doi: 10.66021/.
[52] Muhammad Ahsan Hayat, Syed Affan Ahmed, Sana Fatima, Engr. Faiza Irfan, Muhammad Osama Nizamani, and Ammar Khalil, "TINY MACHINE LEARNING (TINYML) ADVANCEMENTS FOR INTELLIGENT BATTERY-POWERED IOT SENSORS", *SES*, vol. 3, no. 8, pp. 818–832, Aug. 2025.
[53] Anchore, "Syft (SBOM generation) repository," accessed Mar. 12, 2026.

[54] Muhammad Ahsan Hayat, Jahangir Baig, Shayan Ahmed, and Ahmed Faraz Ayubi, "TOWARDS EARLY AND ACCURATE DISEASE DETECTION THROUGH MULTIMODAL PREDICTIVE MODELING: FUSION OF ELECTRONIC HEALTH RECORDS, MEDICAL IMAGING, AND OMICS DATA USING INTERPRETABLE MACHINE LEARNING", *PRJ*, vol. 3, no. 11, pp. 01–29, Nov. 2025.

[55] Musadique Hussain, "META-LEARNING FOR FEW-SHOT AND ZERO-SHOT LEARNING: ENABLING RAPID ADAPTATION TO NEW TASKS WITH LIMITED DATA", *PRJ*, vol. 3, no. 11, pp. 653–669, Nov. 2025.

[56] OWASP Cheat Sheet Series, "Authentication Cheat Sheet," accessed Mar. 12, 2026.