BLACKDUCK®

2024
Software Vulnerability Snapshot

Insights into Critical Vulnerabilities from over 200,000
Application Security Scans by Black Duck

Table of contents

Executive Summary

This report analyzes data from over 200,000 dynamic application security testing (DAST) scans conducted by Black Duck on approximately 1,300 applications across 19 industry sectors from June 2023 to June 2024.

The findings provide insights into the current state of security for web-based applications and systems, and the potential impact of security vulnerabilities on business operations in high-risk sectors such as Finance, Insurance, and Healthcare.

The report also examines how DAST offers a crucial complement to other security testing methods, such as static application security testing (SAST) and software composition analysis (SCA), and provides a unique perspective on application security by mimicking real-world attack scenarios.

About Black Duck

Formerly the Synopsys Software Integrity Group, Black Duck offers the most comprehensive, powerful, and trusted portfolio of AppSec solutions in the industry. We have an unmatched track record of helping organizations secure their software quickly, integrate security efficiently in their development environments, and safely innovate with new technologies.

Key Findings

The Vulnerability Landscape

A total 96,917 vulnerabilities were identified in scans conducted 2023–24. These are the top critical-risk vulnerabilities identified.

Cryptographic Failures (Sensitive Data Exposure)
These are weaknesses in how an application secures sensitive information. This category includes issues like not encrypting important data when it's being sent over the internet, using outdated or weak encryption methods, and failing to properly protect passwords or other secret information. These failures can lead to data breaches, where attackers can steal or tamper with sensitive information such as personal details, financial data, or login credentials.

This category of weakness was found to be widespread in our DAST analysis, affecting 86% of clients and accounting for 30,726 vulnerabilities, including 4,882 critical-risk instances. This makes it one of the most common and serious security issues across industries. To address these vulnerabilities, organizations need to implement strong encryption practices, use up-to-date security protocols, and ensure that sensitive data is properly protected both when it's being transmitted and when it's stored.

Injection Vulnerabilities
This is a type of security vulnerability that allows an attacker to insert malicious code or commands into an application, tricking it into executing unintended actions or accessing data without proper authorization. The analysis found 4,814 Injection vulnerabilities, with a high prevalence of 59% per client. This category had the second-highest number of critical vulnerabilities (2,491), indicating its potential for causing severe security breaches.

Injection vulnerabilities often occur when user input is not properly validated or sanitized before being used in database queries, operating system commands, or web page content. Common Injection attacks include SQL Injection, Command Injection, and Cross-Site Scripting (XSS), with successful attacks leading to data theft, unauthorized data manipulation, or even full system compromise.

To prevent Injection vulnerabilities, organizations need to implement proper input validation, use parameterized queries, and follow secure coding practices. While both SAST and DAST can detect Injection vulnerabilities, DAST is particularly effective at identifying complex, runtime-dependent issues. Regular security testing, especially using DAST, can help identify and address these vulnerabilities.

Industry-Specific Insights

High-risk sectors included Finance and Insurance (1,299 critical vulnerabilities), Healthcare and Social Assistance (992 critical vulnerabilities), and Information Services (446 critical vulnerabilities). The Finance and Insurance industry (FSI) had the highest number of critical vulnerabilities across all site complexities, with 565 critical vulnerabilities identified for small FSI sites, 580 for medium sites, and 154 for large sites. The next-highest industry was Healthcare and Social Assistance, with 367, 486, and 139 critical vulnerabilities for small, medium, and large sites respectively.

The data indicates that small and medium-sized sites tend to have more critical vulnerabilities than larger sites, particularly in the FSI sector.

Time-to-Close Analysis

The data shows significant variations across industries when it came to vulnerability time-to-close. For critical vulnerabilities, the Utilities industry had the longest time-to-close across all sites. The extended time-to-close for small (107 days) and medium (876 days) sites versus larger (1 day) in the Utilities sector may be due to limited cybersecurity resources and budget constraints. Utilities often operate with legacy systems that are difficult to patch and update. Large sites might have dedicated security teams and more robust processes, allowing them to address vulnerabilities more quickly.

The next-longest time-to-close was the Educational Services sector with closure times as 342 days for small sites, 111 days for medium sites, and 1 day for large sites. Small educational institutions often face budget limitations and may lack dedicated cybersecurity personnel, leading to longer times to address vulnerabilities. Large educational institutions such as major universities, however, are likely to have better-funded IT departments and more resources to quickly mitigate critical vulnerabilities.

Conversely, Finance and Insurance closed critical vulnerabilities for small sites in just 28 days, medium sites in 53 days, and large sites in 78 days. This sector is heavily regulated and deals with highly sensitive data, necessitating a rapid response to vulnerabilities. These organizations typically have substantial cybersecurity budgets and dedicated teams to ensure compliance with regulations like Payment Card Industry Data Security Standard (PCI DSS) and to protect financial data.

Organizations in the Healthcare and Social Assistance sector took an average of 87 days to close critical vulnerabilities for small sites, 30 days for medium sites, and 20 days for large sites. The Healthcare sector is also highly regulated (e.g., the Health Insurance Portability and Accountability Act [HIPAA]) and handles sensitive patient data, which drives the need for prompt vulnerability remediation. Larger Healthcare organizations often have more resources and dedicated security teams, enabling faster closure times.

The variations in time-to-close metrics across different sectors highlight the impact of resource allocation and the challenges legacy systems can have on security initiatives. Sectors with significant regulatory pressures and sensitive data tend to act swiftly to mitigate vulnerabilities, reflecting their proactive stance. On the other hand, sectors with limited resources and budget constraints face longer exposure times, underscoring the need for tailored cybersecurity strategies and increased investment in under-resourced industries.

Black Duck analysts use a proprietary metric to rank the relative “site complexity” of applications assessed by Black Duck® Continuous Dynamic scanning. This metric is based on the number and sophistication of interactions performed during the scanning process. Applications with less complexity may have minimal interactivity and a simple crawl tree—that is, an application with a straightforward structure of URLs. Higher-complexity applications may have many interactive elements and dynamically generated content.

This metric allows our specialists to customize scan behaviors, adjusting the depth and aggression of scans based on the complexity of the application. The complexity metric can also be weighted across industries for comparison and baselining.

Potential Business Impact Suggested by the Data

These are some of the risks of these vulnerabilities.

Data Breaches: Sensitive Data Exposure and Injection vulnerabilities pose significant threats to sensitive data across all industries, potentially leading to data leaks, fines, financial losses, and reputational damage. Sensitive data at risk includes personally identifiable information such as Social Security numbers, banking information, login credentials, credit card numbers, medical records, and trade secrets.

Regulatory Noncompliance: High-risk sectors face increased exposure to noncompliance with data protection regulations, risking severe penalties. For example, the Cyber Incident Reporting for Critical Infrastructure Act of 2022 (CIRCIA) applies to critical infrastructure sectors (including FSI, Healthcare, and Waste Management among others) and requires covered entities to report covered cyber incidents and ransomware payments to CISA. PCI DSS applies to all organizations that handle credit card information and mandates security standards for protecting cardholder data. HIPAA requires protection of patient health information and mandates reporting of data breaches. For organizations handling the data of EU citizens, the General Data Protection Regulation (GDPR) requires strict data protection measures and breach reporting.

Operational Disruptions: Widespread security misconfigurations and Denial-of-Service vulnerabilities across industry sectors threaten business continuity and service availability. Operational disruptions caused by vulnerabilities and misconfigurations can have significant consequences across sectors. In the Healthcare sector for example, some of the potential critical disruptions include

- **Disruption of Life-Saving Equipment:** A cyberattack that targets a hospital's network could lead to the shutdown of critical medical devices such as ventilators, infusion pumps, and heart monitors. Patients relying on these devices for life support could face immediate life-threatening situations. For example, if ventilators are turned off, patients who cannot breathe independently may suffer severe health consequences or even death.
- **Compromise of Electronic Health Records:** A ransomware attack that encrypts patient records would make them inaccessible to healthcare providers. The inability to access patient histories, medication records, and treatment plans could lead to delays in care, incorrect treatments, and medication errors. This can severely impact patient outcomes, particularly in emergency situations where timely access to accurate information is critical.
- **Medication Errors Due to Pharmacy System Interruptions:** An exploited vulnerability in a pharmacy system could cause the system to go offline. Interruptions in the pharmacy system can lead to delays in dispensing medications, incorrect dosages, or missed treatments. This can be particularly dangerous for patients with critical conditions requiring precise medication management.

Extended Vulnerability Exposure: Long closure times in sectors like Utilities and Educational Services increase the risk of exploitation and potential business impact. For example, a vulnerability in the power grid control system that remains unpatched could lead to prolonged power outages affecting millions of households and businesses. In extreme cases, it could result in cascading failures across interconnected power systems, potentially causing blackouts across entire regions. A vulnerability in a water treatment plant's control system going unaddressed could potentially alter chemical treatment processes, leading to water contamination. This could result in widespread illness, the need for extensive system flushing, and a loss of public trust in water safety.

In the Educational Services sector, an unaddressed vulnerability in a student information system could lead to the exposure of sensitive student data, including personal information, academic records, and financial details. Such a breach could result in identity theft, academic fraud, and violation of privacy laws like FERPA, leading to legal consequences and loss of trust in the institution.

Recommendations

Based on the findings data from the over 200,000 scans conducted by Black Duck, organizations should prioritize addressing Sensitive Data Exposure (called Cryptographic Failures in the OWASP 2021 taxonomy) and Injection vulnerabilities, including SQL Injection and Cross-Site Scripting, especially in high-risk sectors.

Organizations in all sectors should focus on reducing time-to-close for critical vulnerabilities, particularly in sectors that permit long remediation times. Security misconfigurations across all industries should be addressed to minimize potential information disclosure and reputational damage.

Overall, development and security teams should implement a multifaceted security approach integrating DAST, SAST, and SCA to achieve the most comprehensive coverage throughout the software development life cycle. The findings indicate that if such a full spectrum approach to application security testing were applied, potential exposure to critical vulnerabilities would be markedly reduced.

Industry Sectors Represented in This Report

Agriculture, Forestry, Fishing and Hunting

Mining/Quarrying, Oil/Gas Extraction

Construction

Wholesale Trade

Real Estate Rental and Leasing

Management of Companies & Enterprises

Manufacturing

Administrative Support & Waste Management

Accommodation and Food Services

Arts, Entertainment, and Recreation

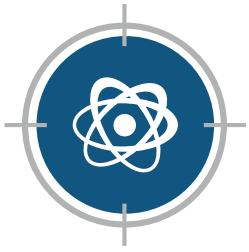
Educational Services

Finance and Insurance

Healthcare and Social Assistance

Information Services

Other Services

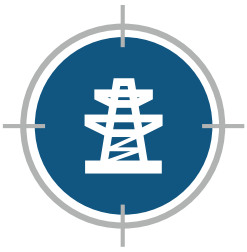


Professional, Scientific, and Technical Services

Public Administration

Retail Trade

Transportation and Warehousing



Utilities

Fundamentals of Dynamic Application Security Testing

DAST is a critical component in the application security landscape, particularly as organizations grapple with increasingly complex and vulnerable web applications. This section provides a comprehensive overview of DAST, its significance, and its role in a robust application security strategy.

Key Characteristics of DAST

DAST is a black-box security testing methodology that analyzes applications in their running state, without any privileged access to that application's design, architecture, or internals. Unlike SAST, which examines source code, DAST simulates real-world attacks on a live application, identifying vulnerabilities that may manifest only during runtime or in interactions between multiple subsystems. This approach allows DAST to detect issues such as authentication problems, server configuration errors, and Cross-Site Scripting vulnerabilities that might be missed by other testing methods.

DAST in the Modern Security Landscape

The relevance of DAST has grown significantly due to several factors.

- **The increasing complexity of web applications:** With the rise of the microservices architecture, API-driven development, and cloud-native applications, attack surfaces have expanded considerably.
- **Evolving cyberthreats:** As attackers become more sophisticated, DAST's ability to simulate real-world attacks becomes invaluable.
- **Regulatory compliance:** Regulatory frameworks like GDPR and PCI DSS require robust, continuous security testing, making DAST an essential tool for compliance.
- **DevSecOps integration:** DAST's ability to be integrated within continuous integration/continuous deployment (CI/CD) pipelines aligns with modern DevSecOps practices.
- **Cost implications:** Early detection of vulnerabilities can significantly reduce the cost of fixing security issues after deployment.

DAST and Other Testing Methodologies

While DAST is powerful, it's most effective when used in conjunction with other security testing methods. For example, SAST's strengths include early detection of coding flaws, but it may miss runtime vulnerabilities or unintended interactions between components, which can be found by DAST. Likewise, SCA identifies vulnerabilities in third-party components; DAST can verify if these vulnerabilities are exploitable in a running application.

By implementing DAST in conjunction with other testing methodologies, organizations can significantly enhance their security posture in a complex online application landscape.

The key characteristics of DAST include

- External perspective testing, mimicking an attacker's view
- Visibility into trending behaviors
- Runtime analysis of applications
- Continuous testing
- Ability to test without access to source code

DAST in Preproduction and Production

DAST solutions can be integrated into CI/CD pipelines to identify vulnerabilities early, accelerating remediation and reducing the cost of fixes. This approach is particularly valuable for detecting issues that may manifest only in a running application, such as certain types of Injection vulnerabilities or unexpected interactions between services and components.

DAST solutions used in production offer additional benefits, especially for organizations dealing with complex, dynamic applications or those in highly regulated industries. This can also provide continuous monitoring, detecting vulnerabilities that may arise due to configuration changes, newly discovered exploits, or changes in the application's runtime environment. It's particularly valuable for identifying issues with third-party components that may become vulnerable over time, and for verifying the effectiveness of patches.

A combination of preproduction and production DAST testing may provide the most comprehensive security coverage for some organizations. The ideal scenario in these cases is implementing extensive preproduction DAST testing, and using production testing as a supplementary measure rather than the primary security strategy. The combination offers benefits including

- **Risk mitigation:** Preproduction testing eliminates the risk of unplanned downtime or data corruption in live environments.
- **Cost-effectiveness:** Fixing vulnerabilities earlier in the development cycle is typically much less expensive than addressing them in production.
- **Developer-friendliness:** Integration with IDEs and CI/CD pipelines makes security testing a natural part of the development process.
- **Compliance:** Many regulatory standards prefer or require testing before production deployment.
- **Comprehensive testing:** Preproduction environments enable more thorough and aggressive testing without fear of impacting users or data.

DAST testing in production can also be beneficial in several scenarios.

- **Continuous security validation:** It can act as an additional layer of security to catch any issues that might have slipped through preproduction testing.
- **Detection of emerging threats:** DAST can automatically detect emerging threats.
- **Cloud-native applications:** It is useful in production environments that may have unique configurations that are difficult to replicate in testing.
- **Legacy systems:** Production DAST may be more effective dealing with older applications that lack comprehensive preproduction environments.

The strategic implementation of DAST in both preproduction and production environments offers a balanced approach, allowing for thorough testing without compromising system integrity or user experience. This multifaceted strategy not only enhances vulnerability detection across various stages of the application life cycle but also addresses the unique challenges posed by diverse application architectures, from legacy systems to cloud-native applications. Ultimately, this approach enables organizations to build a more resilient security posture capable of adapting to the evolving threat landscape.

Vulnerability Landscape Analysis

The analysis of over 200,000 DAST scans across approximately 1,300 applications reveals a concerning vulnerability landscape. This section delves into the most prevalent and critical vulnerabilities, their distribution across industries, and their potential impact on business operations.

Top 10 Vulnerability Classes Identified

	Vulnerability Class	Description	Vulnerabilities Identified
1	**Insufficient Transport Layer Protection**	Failure to properly encrypt data in transit, allowing interception and tampering.	30,712
2	**Missing Secure Headers**	Absence of important HTTP security headers that help protect against various web-based attacks. Examples include X-XSS-Protection, X-Frame-Options, and Content-Security-Policy.	22,321
3	**Information Leakage**	Unintentional exposure of sensitive information through error messages, comments, or other application responses that can be leveraged by attackers to gain insights into the system's architecture or vulnerabilities.	8,097
4	**Predictable Resource Location**	Resources or files being stored in locations that can be easily guessed or predicted by attackers, potentially allowing unauthorized access to sensitive information or functionality.	5,468
5	**Frameable Resource**	A vulnerability in which a web page can be embedded in an iframe on another site, potentially leading to clickjacking attacks. This occurs when the X-Frame-Options header is missing or improperly configured.	4,481
6	**Vulnerable Library**	The use of third-party libraries or components with known security vulnerabilities, which can introduce weaknesses into the application that uses them.	4,215
7	**Fingerprinting**	Attackers gaining information about the technology stack, versions, or configurations of a system, which can be used to identify potential vulnerabilities or plan more-targeted attacks.	3,700
8	**Cross-Site Scripting**	A vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users, potentially leading to theft of sensitive information or session hijacking.	2,415
9	**Insufficient Authorization**	Inadequate access controls that allow users to perform actions or access resources beyond their intended privileges, often due to improper implementation of authorization checks.	2,396
10	**Susceptibility to Brute Force**	An attack method in which attackers systematically attempt many passwords or passphrases with the hope of eventually guessing correctly, often exploiting weak password policies or lack of account lockout mechanisms.	2,235

Figure 1. Vulnerability frequency

Our analysis identified a total of 96,917 vulnerabilities. Figure 1 shows the vulnerabilities identified most frequently, and Figure 2 lists the percentage of clients with each vulnerability.

Insufficient Transport Layer Protection, the most prevalent issue, exposes organizations to data interception and tampering, potentially leading to data breaches and compliance violations. Missing Secure Headers, the second-most-common vulnerability, leaves applications susceptible to various web-based attacks, undermining overall security posture.

Information Leakage and Predictable Resource Location represent significant vulnerabilities that often provide attackers with an easy entry point into systems. Information Leakage, ranked third with 8,097 identified vulnerabilities, involves the unintentional exposure of sensitive information through error messages, comments, or application responses. This can provide attackers with valuable insights into system architecture or vulnerabilities.

Predictable Resource Location, fourth with 5,468 vulnerabilities, occurs when resources or files are stored in easily guessable locations, potentially allowing unauthorized access to sensitive information or functionality.

Both these vulnerability classes highlight a common issue: the inadvertent disclosure of information that can be leveraged by attackers. These vulnerabilities are particularly concerning because they often result from oversight or inadequate security practices rather than complex technical issues, making them both common, and, at least theoretically, more straightforward to address with proper security awareness and protocols.

Vulnerability Class	Percentage of Clients with Vulnerability
Missing Secure Headers	97%
Insufficient Transport Layer Protection	87%
Information Leakage	66%
Frameable Resource	60%
Vulnerable Library	57%
Insufficient Authorization	50%
Fingerprinting	46%
Cross-Site Scripting	41%
Improper Input Handling	40%
Predictable Resource Location	35%

Figure 2. Percentage of clients experiencing one or more of a given vulnerability class

Critical-Risk and Urgent Vulnerabilities

Vulnerability Class	Critical-Risk Vulnerabilities	Urgent Need of Attention
Insufficient Transport Layer Protection	4,882	2
Cross-Site Scripting	2,256	1
Information Leakage	510	-
Cross-Site Request Forgery	434	-
Abuse of Functionality	248	36
HTTP Response Splitting	203	-
Content Spoofing	76	-
Path Traversal	73	-
Insufficient Password Policy Implementation	67	-
Insufficient Process Validation	61	-

Figure 3. Critical-risk and urgent vulnerabilities

Among the identified vulnerabilities, several stand out as critical-risk or requiring urgent attention. These include Insufficient Transport Layer Protection: (4,882 critical instances), Cross-Site Scripting (2,256 critical instances), and Information Leakage (510 critical instances).

The high number of critical Insufficient Transport Layer Protection vulnerabilities is particularly alarming, as it indicates widespread exposure to potential data breaches. Cross-Site Scripting vulnerabilities, while fewer in number, pose a significant threat due to their potential for session hijacking and data theft.

Abuse of Functionality is notable for its high number of critical vulnerabilities in urgent need of attention. Abuse of Functionality vulnerabilities often involve misuse of legitimate features of an application. This means they can be exploited immediately without requiring complex ways to bypass security measures. These vulnerabilities can affect core functionalities of an application, potentially impacting a wide range of users or critical business processes. They can sometimes lead to privilege escalation or access to sensitive data, making them critical to address quickly.

OWASP Top 10 Category Analysis

OWASP Category	Vulnerabilities Identified	Contributing CWEs
A01:2021 – Broken Access Control	9,954	CWE-35, CWE-548, CWE-287, CWE-352, CWE-425, CWE-601, CWE-424
A02:2021 – Cryptographic Failures	30,726	CWE-319, CWE-330, CWE-311
A03:2021 – Injection	4,814	CWE-943, CWE-91, CWE-20, CWE-643, CWE-113, CWE-77, CWE-89, CWE-610, CWE-652, CWE-94, CWE-79, CWE-78, CWE-97, CWE-90
A04:2021 – Insecure Design	7,581	CWE-799, CWE-840, CWE-525, CWE-1021
A05:2021 – Security Misconfiguration	36,321	CWE-497, CWE-550, CWE-209, CWE-611, CWE-525, CWE-200, CWE-703, CWE-202, CWE-693, CWE-16, CWE-1004, CWE-614, CWE-544
A06:2021 – Vulnerable and Outdated Components	4,215	CWE-1104
A07:2021 – Identification and Authentication Failures	1,057	CWE-521, CWE-640, CWE-613, CWE-285, CWE-384
A08:2021 – Software and Data Integrity Failures	1,929	CWE-345, CWE-451, CWE-148, CWE-829
A10:2021 – Server-Side Request Forgery	1	CWE-918
A11:2021 – Denial-of-Service	319	CWE-400

Figure 4. Alignment with OWASP Top 10 categories and contributing CWEs

Aligning our findings with OWASP Top 10 categories provides several insights.

A02:2021 – Cryptographic Failures/Sensitive Data Exposure: 30,726 vulnerabilities, including 4,882 critical-risk instances. This category had the second-highest prevalence per client (86%), indicating widespread issues with adequately protecting sensitive data. Contributing CWEs for this category include

- CWE-319: This CWE refers to an application transmitting sensitive information in cleartext, which can be intercepted and read by an attacker.
- CWE-330: This CWE refers to an application using predictable values in a context that requires unpredictability, such as random passwords or cryptographic keys.
- CWE-311: This CWE refers to an application failing to encrypt sensitive data before storage or transmission, leaving it vulnerable to interception or unauthorized access.

A03:2021 – Injection: 4,814 vulnerabilities, with 2,491 critical instances and a high prevalence per client (59%). This underscores the persistent threat of Injection attacks across various applications.

Contributing CWEs include CWE-943, CWE-91, CWE-20, CWE-643, CWE-113, CWE-77, CWE-89, CWE-610, CWE-652, CWE-94, CWE-79, CWE-78, CWE-97, and CWE-90. Many of these CWEs are related to attacks in which code, commands, or data is injected into an application, potentially causing the application to execute unintended commands or access data without proper authorization.

Injection attacks, such as SQL and Command Injections, rank high in the CWE list of most dangerous vulnerabilities. These CWEs include CWE-91, CWE-77, CWE-89, CWE-94, CWE-79, CWE-78, CWE-97, and CWE-90.

Other CWEs contributing to the Injection category include CWE-610, CWE-652, CWE-643, and CWE-943, all related to improper neutralization of data or special elements, which can lead to unintended behavior.

A05:2021 – Security Misconfiguration: 36,321 vulnerabilities identified, with the highest prevalence per client (98%).

The bulk (84%) of the identified security misconfiguration vulns were termed "informational" vulnerabilities by Black Duck experts. That is, the configuration has potential to disclose sensitive information but does not pose a specific security risk to the environment, host, or application.

However, this information can include such details as installed software, open ports, and general information about a system and how it operates, possibly opening a doorway to exploit. At a minimum, organizations should be aware of the existence of these proto vulnerabilities to make an informed decision on whether configuration changes should be made to strengthen security.

Industry-Specific Vulnerability Trends

The vulnerability landscape varies significantly across industries: Finance and Insurance had the highest number of critical vulnerabilities (1,299), indicating substantial risk in this highly regulated sector. Healthcare and Social Assistance was the second-highest with 992 critical vulnerabilities, raising concerns about patient data protection and regulatory compliance. Information Services had a total of 446 critical vulnerabilities, highlighting the need for robust security in data-centric industries.

Notably, small and medium-complexity sites tended to have more critical vulnerabilities than larger sites, particularly in the Finance and Insurance sector. This trend suggests that organizations may be underestimating the security needs of smaller sites or less complex applications.

The implications of these findings are significant. For the Finance and Insurance sector, the high number of critical vulnerabilities could lead to financial losses, regulatory penalties, and reputational damage. In Healthcare, vulnerabilities could compromise patient data, violating HIPAA and other regulations. Information Services companies face risks of data breaches that could undermine client trust and business operations.

The prevalence of basic security issues like Insufficient Transport Layer Protection across industries indicates a need for fundamental improvements in security practices. Organizations should prioritize addressing these vulnerabilities to protect sensitive data, maintain regulatory compliance, and safeguard their operations and reputation.

Figure 5 shows that small sites often had the highest number of critical vulnerabilities and vary widely in closure times. Medium-sized sites frequently had moderate to high vulnerability counts with variable closure times. Large sites generally had fewer critical vulnerabilities and often close them faster.

Industry	Site Complexity	Critical Vulnerabilities	Time-to-Close Critical Vulnerabilities (days)
Agriculture, Forestry, Fishing and Hunting	Small	2	-
	Medium	0	
	Large	0	
Mining/Quarrying, and Oil/Gas Extraction	Small	0	
	Medium	0	
	Large	0	
Construction	Small	3	234
	Medium	2	150
	Large	1	-
Wholesale Trade	Small	11	20
	Medium	1	1
	Large	0	
Real Estate Rental and Leasing	Small	27	413
	Medium	4	158
	Large	7	1
Management of Companies and Enterprises	Small	2	397
	Medium	2	9
	Large	0	-
Manufacturing	Small	66	14
	Medium	22	248
	Large	2	-
Administrative Support and Waste Management	Small	3	5
	Medium	0	-
	Large	1	-
Accommodation and Food Services	Small	30	1
	Medium	33	1
	Large	5	1
Arts, Entertainment, and Recreation	Small	62	34
	Medium	87	30
	Large	15	63

Figure 5. Critical vulnerabilities identified and time-to-close by industry site complexity

Industry	Site Complexity	Critical Vulnerabilities	Time-to-Close Critical Vulnerabilities (days)
Educational Services	Small	72	342
	Medium	35	111
	Large	69	1
Finance and Insurance	Small	565	28
	Medium	580	53
	Large	154	78
Healthcare and Social Assistance	Small	367	87
	Medium	486	30
	Large	139	20
Information Services	Small	235	132
	Medium	140	37
	Large	71	111
Other Services	Small	65	38
	Medium	54	87
	Large	4	43
Professional, Scientific, and Technical Services	Small	124	90
	Medium	94	36
	Large	12	240
Public Administration	Small	15	2
	Medium	40	2
	Large	12	9
Retail Trade	Small	341	63
	Medium	29	17
	Large	6	1
Transportation and Warehousing	Small	21	1
	Medium	47	13
	Large	3	221
Utilities	Small	28	107
	Medium	23	876
	Large	4	1

Figure 5. (cont.) Critical vulnerabilities identified and time-to-close by industry site complexity

The Interplay of DAST, SAST, and SCA

In today's complex cybersecurity landscape, a comprehensive approach to application security is crucial. This section explores the interplay between DAST, SAST, and SCA, highlighting how these methodologies complement each other to provide robust security coverage.

Comparative Strengths in Detecting Specific Vulnerabilities

DAST

- **Runtime Analysis:** DAST excels at identifying vulnerabilities that manifest only when the application is running. This includes issues that depend on the application's runtime behavior, such as reflected and DOM-based Cross-Site Scripting vulnerabilities.
- **Black-Box Real-World Attack Simulation:** DAST simulates real-world attacks by interacting with the application as an external attacker would. This uncovers vulnerabilities that static analysis might miss, such as SQL Injection vulnerabilities that arise from dynamic query construction or complex application logic.
- **Detection of Complex Injection Vulnerabilities:** While SAST can detect many Injection vulnerabilities, DAST is particularly effective at identifying those that are dependent on runtime conditions. For example, DAST can more effectively detect LDAP Injection vulnerabilities by interacting with the application and observing its responses.
- **Verification of Exploitability:** DAST can confirm whether vulnerabilities identified by SCA in third-party components are exposed and exploitable in the running application.
- **Continuous Monitoring:** DAST can be implemented as a continuous process in production environments, allowing for real-time detection of newly introduced vulnerabilities. This continuous monitoring is crucial for identifying issues in third-party components that may become vulnerable over time, and for verifying the effectiveness of patches in the running application.
- **Comprehensive Coverage of Web-Based Attacks:** DAST can detect a wide range of web-based vulnerabilities, including Insufficient Transport Layer Protection, Missing Secure Headers, Information Leakage, and more. This comprehensive coverage helps ensure that applications are protected against a variety of attack vectors.

SAST

- **Early Detection:** SAST can catch potential coding weaknesses early in the development process, before the application is in a running state.
- **Code-Level Analysis:** SAST can identify issues like Improper Output Encoding, which can lead to vulnerabilities such as Cross-Site Scripting.
- **Comprehensive Code Coverage:** SAST can analyze the entire codebase, including parts that might not be easily accessible during runtime testing.
- **Identification of Certain Injection Vulnerabilities:** While DAST is often more effective for runtime-specific Injection issues, SAST can detect many instances of SQL Injection vulnerabilities, especially those that are apparent in the source code.
- **Detection of Coding Flaws:** SAST is particularly good at identifying coding errors that could lead to security vulnerabilities.
- **Secure Coding Practices:** SAST can help enforce secure coding standards and best practices by identifying deviations from these standards early in the development process.
- **Cost-Effective for Early Fixes:** By identifying issues early in the development cycle, SAST allows for more cost-effective remediation of vulnerabilities.

Best Practices for Integrating DAST, SAST, and SCA

- Use SAST and SCA early and often in the development process to catch potential coding weaknesses or vulnerabilities introduced by third-party software.
- Implement DAST to test applications in preproduction environments and identify vulnerabilities that may only be apparent during execution.
- Prioritize vulnerabilities based on their criticality and exploitability in the running application.

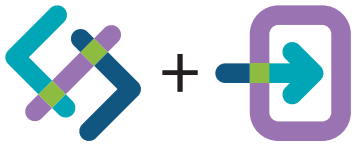
SCA

- **Identification of Known Vulnerabilities:** SCA excels at identifying known vulnerabilities in third-party libraries and open source components. By cross-referencing the components used in an application against databases of known vulnerabilities, SCA can quickly highlight potential risks.
- **Risk Prioritization:** SCA tools often include risk-scoring mechanisms that help prioritize vulnerabilities based on their severity and potential impact. This allows organizations to focus their remediation efforts on the most critical issues first.
- **Detailed Inventory of Dependencies:** SCA provides a comprehensive inventory of all third-party components and the versions used within an application. This detailed inventory helps organizations understand their dependency landscape and manage it effectively.
- **Early Detection of Vulnerable and Outdated Components:** SCA is particularly effective at identifying components that are outdated or no longer maintained.
- **License Compliance:** Beyond security vulnerabilities, SCA also helps with the management of legal risks by identifying the licenses of third-party components. This ensures that organizations comply with the licensing terms of the open source software they use.
- **Early Detection in the Development Cycle:** SCA can be integrated early in the development process, allowing developers to identify and address vulnerabilities in third-party components before they become part of the production environment. This early detection helps reduce the overall risk and cost associated with fixing vulnerabilities later in the life cycle.
- **Continuous Monitoring:** SCA tools can continuously monitor third-party components, and alert organizations to new vulnerabilities as they are discovered. This ensures that applications remain secure over time, even as new threats emerge.

While each methodology has its strengths, their true power lies in their combined use. This comprehensive approach is essential for addressing the complex security challenges revealed in our analysis.

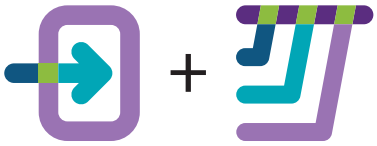
Synergies Between Testing Methodologies

SAST + DAST



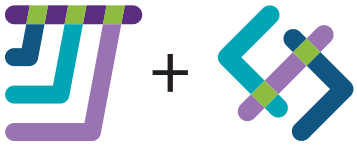
For Cross-Site Scripting, SAST identifies Improper Output Encoding, while DAST catches runtime-specific instances. For SQL Injection (example: CWE-89), SAST detects obvious vulnerabilities, while DAST identifies runtime-dependent issues.

DAST + SCA



SCA identifies potential vulnerabilities in components, while DAST verifies their exploitability in the running application. DAST provides continuous monitoring in production, complementing SCA's static analysis. DAST can also verify the effectiveness of patches identified by SCA in the actual runtime environment.

SCA + SAST



SCA detects known vulnerabilities in open source and third-party libraries, while SAST evaluates the source code, bytecode, or binary code of an application for vulnerabilities without the need for execution.

Conclusion

The findings from our 2023-24 DAST analysis of 1,300 applications, systems, and servers across a wide spectrum of industry sectors reveal a pressing need for organizations to enhance their application security strategies.

A total of 96,917 vulnerabilities were identified, with critical categories such as Sensitive Data Exposure and Injection vulnerabilities posing significant risks across various industries. The prevalence of these vulnerabilities—30,726 instances of Sensitive Data Exposures and 4,814 instances of Injection vulnerabilities alone—underscores the urgent need for organizations to prioritize their security efforts.

DAST is not a "silver bullet" that will solve all web application security issues; however, DAST plays a crucial role in a comprehensive security program that also includes SAST and SCA as fundamental pillars.

While SAST is effective at identifying coding flaws early in the development process, and SCA provides valuable insights into third-party components, DAST excels at detecting runtime vulnerabilities and verifying the exploitability of identified issues. The interplay between these methodologies allows organizations to achieve a more holistic view of their security posture, addressing vulnerabilities that may otherwise go unnoticed.

When it comes to preproduction and development environments, DAST solutions can be integrated into CI/CD pipelines to identify vulnerabilities early, accelerating remediation and reducing the cost of fixes. This approach is particularly valuable for detecting issues that may manifest only in a running application, such as certain types of Injection vulnerabilities or unexpected interactions between services and components.

DAST used in production offers additional benefits, especially for organizations dealing with complex, dynamic applications or those in highly regulated industries. Production DAST can provide continuous monitoring, detecting vulnerabilities that arise due to configuration changes, newly discovered exploits, or changes in the application's runtime environment. It's particularly valuable for identifying issues with third-party components that may become vulnerable over time, and for verifying the effectiveness of patches in the actual production environment.

As organizations face an increasingly complex threat landscape, it is imperative that they adopt a multifaceted approach to application security. This includes integrating DAST, SAST, and SCA into their development and deployment processes. By doing so, organizations can identify and remediate vulnerabilities more effectively, as well as ensure compliance with regulatory requirements while maintaining customer trust.

This report's findings make it crystal clear that organizations in every industry need to take proactive steps to enhance their security posture. By implementing these measures, organizations can significantly reduce their risk exposure and better protect their sensitive data and critical systems from emerging threats.

About Black Duck

Black Duck® offers the most comprehensive, powerful, and trusted portfolio of application security solutions in the industry. We have an unmatched track record of helping organizations around the world secure their software quickly, integrate security efficiently in their development environments, and safely innovate with new technologies. As the recognized leaders, experts, and innovators in software security, Black Duck has everything you need to build trust in your software. Learn more at www.blackduck.com.