What is Application Security?

Meta: Learn what application security is, and how it is used to protect application software from attackers looking to steal sensitive company and customer data.

Application security (AppSec) is an arm of cybersecurity that focuses on protecting software applications using a series of advanced tools and best practices. This includes identifying, fixing, and preventing app-based vulnerabilities before attackers can exploit them.

AppSec is a continuous process that starts early in development and extends well beyond deployment, protecting apps from unauthorized access, data breaches, and code manipulation at every stage of their lifecycle—not just as an afterthought.

Why is Application Security Important?

Whether they're facing customers or supporting staff behind the scenes, applications are the lifeblood of most businesses. Unfortunately, this also makes them prime targets for attackers looking to steal sensitive company and customer data.

The complexity of hybrid networks, open-source software, and distributed supply chains only adds to this risk—creating more opportunities for attackers to exploit app-based vulnerabilities. AppSec helps organizations protect against these threats, catch vulnerabilities early, and stay compliant with strict industry regulations.

What Types of Applications Need Protecting?

- **Web Applications:** Web apps are exposed to the internet, making them prime targets for attacks like data breaches and injection threats. Strong security measures like encryption and regular testing are crucial for keeping them safe.
- **Mobile Applications:** Mobile apps, which we rely upon for services including work, banking, and shopping, hold sensitive corporate and consumer data. Whether native (built for platforms like iOS and Android) or web-based, attackers can target mobile apps to gain key credentials, financial information, and intellectual property.
- **Cloud-Based Applications:** When migrating to the cloud, it can be easy to overlook how important it is to secure cloud-native apps and their supporting infrastructure. Cloud environments are dynamic and can be tricky to manage, which opens the door to risks like misconfigurations, unauthorized access, and data leaks.
- **Microservices and APIs:** Modern apps use microservices and APIs to connect different parts of the system. Securing these APIs is key to protecting data and ensuring proper access control across your network.
- **Legacy Applications:** Legacy apps are often overlooked as vulnerabilities. Since they're typically not updated with the latest security patches or modern authorization tools, they're an easy target for attackers looking to breach your network. These "time bomb" apps need regular checks and should be decommissioned if they're no longer in use.

Common Application Security Risks

The Open Web Application Security Project (OWASP) Top Ten is a solid starting point for understanding the most common application security risks. These best practices, recognized worldwide, help ensure you're tackling the most important vulnerabilities that could affect your apps. The OWASP list warns of common application security risks such as:

Security Risk	Examples	Impact
Access and Authentication Flaws	Broken Authentication (OWASP A01:2021)	Attackers exploit weak or missing access controls to bypass authentication, exposing sensitive systems and stealing data.
Weak Encryption and Data Integrity	Cryptographic Failures (OWASP A02:2021)	Poor encryption practices leave sensitive data vulnerable to theft and manipulation, compromising trust and system operations.
Exploitable Code and Design	Injection (OWASP A03:2021) Insecure Design (OWASP A04:2021)	Attackers can use injection flaws or weak design to manipulate data, take over applications, and disrupt business functions.
Misconfigurations	Security Misconfiguration (OWASP A05:2021) Vulnerable and Outdated Components (OWASP A06:2021)	Misconfigurations in cloud, on-prem, and containerized environments—as well as unpatched systems—can expose known vulnerabilities.
Monitoring and Detection Gaps	Insufficient Logging and Monitoring (OWASP A09:2021) Server-Side Request Forgery (SSRF) (OWASP A10:2021)	Attackers exploit weak monitoring to stay undetected, using flaws like SSRF to infiltrate internal systems or exfiltrate data.

What is Application Security Testing?

To protect your apps, you need to understand their vulnerabilities—and testing is the key. Application security testing involves identifying weaknesses that attackers could exploit. It combines manual methods and automated tools across the app's lifecycle to catch any vulnerabilities in code and during runtime. Here's a breakdown of the main types of application security testing:

- **Static Application Security Testing (SAST):** SAST inspects source code, bytecode, or binary code for vulnerabilities. It's often used early in the development lifecycle to detect security issues before the application is deployed.
- **Dynamic Application Security Testing (DAST):** DAST analyzes applications while they're running to spot vulnerabilities like injection flaws and insecure configurations. It's essential for catching runtime issues that SAST might miss.
- **Interactive Application Security Testing (IAST):** IAST combines the strengths of SAST and DAST by analyzing code in real-time while it's running in a test environment. This dual-layer insight helps uncover both static and dynamic vulnerabilities.
- **Runtime Application Self-Protection (RASP):** RASP is a more modern type of testing that integrates directly into the application to detect and block threats in real time. It offers a dynamic layer of defense during the application's execution phase.
- **Software Composition Analysis (SCA):** Scans your application's third-party components, such as libraries and open-source software, to identify vulnerabilities and licensing issues.
- **Fuzzing:** A technique that inputs random data into apps to find vulnerabilities like crashes or memory leaks. It's typically automated for continuous testing.

Common Application Security Tools

- **Code Analysis Tools:** These tools check the app's code early in development to spot vulnerabilities that could turn into security risks. They're typically integrated into continuous integration (CI) workflows to keep security checks running throughout the app's lifecycle.
- **Security Testing Tools:** Perform DAST and SAST to identify issues in both running applications and static code.
- **Container Security Tools:** Protect containerized environments like Docker and Kubernetes. These tools help ensure that containers are properly configured, free from vulnerabilities, and secure by enforcing policies for access control.
- **Application Shielding Tools:** These tools are designed to secure apps during the execution phase, preventing attacks that might otherwise exploit runtime vulnerabilities by obfuscating code, encrypting data, and continuously monitoring for security issues.

Essential Application Security Best Practices

- **Integrate Security Early:** "Shift left" and prioritize application security at every stage of development with regular code reviews, vulnerability scans, and pen tests.
- **Strengthen Access and Data Protection:** Use strong authentication (like MFA), authorization (such as RBAC), and encryption protocols to protect sensitive data, as well as a holistic identity and access management (IAM) solution to manage digital identities.
- **Build a Solid Defense:** Use web application firewalls and patch apps, APIs, and third-party components regularly. Educate your team on best practices and use trusted references like the OWASP Top Ten for more best practices help.
- **Continuously Monitor for Threats:** Monitor apps and APIs with real-time logging and tools such as identity and access management. Build a threat model to uncover vulnerabilities before they escalate into security risks.

Read more on our top 12 application security best practices here.

Top Trends in Application Security

Rise of DevSecOps

DevSecOps, which ensures security is woven into every stage of the development pipeline, is becoming a standard practice. Automated testing tools are also now a staple within continuous integration and delivery (CI/CD) workflows to help detect vulnerabilities in real time.

API Security

APIs are key to modern apps, especially in microservices, as they enable communication between different components. However, this makes them attractive targets for attackers. With more breaches involving APIs, companies are increasingly focusing their AppSec efforts here. The API security market is expected to grow at a compound annual growth rate (CAGR) of 32.5%, reaching around $3 billion by 2028.

AI-Powered Security

34% of companies are already using artificial intelligence (AI) tools for application security. They can quickly identify patterns, detect threats, and predict vulnerabilities, making AppSec faster and more proactive.

Supply Chain Security

Attacks like Log4j have made it clear why securing third-party components is so important. Organizations are now using dependency management tools, Software Bill of Materials (SBOM), and continuous monitoring to secure their software supply chains.

Application Security Posture Management (ASPM)

ASPM tools offer a holistic solution for organizations to map and protect their application assets. By 2026, Gartner predicts that 40% of organizations developing their own apps will be using ASPM tools.

Master Your Application Security with AlgoSec

Take charge of your application security with AlgoSec. Gain full visibility across both on-prem and cloud environments, simplify your security policies, and fend off sophisticated threats—all from a single platform. AlgoSec ensures your applications remain secure, agile, and audit-ready. Book a quick demo today.

FAQ

What is the OWASP Top 10, and why is it important?

The OWASP Top 10 outlines the most critical web application security risks. It's a must-have reference for addressing the latest application vulnerabilities, and it's updated regularly.

What are the challenges of application security right now?

AppSec teams face challenges like securing legacy applications with outdated tech, managing complex third-party components, and integrating security into fast-moving DevSecOps workflows. There's also a shortage of skilled AppSec professionals, and many organizations lack the right tools for managing (and centralizing) their AppSec operations.

What is a Software Bill of Materials?

A Software Bill of Materials (SBOM) lists all the components of an application, including open-source libraries. It helps organizations track dependencies and identify vulnerabilities in third-party code.

How often should applications undergo security testing?

Testing should be continuous throughout the Software Development Lifecycle (SDLC). Automated tools should run regularly, with periodic manual reviews for high-risk applications.

What's the difference between application and API security?

Application security focuses on the overall application, including frontend and backend components. API security is a subset of application security, and it specifically targets APIs, ensuring they're protected against vulnerabilities like improper authentication, data exposure, and excessive privileges.