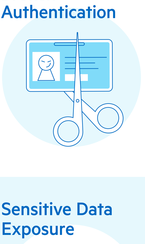
Progress Sitefinity™

Protecting Against the 10 Most Critical Web Security Risks

With the number of cyberattacks and security breaches skyrocketing, web application security is vital for today's businesses. Here are the top 10 most critical web security risks, as identified by the Open Web Application Security Project (OWASP), and how Progress® Sitefinity™ CMS protects you from them.

1/ Injection



▲ Security Risk

Injection attacks (SQL injection, for example) are when attackers send untrusted input that tricks the interpreter into executing unintended inputs or accessing data without proper authorization.

● How Progress Sitefinity CMS Protects

Applications should provide an API that avoids the use of the interpreter or exposes an entirely parameterized interface. Sitefinity does both; it calls the underlying provider that manages data access through Data Access ORM. Additionally, it internally provides an entirely parameterized interface, ensuring no single method can be executed without privileges.

2/ Broken Authentication

▲ Security Risk

If authentication and session management functions are implemented incorrectly, attackers can easily exploit implementation flaws to gain unauthorized access to user data.

● How Progress Sitefinity CMS Protects

Sitefinity uses three authentication models that comply with security standards such as FIPS to prevent broken authentication. The default authentication is based on OAuth 2.0 and OpenID Connect protocols, and uses IdentityServer3. Passwords are encrypted when stored and settings can be adjusted for stricter security policies.

3/ Sensitive Data Exposure

▲ Security Risk

Many web applications and APIs do not properly protect sensitive data. This enables attackers to compromise weakly protected data to commit credit card fraud, identity theft or other cybercrimes.

● How Progress Sitefinity CMS Protects

Sitefinity stores the minimal set of sensitive data that is required to operate. At rest, all internal and custom sensitive data is protected using a cryptographic API with strong standard algorithms. In transit, an encrypted transport layer security (TLS) protocol should be enforced by configuring a strict transport security header (HSTS) and public key pins header (PKP).

4/ XML External Entities (XXE)

▲ Security Risk

Older or poorly configured XML processors evaluate external entity references within XML documents. Attackers can use these external entities to exploit vulnerabilities.

● How Progress Sitefinity CMS Protects

With Sitefinity, all XML processing relies on Microsoft .NET Framework parsers and it regularly updates all libraries and frameworks to the latest versions of .NET Framework (4.5.2+). Any XML files the system processes come from trusted sources, with the exception of SVG images uploaded by users—but XXEs are prevented by removing the XmlResolver to disable the DTD processing.

5/ Broken Access Control

▲ Security Risk

Access control governs which actions users are allowed to take once logged in. However, if poorly enforced, flaws in access control can be exploited to access unauthorized functionality or data.

● How Progress Sitefinity CMS Protects

Sitefinity checks for authentication permissions for each CRUD operation. Bypassing security checks is impossible externally through any mechanism—URL, service call or API.

6/ Security Misconfiguration

▲ Security Risk

This is when web applications or servers are not configured correctly, resulting in access to admin interfaces, error messages that expose sensitive information or other vulnerabilities.

● How Progress Sitefinity CMS Protects

While a big part of the burden lies on system admins, Sitefinity provides an easy infrastructure for deploying and applying updates to a secured environment. It runs on the latest .NET Framework security features and is put through independent audits to meet top security standards.

7/ Cross-Site Scripting (XSS)

▲ Security Risk

Cross-site scripting (XSS) is when attackers inject malicious scripts into a legitimate web app or website, enabling them to target users through websites or apps they routinely visit.

● How Progress Sitefinity CMS Protects

Sitefinity offers a number of protection measures, including HTML sanitizers, HTTP headers, CSP headers and X-XSS-Protection headers.

8/ Insecure Deserialization

▲ Security Risk

Apps and APIs are vulnerable if they deserialize hostile or tampered objects supplied by an attacker. Insecure deserialization often leads to remote code execution and can be used to perform attacks.

● How Progress Sitefinity CMS Protects

Sitefinity uses Json.NET, JavascriptSerializer and DataContractJsonSerializer. By default, .NET serializers are protected unless configured with non-default settings or the user controls the deserialized type, something not used in Sitefinity. Sitefinity uses the serializers securely.

9/ Using Components with Known Vulnerabilities

▲ Security Risk

Websites and apps are often built using components such as libraries and frameworks to save time. Using components with known vulnerabilities may open avenues for attackers.

● How Progress Sitefinity CMS Protects

Sitefinity is built on top of the .NET Framework and uses many external libraries and services. They are strictly checked for updates and security patches (which are quickly applied).

10/ Insufficient Logging and Monitoring

▲ Security Risk

Without sufficient logging and monitoring measures, it may take enterprises even longer to detect attacks and breaches, which could lead to an even greater financial impact.

● How Progress Sitefinity CMS Protects

Sitefinity provides a logging mechanism that is extensible and can be used to persist information in different auditing systems