Progress®

SITEFINITY CMS SECURITY AND BEST PRACTICES

WHITEPAPER

Table of Contents

Abstract

This document provides information about the security aspects of Progress Sitefinity CMS.

Purpose

- To help in the decision-making process.
- To provide an overview for software engineers and site administrators of the available security configurations in the product.

Scope

Progress Sitefinity CMS on premise – security aspect of the:

- Progress Sitefinity organization
- Processes
- Product features

Disclaimer

Progress Sitefinity does not protect from poor implementation or misconfiguration of the product. It cannot protect from infrastructure issues. Although the CMS may protect your site from some attacks, it cannot protect from each vulnerability type (e.g. social engineering attacks). We strongly recommend providing proper and regular security trainings for each person that builds, maintains and uses the site.

Executive Summary

Many large organizations rely on Progress® Sitefinity™ CMS for delivering their web presence—from government agencies, financial institutions and Fortune 500 companies to various businesses all over the world.

Security is not open to compromise and this is reflected in the CMS evaluation process. We often receive a great spectrum of questions about security in Progress Sitefinity CMS.

This whitepaper addresses those questions by listing the most common threats that organizations face today, explaining what they are, what Progress Sitefinity CMS is doing to prevent them and what extra steps are available to make your environment more secure.

The document explains the most important security aspects of Progress Sitefinity CMS:

- The organization's commitment to security and how this affects the secure software development lifecycle, team structure and trainings.
- Security features of the product.
- OWASP Top 10 – 2017 most common threats that any web application faces and how Progress Sitefinity CMS handles these kinds of threats to ensure the security of your system.
- Best practices to secure your website on a production environment.

Software Assurance Maturity Model (SAMM)

Progress Sitefinity CMS uses OWASP SAMM framework to implement a strategy for software security. This model helps organizations identify the areas in a web content management system which have higher risk, and to focus resources to deliver a secure product that each customer can rely on.

Team

There is a cross function team at Progress - Sitefinity Security Group. The team has engineers from different teams and positions, including the following:

- Software architects
- Team managers
- Software engineers
- QA engineers
- Support engineers

The team conducts regular meetings and all security related issues are discussed, planned and executed to maintain the high security standard of the product.

Secure Product Development

Process

Each release of Progress Sitefinity CMS goes through several phases from planning, design and implementation to testing and maintaining the released version. Security is an important part of the entire system development lifecycle of the CMS. There are security review procedures in the design and implementation phases that include highly qualified security experts who check for security vulnerabilities. All external libraries integrated in the product are regularly checked. Testing of a new feature includes specific security scans (static and dynamic) to prevent

vulnerabilities in the product. During the maintenance phase of the product, security related reports are considered and handled with highest priority.

Standards, Certificates and Compliance

Progress Sitefinity CMS platform is certified by an independent service auditor to comply with the Service Organization Control Standards (SOC 2) developed by the Association of International Certified Professional Accountants (AICPA).

Compliance with SOC 2 is a testament that Progress has established a comprehensive set of internal procedures and controls to ensure the security, processing integrity, confidentiality and availability of software development infrastructure. This increases the confidence that organizations have when choosing to rely on Progress services and products for their business.

The Progress SOC 2 certification report covers the following areas of internal controls:

- **Security** – helps protect against unauthorized access, use, or modification
- **Availability** – ensures service is available for operation and use as committed or agreed upon
- **Confidentiality** – ensures confidential information is well protected

GDPR – Progress also operates a GDPR Office that conducts a range of activities that address GDPR regulatory requirements. Administrative, technical and operational capabilities are in place that can assist customers with GDPR questions, needs and requirements.

Trainings

Each year, engineers on the Progress Sitefinity CMS team participate in at least two different in at least two different platforms for security trainings - Veracode and Wombat.
Some of the security experts on the team take additional SANS training. There are also internal trainings and knowledge sharing sessions and events to keep knowledge in the field of software security up-to-date.

Progress Sitefinity CMS Release Cycle

There are several types of releases of Progress Sitefinity CMS:

- Major version - 2-4 times a year (e.g. versions 10.0, 10.1, 10.2). They contain new features and improvements in old features. Contains fixes from previous releases. It is extensively tested in all aspects with diverse types of tests.

- Service packs - cumulative update of critical issue fixes only. Security matters with CVSS rating of High and Critical are included.

- Internal build - each week. Contains the Service Pack fixes (if any) and less critical fixes. They contain latest fixes including but not limited to security no matter of the CVSS score (even the one with lower ratings).

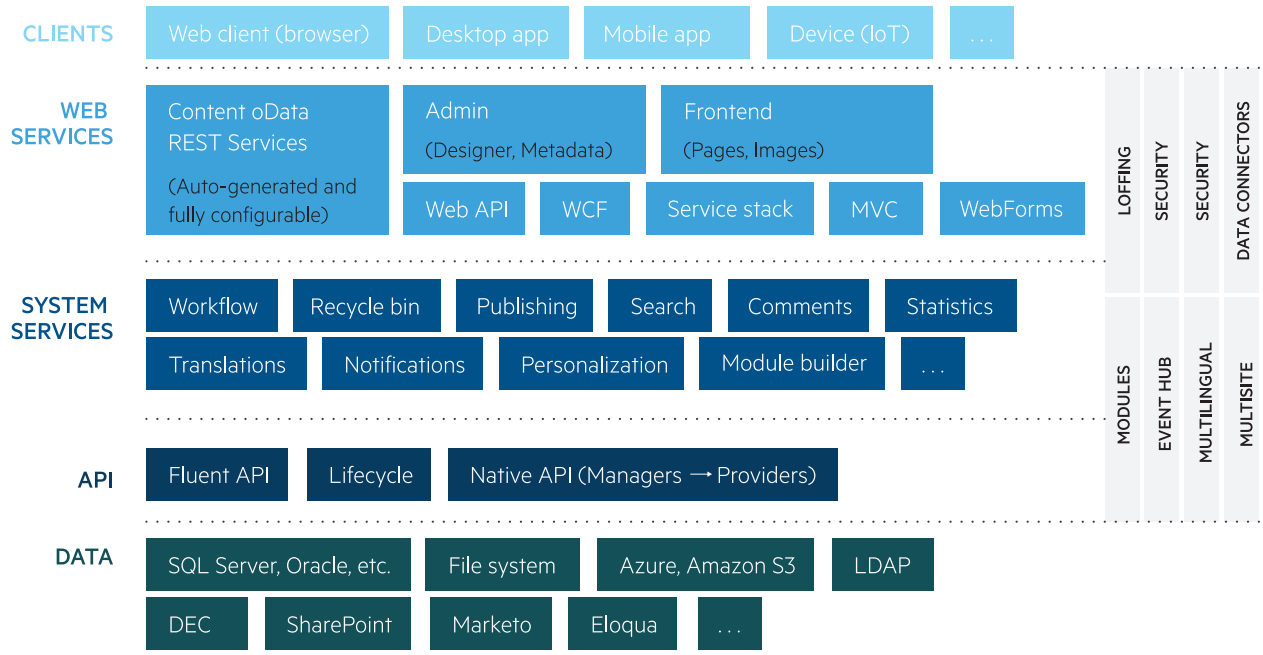
Critical and High security issues are fixed and backported to supported older versions of the product. This leads to new Service Pack versions.

Product Architecture

Sitefinity CMS has many protection layers to ensure data integrity, confidentiality and availability. Appropriate checks are executed on different layers to prevent security attacks on the system.

In the following diagram, you will find the different tiers and their respective modules.

Product Architecture and Security



Data Encryption

Sensitive data is encrypted with appropriate algorithms depending on the risk profile – for example, data at rest or data in transit. Progress Sitefinity CMS uses FIPS compliant algorithms for hashing and encryption.

Sitefinity is FIPS compliant, except for the following areas:

- Export and import functionality.
- Ecommerce's World Pay Provider integration.
- POP3 client, in case the authentication mode is **AuthenticationMethod.APOP** or **AuthenticationMethod.TRYBOTH.AuthenticationMethod.USERPASS** is FIPS compliant.
- In the LibrariesConfig, only the default **ImageUrlSignatureHashAlgorithm.SHA1** algorithm is compliant for **ImageUrlSignatureHashAlgorithm**.

Authentication Mechanisms

Progress Sitefinity CMS offers three major types of authentication out-of-the-box:

- Default authentication is based on OAuth 2.0 and OpenID Connect protocols. It gives flexibility and out-of-the-box integration with many third-party providers, such as ADFS, Windows, LDAP, Facebook, Google, Twitter, Microsoft, etc. In addition, the protocols are designed with security in first place. The implementation is based on certified libraries, such as IdentityServer3 – Certified OpenID Connect implementation.
- WRAP/SWT implementation
- Forms authentication

Permissions, Users, Roles

Progress Sitefinity CMS comes with Role providers and Membership providers that help manage users in the system and assign them different roles. This helps to configure proper permissions for managing different types of content. The product also has a flexible system for defining granular permissions per item. The constrained or allowed principals are Roles and individual Users. Permissions are applied on different types or items for the variety of operations, such as View, Create, Delete, Modify, etc. Depending on the type of object, permissions are verified on the level of different system layers and on different modules level.

API level checks: Permissions for content items, such as news, events, dynamic content items etc., are verified on a lowest API level - providers.

Reading items from the database can have filtering by view permissions to prevent unauthorized read access.

Sitemap filters: Pages with restricted access have specific checks to prevent unauthorized access. Sitemap filters are responsible for this protection. For more information about configuring page permissions, see Sitefinity CMS official documentation: Grant permissions for pages.

Site Shield

You can use the Site shield feature to protect a website that is under development from unauthorized access. You use it to allow users without backend permissions to view the site while it is developed. For example, when stakeholders want to evaluate the progress of a website, but they do not have backend permissions. For more information, see Sitefinity CMS official documentation: Site shield: View unpublished websites.

Sanitizing

HTML sanitization: Progress Sitefinity CMS has an out-of-the-box HTML Sanitizer that prevents dangerous HTML and possible XSS attacks. For more information, see Sitefinity CMS official documentation: HTML sanitization.

SVG sanitization: The product has a built-in file processor that sanitizes the SVG images on upload. It uses a whitelist to prevent dangerous user input.

Audit Trail

Enterprise systems that have to conform to the security standards must have an Audit Trail that preserves the log of the user actions. Progress Sitefinity CMS provides a module that persist this type of information. For more information, see Sitefinity CMS official documentation: Audit Trail module.

Securing Configuration Files

In Progress Sitefinity CMS, config files may contain sensitive data that should not be visible by default – for example, credentials to external systems, connection strings, etc. For this purpose, there is an option to encrypt values in the configuration files. Moreover, they could be stored in external key management service, such as Azure Key Vault or AWS Key Management Service.

Web Security Module

Progress Sitefinity CMS has an additional layer of protection to your site – the Web security module. It prevents from different types of web attacks.

- **Security HTTP headers** – As of Sitefinity CMS 11.0 the system can send HTTP headers to configure web clients (browsers) and turn on their build-in security features. There are various types of attacks that can be prevented – XSS, clickjacking, code injection, MTM – stealing or modifying data in transit.

- **Open redirect protection** – As of Sitefinity CMS 11.1 the system comes with built-in Open Redirect protection that notifies the user when she is leaving the site and is being redirected to an external domain. This can prevent phishing attacks and stealing of end-user data.

- **Cross-Site Request Forgery (CSRF) prevention** – As of Sitefinity CMS 12.1, the Web security module enables IT Administrators to configure a centralized

mechanism that helps secure the website cookies, thus preventing CSRF vulnerability. Website administrators can set a minimum-security policy for all website cookies by configuring the SameSite, HttpOnly, and Secure attributes.

For more information about different types of protections and how to configure your site see Web security module official documentation.

Top Ten Most Critical Security Risks

The following list discusses the OWASP Top 10 Application Security Risks - 2017 and the actions that Progress Sitefinity CMS made in response to these risks.

In general, any web application can expose many ways for attackers to get unauthorized access or compromise its integrity. Some of those threats have become widely popular and discussed – the top ten security risks have been compiled in an extensive list provided by the Open Web Application Security Project (OWASP). Following is a summary of those threats and vulnerabilities in the context of Progress Sitefinity CMS security.

Injection

Security risk: There are several types of injections flaws – SQL, OS Command, LDAP. The attacker's hostile data can trick the interpreter to execute unintended commands or access data without proper authorization. For more information, see OWASP Top 10-2017 A1-Injection.

The most common one is SQL injection, because the product uses database to store most of its data. A SQL injection is often used to attack the security of a website by inserting SQL statements in a web form. The main purpose of a SQL injection is to get a badly designed website to perform operations on the database that were not intended by the designer of the system - for example to dump information stored in the database and expose it to an attacker. An application is vulnerable when data provided by user input can be executed.

Progress Sitefinity CMS response

- To prevent SQL injection, the applications should provide an API that either avoids the use of the interpreter or exposes an entirely parameterized interface. Progress Sitefinity CMS is a combination of these two. It does not execute a single native SQL statement. It calls the underlying provider that manages data access through Data Access ORM – an enterprise level object relational mapping tool. In addition, Data Access internally provides an entirely parameterized interface.

- Furthermore, the security API is on the provider level, ensuring that not a single method can be executed without privileges.

Broken Authentication

Security risk: Application functions related to authentication and session management can be implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.

Progress Sitefinity CMS response

- Sitefinity CMS provides an extensive set of measures to prevent such attacks. The application provides three authentication models that comply with high security standards. The default authentication mode is based on OAuth 2.0 and OpenID Connect protocols and it uses IdentityServer3 - a product that has a certificate for OpenID Connect implementation.

- Passwords are stored in an encrypted format. The default settings in Progress Sitefinity CMS require a minimum of 7 characters per password and secure timeout settings. These settings can be overridden to enforce a stricter security and password policy.

Sensitive Data Exposure

Security risk: Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

Progress Sitefinity CMS response

- Progress Sitefinity CMS stores the minimal set of sensitive data that is required for the functionality of the product.

- To protect data at rest there is a cryptographic API that uses strong standard algorithms. It is used by all internal sensitive data. For example, by default, the CMS stores password hashes. All custom sensitive data, depending on the site implementation, can use the same API to persist it securely.

- To protect data in transit, we strongly recommend using an encrypted transport layer security - TLS protocol. You can enforce it by configuring strict transport security header (HSTS) and public key pins header (PKP) in the Web security module of Progress Sitefinity CMS.

XML External Entities (XXE)

Security risk: Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution and denial of service attacks.

Progress Sitefinity CMS response: There are several layers of protection and each of them can prevent XXE attack:

- Sitefinity CMS is built on top of the Microsoft .NET Framework. There are several places with XML processing in the system. They all rely on .NET Framework parsers. The product regularly updates all libraries and frameworks it uses. According to OWASP XML External Entity (XXE) Prevention Cheat Sheet, the latest versions of .NET Framework (4.5.2 and above) are safe by default. However, .NET Framework version older than 4.5.2 also have XSS protection.

- The XML files, which the system processes, come from trusted sources – for example, generated by the system itself. There is one exception – SVG images, which can be uploaded by end-users, but they are explicitly protected by removing the XmlResolver to disable the DTD processing and no XML External Entities are allowed.

Broken Access Control

Security risk: Restrictions on the actions allowed to authenticated users can often be poorly enforced. Attackers can abuse these flaws to access unauthorized functionality or data, such as accessing other users' accounts, viewing sensitive files, modifying other users' data, changing access rights, etc.

Progress Sitefinity CMS response: Progress Sitefinity CMS checks for authentication permissions for each create, retrieve, update and delete operation. Bypassing security checks is impossible externally through any mechanism – URL, service call, or API.

Security Misconfiguration

Security risk: Security misconfiguration is the most commonly seen issue. This is usually a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers and verbose error messages that contain sensitive information. In addition to securely configuring all operating systems, frameworks, libraries and applications, you must also patch and upgrade them in a timely fashion.

Progress Sitefinity CMS response: While some aspects of security configurations are in the scope of system administrators and not the application itself, Progress Sitefinity CMS provides an easy infrastructure for deploying and applying updates to a secured environment. The system also runs on the latest security features provided by the .NET Framework. Furthermore, to ensure top-line security standards and best practices, the application is run through independent audits - Veracode static and dynamic security scans.

Cross-Site Scripting (XSS)

Security risk: XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data, using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the target's browser that can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Progress Sitefinity CMS response: There are several layers of protection build into the product:

- Sanitizers. Progress Sitefinity CMS has HTML sanitizers that prevent dangerous content to be rendered in the browser. It uses a whitelist with the allowed HTML tags and attributes. This is the most strict and recommended approach for protection against XSS. For more information, see Sitefinity CMS official documentation: HTML sanitization.

- Encoding. All out-of-the-box widgets use the appropriate encoding and sanitization. Depending on the context - HTML, JavaScript, URL, etc., appropriate encoding is applied to prevent rendering potentially dangerous content.

- HTTP security headers. The system sends HTTP headers to configure web clients (browsers) and turn on their built-in security features.

- Content-Security-Policy header. This is one of the most powerful weapons for protection against XSS. To mitigate the risk, a web application can declare that it only expects to load script from specific, trusted sources. This declaration allows the client to detect and block malicious scripts injected in the application by an

attacker. For more information about configuring it, see the official Sitefinity CMS documentation. For more information about the Content-Security-Policy header, see http://content-security-policy.com.

- X-XSS-Protection header. Prevents reflected cross-site scripting attacks. The default value (1; mode=block) prevents rendering the page, if an attack is detected. For more information, see https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection.

Insecure Deserialization

Security risk: Applications and APIs will be vulnerable if they deserialize hostile or tampered objects supplied by an attacker. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks and privilege escalation attacks.

Progress Sitefinity CMS response: Progress Sitefinity CMS uses Json.NET, JavascriptSerializer and DataContractJsonSerializer. By default, .NET serializers are protected, unless they are configured with non-default settings or the user controls the deserialized type. There are no such use cases in Progress Sitefinity CMS. The CMS uses the serializers securely - with default configuration or by specifying the type that is deserialized.

Using Components with Known Vulnerabilities

Security risk: Components, such as libraries, frameworks and other software modules run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

Progress Sitefinity CMS response: The CMS product is built on top of the .NET Framework and uses many external libraries and services. They are strictly checked for updates and especially for security patches. If such are available, they are applied in the product and a new version of Progress Sitefinity CMS is released.

Insufficient Logging and Monitoring

Security risk: Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems and tamper, extract, or destroy data. Most breach studies reveal that time to detect a breach is over 200 days. Usually, breaches are detected by external

parties, rather than internal processes or monitoring.

Progress Sitefinity CMS response: Progress Sitefinity CMS provides a logging mechanism that is extensible and can be used to persist information in different auditing systems. By default, the product provides error logging and audit trail functionalities that persist their data to the file system or can be sent to Elastic for further analysis and monitoring.

Best Practices

The whitepaper discusses the top ten most common attacks that target web applications and the countermeasures that Progress Sitefinity CMS takes to prevent them. There are a lot of other layers, devices and systems where you would have to enforce security. We recommend being aligned with latest security best practices. Most of those countermeasures deal with lower level software and protocols. The list of possible software and network vulnerabilities is long and the first and most important task that any attacker would have is to learn as much as possible about your system, specifics and topology.

Following is a summary of the most common attacks and respective best practices.

Securing Your Network

Network security has various aspects – computer systems, access control, preventing unauthorized information gathering, firewalls, physical security, detection and response to unwanted incursions. The most common attacks that physical networks face are information gathering, sniffing, spoofing and session hijacking. Multiple vulnerabilities enable these kind of attacks, including exposed ports, services, protocols, poorly encrypted data, weak physical security, and the inherently insecure nature of the TCP/IP protocol.

Below is a checklist of best practices that you can follow to build a more solid defense:

1. Enforce strong physical security of your network – this is in broad topic and measures could vary from locking machines that are not in use, to access cards or biometric access.

2. Do not give out custom errors, configuration information, and software versions.

3. Apply the latest patches and updates to your OS, routers, switches and firewalls.

4. Disable ports and services that are not used.

5. Use firewalls between your DMZ and the public network and between your internal LAN network and your DMZ that mask all internal services.

6. Encrypt credentials and application traffic over the network.

7. Apply ingress and egress filtering on perimeter routers to prevent from spoofing.

8. Apply inspection at the firewall. Distributed Denial of Service (DDoS) attacks have become a powerful weapon in any attacker's toolset. While many prevention methods exist, the best way to handle those attacks is at a firewall level.

9. Filter broadcast and ICMP requests.

10. Apply strong password policies.

11. Centralize logging on all allowed and denied activities and have auditing against unusual patterns in place.

Web Server Security

A secure IIS instance can provide a solid foundation to hosting your Progress Sitefinity CMS application. While there are many considerations, measures and resources on the topic of IIS security, this checklist once again aims to give you a summary of the best practices that help you prevent some of the most common attacks and vulnerabilities. The main threats that your server can face include profiling, unauthorized access, elevation of privileges, viruses and worms, etc.

1. Block all unnecessary ports, ICMP traffic and unnecessary protocols and services. This will prevent port scans and ping sweeps that may give out information or locate doors open for attacks.

2. Apply the latest system patches and updates frequently.

3. Use separate application pool identities for each instance of Progress Sitefinity CMS you are hosting.

4. Do not give administrative rights to the application pool identity. It must have access only to the web application files, rather than the entire server.

5. Reject URLs with ../ to prevent path traversal.

6. Run processes using least privileged accounts.

7. Remove unnecessary file shares.

8. Disable unused ISAPI filters.

9. Properly configure the UrlScan tool, if you are utilizing it.

10. Carefully analyze the default and the installed IIS services and disable those that are not needed by the system, as defined by our installation guide. Disable FTP, SMTP and NNTP, unless you require them.

11. Install the SQL Server (or any of the other supported databases) on a separate, dedicated, physically secured, patched and updated server. Do not install unnecessary tools and debug symbols on the production server.

Other Countermeasures

Other good practices should be considered when you deploy Progress Sitefinity CMS.

1. Give out as little information about your system as possible. The first step prior to deploying your system, is enabling friendly error pages. Attackers love as much information as they can get their hands on.

2. Monitor the Progress Sitefinity CMS logs for any unusual patterns and errors.

3. We always recommend upgrading to the newest version of the software including Progress Sitefinity CMS itself.

4. Look at our hosting recommendations and setup to review some of the best practices for infrastructure and deployment on highly secure and highly scalable environments.

5. Think about the password policy that you want to apply to both frontend and backend CMS users. Progress Sitefinity CMS enables a wide set of measures including minimal password length, validating against a regular expression and maximal password attempts, to make a brute force attack technically impossible. The default restrictions are not too limiting to ensure ease of use, but they can also be tailored to fit a very strict security policy. Progress Sitefinity CMS natively supports Windows Authentication, LDAP and ADFS integration. Therefore, you can avoid maintaining a separate user base and password policy for your CMS and enforce this at a system level in your organization.

6. Progress Sitefinity CMS is designed according to current best practices in security. However, as with any other systems, you must be careful with how you extend it. As a powerful framework, Progress Sitefinity CMS allows you to plug any custom functionality into the system. Make sure that you follow the patterns and practices that would make your application secure when developing to avoid

Injection, XSS attacks etc. Utilize the tools and libraries provided by Progress Sitefinity CMS, because they are designed in a secure way and ensure stable implementation throughout your custom functionality.

Conclusion

Security is a sensitive and important subject in today's digital world and an application like Progress Sitefinity CMS can provide a stable platform for your entire online presence. Progress Sitefinity CMS implements several layers that can prevent different types of attacks, which makes it a reliable partner in this never-ending war.

By following the best practices and guidelines for network security and applying measured project specific and system specific prevention mechanisms, Progress Sitefinity CMS enables you to have a powerful, scalable and secure solution to power your organization's online presence.

For more information about security, get in touch with us to discuss your specific security needs at sales@sitefinity.com.

If you encounter a security issue, contact Progress Sitefinity CMS support. For more information, see Sitefinity Support Escalations.

For more information, see Sitefinity Support Escalations.

Rev 2019/09 | RITM0020573

Progress®