A Security Guide for

Developers

kiuwan
An Idera, Inc. Company

INDEX

Overview.....1

What Is Application Security.....2

What Is ASP.NET.....2

Best Practices When Coding in ASP.NET.....3

Top Risks for Developing in ASP.NET.....6

Risks for Developing in ASP.NET Compared to Other Development Languages.....8

Cost of Not Securing ASP.NET Applications.....9

How Teams Can Work to Secure Their ASP.NET Applications 10

Benefits of Securing ASP.NET Applications.....10

Final Words.....11

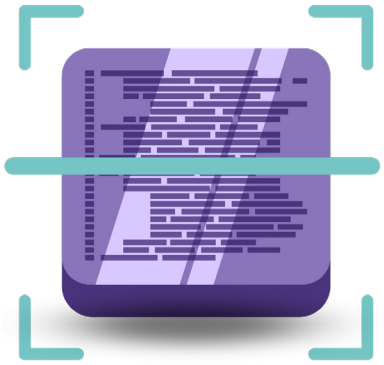
Overview

The growth of the internet and web-based applications has made software security an increasingly important consideration for all businesses. With the development of sophisticated tools and techniques, hackers can gain access to data and system resources through weakly secured applications. To prevent these types of attacks, companies must invest in application security best practices and ensure that their software is secure.

The choice of using ASP.NET is growing, especially in legacy programs, as it offers a reliable, secure and well-supported platform for developing web applications. However, with security being an essential aspect of software development, it is vital to understand how to secure the application, data, and users. In this guide, we provide a comprehensive overview of the various measures taken to ensure the data and software security of ASP.NET applications outlined here:

- Defining AppSec
- ASP.NET
- Best Practices
- Risks Developing in ASP.NET
- Costs of Not Securing ASP.NET Applications
- Benefits of Securing

What Is *Application Security*?



Application security is a set of processes, policies, and technologies designed to protect an application from unauthorized access. It focuses on protection from malicious acts aimed at the application itself and its associated data, including denial of service (DoS) attacks, data breaches, injection flaws, and other methods used to gain unauthorized access.

Application security involves multiple tools and techniques, such as Static Application Security Testing (SAST) and Software Composition Analysis (SCA). These allow organizations to identify security flaws and vulnerabilities in their applications before attackers can exploit them.

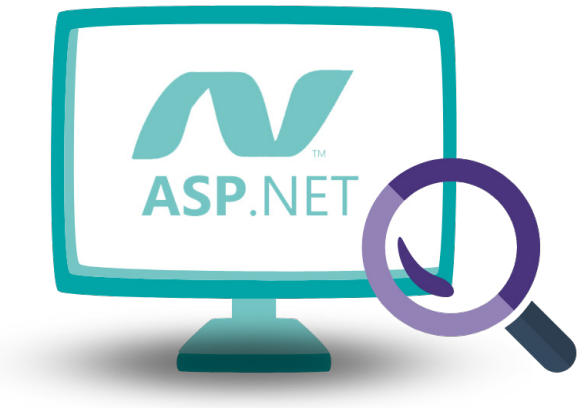
Why Is Application Security Important?

Application security is an essential part of any effective cybersecurity strategy. It helps protect sensitive data, secure applications from malicious attacks, and assure the integrity of systems and networks. This ensures that the information contained in an application is safe and secure.

What Is *ASP.NET*?

ASP.NET is a server-side programming framework used for web development and creating dynamic websites, services, and applications. ASP.NET is an open-source framework developed by Microsoft and used for building a wide range of web applications.

ASP.NET is an important tool to help manage the risks associated with building large-scale web applications and services, such as corporate websites and e-commerce platforms. It provides a secure environment for developers to create complex applications with features like authentication, error handling, caching, and more. ASP.NET is also a significant part of the .NET framework, which provides a complete application development platform and the ability to build web applications. ASP.NET is an efficient way for developers to create dynamic websites and applications, providing the tools necessary to deliver a more secure and reliable web experience.



What Are the Features of ASP.NET Applications?

ASP.NET is a versatile, feature-rich, and advanced framework for building dynamic and secure web applications. It provides developers with a range of features to create robust applications quickly. ASP.NET takes advantage of the ASP model and adds new features, such as caching, separation of code and HTML, object-oriented programming, and the use of an extensive class library. ASP.NET also allows developers to access data quickly and prevent potential security risks with built-in authentication and authorization mechanisms.

ASP.NET offers a wide range of deployment options, making it easy to build applications for mobile devices and the cloud. It provides powerful tools and technologies, enabling developers to quickly build secure, scalable applications with minimal effort.

ASP.NET delivers multiple development modes, including ASP.NET Web Forms and ASP.NET MVC (Model View Controller). ASP.NET Web Forms allows developers to quickly create web applications that are developed and deployed in fewer steps. ASP.NET MVC is more powerful and allows developers to create applications with high flexibility and scalability. ASP.NET also provides a wealth of options for creating web services, including ASP.NET Web API, WCF (Windows Communication Foundation), and ASP.NET SignalR.

Best Practices When Coding in ASP.NET

When coding in ASP.NET, there are several key best practices to keep in mind. Although ASP.NET is a powerful framework, it poses certain risks when not properly used. Therefore, it is essential to adhere to the following best practices:

Caching

Caching is a programming technique that stores frequently used data in memory, allowing faster retrieval of this data when needed. This reduces the time and resources required to process a request, eliminating potential risks associated with resource-intensive processes.

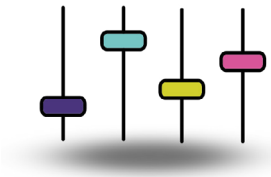
ASP.NET provides many powerful tools for caching data, allowing developers to easily implement caching in their applications. This significantly improves application performance. As a result, ASP.NET is an ideal framework for applications that need quick and efficient data. With the help of ASP.NET's caching functions, developers can build highly optimized applications that exploit the full potential of the ASP.NET platform.

Avoid Blocking Calls

When coding in ASP.NET, it is important to avoid blocking calls. The framework offers powerful functionality and improved scalability, but if used incorrectly, it can create risks. Blocking calls are synchronous requests and require waiting for a response before continuing with the program execution. This could lead to the ASP.NET application becoming unresponsive, causing system performance issues. Therefore, it is best practice to use asynchronous calls instead to maximize the performance and scalability of the ASP.NET application.

Validation Controls

The ASP.NET framework offers validation controls as a best practice for secure coding. These controls reduce the risks associated with security vulnerabilities, such as cross-site scripting and SQL injection. Validation controls can verify that user input is in the expected data type and format, ensuring that all external requests are safe and secure. Validation controls also help protect against malicious code or data that could otherwise compromise a **website's security**. By incorporating validation controls into ASP.NET code, developers can create a secure and reliable web application.

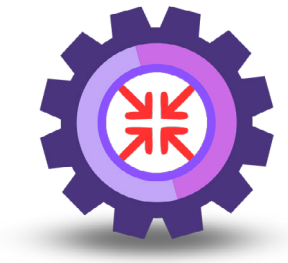


Minimize Exceptions

When coding in ASP.NET, it is important to minimize the use of exceptions. Exceptions can cause the application to fail, resulting in unexpected results and potential system crashes. The use of exceptions should be rare and only used to handle errors that cannot be easily prevented or anticipated. This will reduce the risk of security vulnerabilities exposed due to unexpected situations.

Compress Responses

When programming in ASP.NET, it is a best practice to compress responses. This reduces the amount of data sent over the network between the server and client, thereby improving the performance of ASP.NET applications. Compressing responses can also help to reduce bandwidth usage and enhance security by minimizing the risks associated with sending sensitive data over the internet.



ASP.NET supports various compression techniques, including Gzip, Deflate, and Brotli. Compressing responses should be part of any ASP.NET programmer's arsenal of best practices when creating ASP.NET applications.

Avoid Hard-Coding Sensitive Data

When coding in ASP.NET, it is important to avoid hard coding sensitive data into the codebase as it can increase security risks. Sensitive data includes passwords, encryption keys, database connection information, and other confidential information. Hard coding this type of sensitive data into the ASP.NET codebase can make it more vulnerable to hacker attacks, as this data is easily readable. It also increases the risk of theft or misuse of confidential information, which can lead to serious consequences. It is, therefore, best practice to use secure methods, such as encryption or hashing, to protect sensitive data stored in the ASP.NET application.

Naming Convention

A naming convention is an important best practice when coding in ASP.NET and any other programming language. It helps organize code, makes code easier to read and understand, reducing the potential for errors. ASP.NET uses a case-sensitive naming convention, meaning all names should be in the same case when used. This is to avoid confusion when a name contains upper and lowercase characters since ASP.NET will interpret them as two different names.

Additionally, ASP.NET allows the use of special characters and spaces within naming conventions. However, this can also be risky, as it is more difficult to read and maintain. ASP.NET coders need to adhere to a consistent naming convention to reduce any risks or confusion in their code.

Avoid Using Session Objects

The ASP.NET programming language provides an easy way to store and access data through the use of session objects. Although it is convenient, there are risks associated with using session objects. For example, ASP.NET stores the data on the server and can lead to a decrease in performance as more information is continuously stored. Additionally, session objects are not necessarily secure and can be easily accessed by malicious users. Therefore, it is best practice to avoid using session objects when coding in ASP.NET.

Using other techniques, such as caching and cookies, are much more secure and efficient ways to manage data when coding ASP.NET. It is also important to remember that session objects are only meant to store small amounts of data that do not need to be on the server. These should not replace other measures for data security, such as encryption and authentication techniques.

Avoid Using Session Objects

When coding in ASP.NET, it is essential to secure user input and output to protect against security vulnerabilities. This includes:

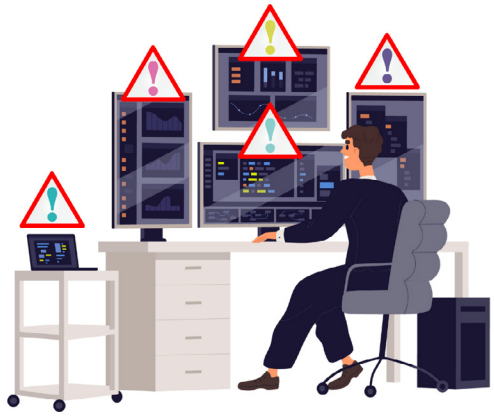
- Validating user input for accuracy and completeness
- Sanitizing it to prevent the introduction of malicious code
- Encrypting output so only authorized users can view it
- Ensuring user input is free from cross-site scripting attacks

Furthermore, developers should also take steps to harden the software by using secure software development practices, such as strong passwords, implementing secure connection protocols and a robust authentication system.

Top Risks for Developing in ASP.NET

ASP.NET development involves risks related to the platform and associated programming language. ASP.NET applications are vulnerable to common attacks, such as cross-site scripting, SQL injection, and authentication bypass. Additionally, ASP.NET's reliance on a stateful programming model requires developers to properly handle session and application variables or risk exposing sensitive data.

The following risks are commonly associated with developing ASP.NET applications.



1 Local File Inclusion (LFI)

The ASP.NET programming framework is vulnerable to the risk of local file inclusion (LFI). LFI occurs when a malicious user attempts to read or manipulate files on a web server exploiting the dynamic nature of ASP.NET. This allows it to execute commands embedded in file names and paths. When ASP.NET processes these malicious requests, it can allow an attacker to access local resource files and sensitive information or even execute arbitrary code on the web server.

ASP.NET developers should be mindful of this risk and take measures to protect their web applications against LFI attacks. This includes implementing input validation, utilizing server-side authentication, and limiting access to sensitive files. Additionally, ASP.NET developers should ensure that all ASP.NET components and library files are up-to-date.

SQL Injection

When developing with ASP.NET, one of the greatest risks is SQL injection attacks. SQL injection occurs when malicious users execute database queries on the web server to gain access to sensitive information or manipulate data. ASP.NET developers must take steps to guard against such attacks by validating user input and using parameterized queries to protect against injection attacks.

Additionally, ASP.NET developers should use caution when creating and manipulating user input as such actions can leave ASP.NET applications vulnerable to injection attacks. Sanitizing user input is also necessary for ASP.NET development as it helps ensure that malicious code is not introduced into the application.

Cross-Site Request Forgery (CSRF)

Websites developed using ASP.NET may be vulnerable to cross-site request forgery (CSRF) attacks. CSRF is a type of attack that uses website vulnerabilities to execute unintended actions. The attack takes advantage of the trust a website has in its users and compromises legitimate requests by tricking users into requesting malicious content without their knowledge.

ASP.NET developers must take measures to protect their applications from CSRF attacks, such as using a Double Submit Cookie pattern for ASP.NET Core or using security tokens for ASP.NET MVC applications. This protects against this type of risk.

ASP.NET developers must also be aware that CSRF vulnerabilities can exist in any component of the application, such as third-party libraries or code snippets. So it is necessary to ensure that all third-party integrations are up-to-date.

Authentication Bypass

The ASP.NET programming language can be vulnerable to authentication bypass attacks. In these types of attacks, an attacker can gain access to a system through a vulnerability in the ASP.NET application code, which bypasses the standard authentication process. This type of attack is often caused by developers not properly validating user input or not implementing secure authentication measures, such as two-factor authentication.

ASP.NET developers must take extra care to ensure their code is securely written to protect against authentication bypass attacks. ASP.NET developers need to stay up-to-date with the latest security best practices and regularly review their code for potential vulnerabilities.

Risks for Developing in ASP.NET Compared to Other Development Languages

Developing ASP.NET applications can lead to many risks that are not present in other programming languages. The following are some key differences between ASP.NET and other programming languages when it comes to security risks.

Reliance on IIS

One such risk is the reliance of the ASP.NET framework on IIS for application hosting, which can lead to issues with scalability and reliability. Hosting ASP.NET applications on IIS can also lead to compatibility issues with other languages and technologies, making it difficult to integrate ASP.NET into existing software architectures.

Additional Licenses and Software

The use of ASP.NET to develop software applications comes with many risks, primarily due to the additional licenses and software needed to host an ASP.NET application. These licenses, such as those for SQL Server and Windows Server, can quickly add up in cost and require additional ongoing maintenance.

Furthermore, the additional software required to host and manage an ASP.NET application can add another layer of complexity when it comes to software and data security. It is, therefore, important to be aware of the need for additional licenses and software when developing in ASP.NET since it puts a heavier burden on the development and maintenance of an application.

Level of Control

ASP.NET also does not provide the same level of control over program execution as other languages, and the ASP.NET runtime can limit the types of applications created. This can present a risk in developing software with ASP.NET, as it may not provide the necessary data and software security to protect the user from possible malicious intent. Additionally, ASP.NET is a proprietary framework, so there is no guarantee of its continued support in the future.

Costs of Not Securing ASP.NET Applications

ASP.NET applications, like any software platform, require strong security to protect data and user privacy. When proper security measures are not in place, businesses face high costs in terms of both money and reputation. Without secure software, malicious actors can access user data or steal intellectual property, resulting in costly data breaches and regulatory fines.

Downtimes

Failure to secure ASP.NET applications can have serious consequences for businesses, including loss of revenue due to unexpected downtimes. Downtimes can arise from software vulnerabilities that are avoidable with proper security measures.

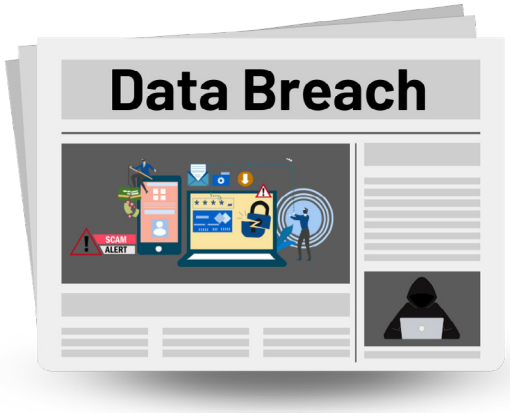
Additionally, the downtime associated with attacks on vulnerable systems can disrupt operations, resulting in further financial losses. To ensure the security of ASP.NET applications, organizations should use static application security testing (SAST) to identify and address any vulnerabilities. SAST provides a proactive approach to software security and can help prevent costly security incidents.

Data Breaches

Data breaches are a weighty cost of not securing ASP.NET applications. Without proper data security measures, any malicious actor can gain access to confidential information stored in the application, resulting in a data breach. These breaches can lead to significant financial losses for businesses and organizations as well as damage to their reputation and credibility. Additionally, data breaches can cause direct harm to individuals whose data is already exposed, leading to identity theft, financial fraud, or other forms of abuse.

Example of a Data Breach

ASP.NET applications are vulnerable to data breaches if not properly secured. In 2019 SmartASP.NET, a company that hosts ASP.NET applications experienced a data breach that exposed over 400,000 records of customer information, including email addresses and passwords. The hackers took down the company's website, as well as customer servers, leading to significant losses for the company.



How Teams Can Work to *Secure* Their ASP.NET Applications

When building an ASP.NET application, teams can take steps to ensure the data security of their applications. By ensuring the application development uses secure coding practices — and by using an authorization service to protect user accounts — teams can ensure their application is robust and secure.

Furthermore, teams should consider utilizing encryption methods to hide data from malicious actors, as well as regular security scans and audits to ensure the identification and addressing of vulnerabilities. Lastly, teams should communicate about data security best practices with anyone authorized to access the application and set up appropriate permissions to prevent unauthorized access.

Benefits of Securing ASP.NET Applications

Securing ASP.NET applications is essential for protecting data and sensitive information from malicious actors. When properly secured, an application can provide a layer of defense that prevents the loss of a company's data and intellectual property while keeping customer and user data safe.

Securing an ASP.NET application also helps organizations comply with privacy and security regulations, reducing the risk of fines or other penalties. Moreover, a secure application provides users with confidence in their online experience and can help build trust between the organization and its customers. Ultimately, security is a crucial factor in building a successful application.

Secure Your ASP.NET Application Today

When it comes to data security, the importance of protecting your application from potential cyber threats cannot be easily overstated. Taking the necessary steps to properly secure your ASP.NET application can help to keep your data safe. With the right security measures, you can ensure that all transactions and communication online are safe from hackers. Secure your application today to protect yourself and your customers from potential security breaches.

SAST and SCA offered by Kiuwan are two important steps to ensure the security of your application. Both solutions provide a comprehensive approach to protecting your data and applications from malicious attacks. Utilizing these methods, you can be sure that any vulnerabilities in your system will be easily identified and patched quickly.

Take the time to secure your applications properly and reduce any risks associated with data security. Invest in SAST and SCA today for maximum protection.

*YOU KNOW **CODE**, WE KNOW **SECURITY!***

GET IN TOUCH:

Headquarters
2950 N Loop Freeway W, Ste 700
Houston, TX 77092, USA

United States **+1 732 895 9870**
Asia-Pacific, Europe, Middle East and Africa **+44 1628 684407**
contact@kiuwan.com
Partnerships: **partners@kiuwan.com**