Cross-site Scripting or XSS is a website security vulnerability that allows a hacker to compromise a user’s interaction with a vulnerable application. XSS attack is a client-side attack where the attacker injects a malicious code in a legitimate website or application and executes malicious scripts in the victim’s web browser. This allows the hacker to perform actions as the victim user, such as accessing the user’s data or gaining privileged access to an application’s data and functionality.

**How Does Cross-Site Scripting Work?**

XSS works by manipulating a vulnerable website into returning malicious JavaScript code to its users. While this malicious code executes inside the user’s browser, attackers can then compromise their entire interaction with the web application.

**What do Attackers use Cross-Site Scripting for?**

Attackers who exploit XSS vulnerability can perform malicious activities such as:

* Carrying out all the actions the user can perform
* Impersonating as the victim user
* Reading all the data accessible by the user
* Capturing the login credentials of the victim
* Injecting trojans into the website
* Virtually defacing the website

**Impact of Cross-Site Scripting Vulnerabilities**

The impact of an XSS attack varies according to the nature of the application, its data and functionality, and the compromised user’s status.

For instance, an application where users are anonymous and the information is already public, the impact will be minimal. However, an application with sensitive customer data like a banking application, email or healthcare data will have serious impact in case of an XSS attack. Moreover, if the compromised user has higher access privileges in an application, the impact will be highly critical. This is because the attacker can then take full control of the application and compromise all user data.

**How to Prevent a Cross-Site Scripting Attack?**

Generally, it’s not very difficult to prevent an XSS attack. However, it can get harder if the application is complex. However, it is possible to effectively plan XSS-related vulnerabilities by using one of more of the following basic measures:

**Filter Input as it Arrives**

When receiving user input, ensure strict filtering by only allowing valid input.

**Encode Output Data**

When the user-controllable data outputs in HTTP response, encode it so that it is not interpreted as active content. Depending upon the context of the output, it may require you to apply a combination of URL, HTML, CSS and JavaScript encoding.

**Use Relevant Response Headers**

In order to avoid cross-site scripting in an HTTP response that does not intend to contain any JavaScript or HTML, use Content-Type and X-Content-Type-Options headers so that the browser interprets all responses according to your desired manner.

Contact Aardwolf Security for detailed cyber security assessment of your website.