

# The Impact of Cross Site Scripting on your business

Andreas Wiegenstein  
Version 1.1 - 2007-07-27

This white paper provides a high level overview of the business risks related to Cross Site Scripting.

## How Cross Site Scripting attacks work

A Cross Site Scripting (XSS) vulnerability exists, if a Web application displays user input without filtering HTML Tags appropriately.

### Example 1

Most Web sites provide a "Search" functionality. Usually the keywords are displayed together with the search results. If you search for e.g. `<b>key</b>` and this keyword appears in bold on the next page, you most probably found an XSS defect.

### Example 2

An online application usually has a "motivation" field. If a malicious user enters tags in it, those tags can affect the page that a recruiter will open later on to examine new applications.

An XSS attack works by manipulating content of a Web application and trick *someone else* into opening that page. In the first example, an attacker would have to build a malicious link and place it e.g. in a forum or send it to someone by mail. In the second example the attack works "by design". The input is stored in the application's database and automatically transferred to the victim.

## Damage potential

Through an XSS defect, arbitrary (JavaScript) code can be executed in the attacked user's browser.

The following is a brief list of the potential damage that can be caused by XSS attacks:

- stealing and continuing the session of the (authenticated) victim
- manipulating files on the victim's computer or the network she has access to
- recording all keystrokes the victim makes in a Web application and sending them to the hacker
- stealing files from the attacked user's computer or the network he has access to
- probing a company's intranet (where the victim is located) for further vulnerabilities
- launching other attacks against systems the victim can reach with her browser (on the Intranet)
- performing brute force password cracking through the attacked user's compromised browser

It is clear from that list, that XSS is a also *compliance violation*, as it affects privacy and accountability. Companies cannot comply to regulations like SOX or Data Protection laws as long as their applications contain XSS defects.

## Critical factors

The danger of XSS defects is that they can *easily be spotted and exploited by attackers*. They are also completely platform independent. Whereas other attacks require a specific operating system or database version to work, JavaScript code runs in virtually every browser on every system.

There are also many technically different variations of XSS attacks, because HTML is a language. And in every language there is more than one way to express the same meaning.

Our experience with numerous security assessments and trainings shows that *even seasoned developers that understand the problem are not able to write code completely free of XSS defects*.

## Conclusion

A single XSS vulnerability in any business application can do substantial damage to a company. Although a Cross Site Scripting attack initially hits a user, it can quickly spread from the victim's browser to many other systems.

Today, more than 80% of all Web applications have Cross Site Scripting vulnerabilities. Due to this large number of opportunities, hackers will adjust their skills accordingly.

It is every company's responsibility to protect its customers' privacy and data through regular security assessments and best practices.

*Someone will find the security defects in your applications. It should better be you, not the hacker.*

## References / Further Information

Online information

- [1] The Web Hacking Incidents Database  
<http://www.webappsec.org/projects/whid/>
- [2] The Cross Site Scripting Threat  
[http://www.virtualforge.de/cross\\_site\\_scripting\\_threat.php](http://www.virtualforge.de/cross_site_scripting_threat.php)
- [3] Cross Site Scripting animated learning material  
<http://www.virtualforge.de/vmovie.php>

Literature

- [4] Gary McGraw, Software Security: Building Security In, Addison-Wesley Professional, 2006
- [5] Michael Howard, David LeBlanc, and John Viega, 19 Deadly Sins of Software Security – Programming Flaws and How to Fix Them, Osborne McGraw-Hill, 2005