## Web Application Security

**Cross-site scripting** - Injecting lines of JavaScript into web pages. If not defended against, a hacker can submit malicious code through the search bar, for example, or post it in a user comment.

**Session Hijacking** - Each unique user is assigned a "session" when they log in to a website. Session hijackers will jump into the session of another user, reading information as it passes between the user and the server.

**Parameter Manipulation** - Websites often pass information from one web page to the next through URL parameters. For example, if you search on Google, your search terms will be passed to the results page through the URL. A hacker can take advantage of this fact to rewrite these parameters in harmful ways.

**Buffer Overflow** - A buffer is a small amount of space allotted to store data. If a buffer is overloaded, the extra data will overwrite data in other areas. Hackers have exploited this knowledge to overfill a buffer, than overwrite other data with their own malicious code.

**Denial of Service** - Denial of Service attacks are simple but effective. They operate by overwhelming a site with requests for information, severely slowing the operation of a website or bringing it down entirely.

**SQL Injection** - SQL injection works similarly to cross-site scripting; in this case, however, it is malicious SQL statements that are inserted into the site. These statements are intended to manipulate the database in some way - either accessing sensitive data, or deleting it entirely, causing major headaches for the owners.

What can you do to avoid these threats? The most important thing is not to underestimate the importance of Web application security - and put your users and yourself at risk. CommonPlaces offers a wide range of [security services](http://www.commonplaces.com/what-we-do/security-services), including industry-leading security scans, code review, and remediation services. [Contact us today](http://www.commonplaces.com/contact) to discuss the security of your site.

You can secure a Web application by restricting access to certain URL patterns in the Web application or programmatically using security calls in your servlet code.

At run time, your user name and password are authenticated using the applicable security realm for the Web application. Authorization is verified according to the security constraints configured in **web.xml** or the external policies that might have been created for the Web application using the Administration Console.

At run time, the WebLogic Server active security realm applies the Web application security constraints to the specified Web application resources. Note that a security realm is shared across multiple virtual hosts.