# Web Application Scanning

As web applications have become an integral part of business activities, the need to perform security checks on internally-developed web applications is a necessity.

To answer this need, Beyond Security has developed Web Site Security Audit (WSSA), a service that can now perform web application level security audits, in manner a very similar to the network level and OS level security scans of its parent product AVDS.

WSSA's web application checks detect with great accuracy whether a web application is vulnerable to web-level attacks. These checks are conducted on all server side scripts, including custom-made web applications.

As custom web applications are usually built in-house and are not commercial applications, vulnerabilities in those web applications may not be detected by the "known vulnerability" checks done as a part of a standard network-level scan. Therefore, a separate scan is necessary.

Custom-made web applications are, by definition, unpredictable in their response to attack. However, certain similar characteristics exist in all vulnerable products. A web application attacked using several different types of SQL injection techniques will respond with a predictable fingerprint, allowing **detection** of vulnerable code.

Layer 7 checks consist of the following groups of vulnerabilities:

1. SQL Injections
2. JSP/ASP/PHP Code Injections
3. Command Execution (through piping)
4. File disclosure (Windows and UNIX style)
5. Cross Site Scripting (HTML and JavaScript injection)

Before any of the above tests are conducted, the WSSA Layer 7 check creates an initial **fingerprint** of what would be a normal behavior of the file being tested. This is done by accessing the scripts, and sending the web form complete, partial, and malformed content for each of the parameters that are provided as input to the page.

After this initial fingerprinting phase, the check goes on to sending for each of the group of vulnerabilities, predefined strings that are known to trigger a specific vulnerability that is part of the group. As the tool has already gathered what the normal behavior of the page is (using the **fingerprinting** process described previously), it can now conclude very accurately that the application misbehaved when it was sent the attack string.

At this point, it would have been very easy to simply print out a list of vulnerable pages, and how WSSA triggered these misbehaviors. However, this may result in many false positives (reporting vulnerabilities that do not exist). To avoid this, WSSA sends special predefined strings that are known to trigger certain behavior in specific back-end programs while not triggering other back-end programs (for example, MS SQL Server vs. MySQL).

This allows WSSA to better ascertain whether the misbehavior is due to an actual vulnerability or due to bad fingerprinting. This dramatically reduces the number of false positives generated during testing.