

CAST AIP – Support for “The Open Web Application Security Project (OWASP) Top 10”

The aim of this document is to describe CAST AIP’s current support and future roadmap for OWASP Top Ten 2010 (the document applies the very same way to OWASP Top Ten 2007).

CAST Strategy for the on-going support of OWASP Top Ten is to provide users with the means to check that valid protection is in place and whenever possible to provide development teams detect places where vulnerability is left in the code.

OWASP 2010 - A1 – Injection Flaws

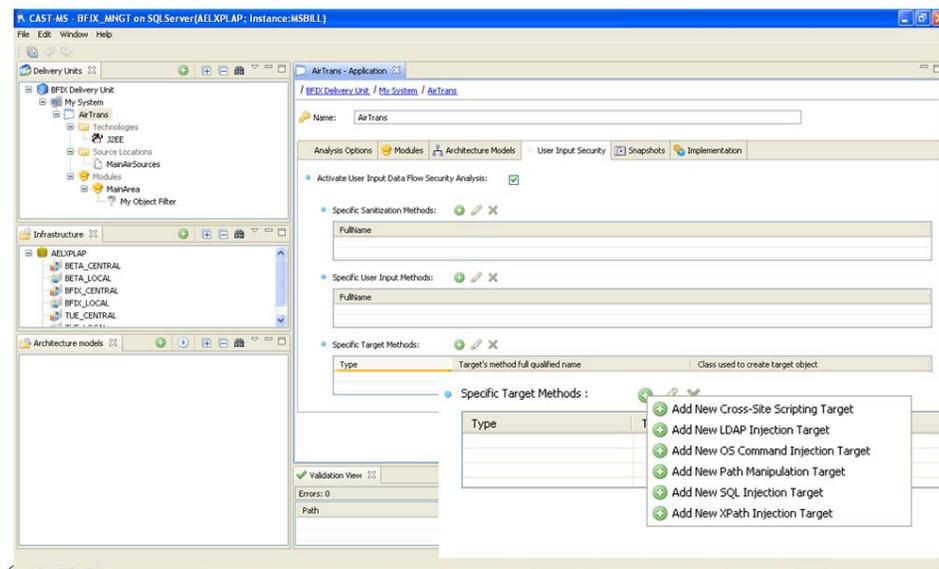
“Injection flaws, particularly SQL injection, are common in web applications. There are many types of injections: SQL, LDAP, XPath, XSLT, HTML, XML, OS command injection and many more. Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. Attackers trick the interpreter into executing unintended commands via supplying specially crafted data.”

CAST AI Platform integrates a User Input Security Analyzer based on dataflow technology. This capability is available for J2EE and will be available in S2 2010 for Microsoft C# (C# beta already available on demand).

The Dataflow User Input Security Analyzer is a technology embedded in the code analyzer that uses tainted variable analysis and tracks input data thru out the source code to find a path where the input data is sent to a target server without prior sanitization / encoding.

The User Input Security Analyzer offers directly supports the following type of injection flaws grouped into the technical criteria metric called “**Secure Coding - Input Validation**” :

- SQL injection flaws
- LDAP injection flaws
- OS command injection flaws
- XPath injection flaws
- File path manipulation flaws



The User Input Security Analyzer can be customized to specific need interactively to add specific sanitization, input or target methods for a given type of flaw.

The Dataflow User Input Security Analyzer offer manager and developers the view of the precise path(s) where the vulnerability exist in a call path view directly in CAST AD Dashboard.

In addition to the dataflow searched quality rules, CAST AI Platform provides out of the box the following framework based rules also grouped into the technical criteria metric called “**Secure Coding - Input Validation**” :

- ASP.NET: Always validate user input with Request variables
- Struts Validator: Avoid action mappings validator turned off
- Struts Validator: Avoid Form Field without Validator
- Struts Validator: Avoid multiple validation form with the same name
- Struts Validator: Avoid unused validation form
- Struts Validator: Avoid Validator field without Form Field
- Struts Validator: Enable Struts Validator plugin
- Struts Validator: Form Bean must extend Validator Class
- Struts Validator: Validator form validate() method must call super.validate()

For application following the OWASP recommendation to implement object relational mapping (ORM) libraries such as Hibernate, CAST provides a specific rule that checks that only Hibernate is used to access the database:

- Hibernate: Use only Hibernate API to access to the database

CAST provides also 13 rules to check that Hibernate architecture & performance best practices are in place.

OWASP 2010 - A2 – Cross Site Scripting (XSS)

“Cross-site scripting, better known as XSS, is in fact a subset of HTML injection. XSS is the most prevalent and pernicious web application security issue. XSS flaws occur whenever an application takes data that originated from a user and sends it to a web browser without first validating or encoding that content.”

Being a special form of injection, a part of the solution to get rid of the XSS threat comes from checking the protection in place for A2 Injection Flaws (see A2). In deed, beside A2 protection (validation of all incoming data), the OWASP Top Ten document recommends the “appropriate encoding of all output data”.

CAST supports checking that using the same technology as for Injection Flaws : dataflow and tainted variable analysis to track input data thru out the source code in order find a path where the input data is sent back to the user without prior sanitization / encoding.

The User Input Security Analyzer offers directly supports Cross-Site Scripting flaws that can be found in the technical criteria metric called “**Secure Coding - Input Validation**”.

As for A1-Injection flaws, the user can customize interactively the analysis to meet specific needs: adding specific sanitization, input or target methods for a given type of flaw is done directly thru the GUI.

XSS results are displayed in the dashboard which offers the view of precise path(s) where the vulnerability exist in a call path view directly in CAST AD Dashboard.

OWASP 2010 - A8 – Insecure Cryptographic Storage

“Protecting sensitive data with cryptography has become a key part of most web applications. Simply failing to encrypt sensitive data is very widespread. Applications that do encrypt frequently contain poorly designed cryptography, either using inappropriate ciphers or making serious mistakes using strong ciphers. These flaws can lead to disclosure of sensitive data and compliance violations.”

CAST supports checking that the OWASP recommended protection is in place thru the implementation of custom quality rules. OWASP recommends to check that the application : *“Do not use weak algorithms, such as MD5 / SHA1. Favor safer alternatives, such as SHA-256 or better”*.

Custom quality rule checks for A8 will check that these wrong API are not used.

OWASP 2007 - A6 – Information Leakage and Improper Error Handling

“Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Applications can also leak internal state via how long they take to process certain operations or via different responses to differing inputs, such as displaying the same error text with different error numbers. Web applications will often leak information about their internal state through detailed or debug error messages. Often, this information can be leveraged to launch or even automate more powerful attacks.”

CAST supports the protection against “A6 – Information Leakage and Improper Error Handling” thru the use of the following rules integrated in the technical criteria metric called **“Programming Practices - Error and Exception Handling”**:

- .NET: Avoid catching an exception of type Exception
- .NET: Avoid empty catch blocks
- .NET: Avoid empty finally blocks
- .NET: Avoid throwing an exception of type Exception
- ABAP: Avoid missing WHEN OTHERS in CASE statements
- ABAP: Avoid unchecked return code (SY-SUBRC) after OPEN SQL or READ statement
- ABAP: Avoid using AT events in combination of LOOP AT WHERE constructs
- ASP: Use of error handling page
- COBOL: Avoid DISPLAY ... UPON CONSOLE
- COBOL: Avoid using HANDLE ABEND
- COBOL: Avoid using HANDLE CONDITION
- COBOL: Avoid using IGNORE CONDITION
- COBOL: Include a WHEN OTHER clause when using EVALUATE
- COBOL: Programs accessing relational databases must include the SQLCA copybook
- Java: Avoid catch blocks with assertion
- Java: Avoid catching an exception of type Exception, RuntimeException, or Throwable
- Java: Avoid declaring throwing an exception and not throwing it
- Java: Avoid direct Class inheritance from java.lang.Throwable
- Java: Avoid empty catch blocks
- Java: Avoid empty finally blocks
- Java: Avoid missing default in switch statements
- Java: Avoid return statements in finally blocks
- Java: Avoid throwing an exception in a catch block without chaining it
- Java: Avoid throwing an exception of type Exception

Java: Avoid using 'java.lang.Error'
 Java: Avoid using 'java.System.exit()'
 Java: Avoid using 'System.err' and 'System.out' outside a try catch block
 Java: Avoid using 'System.err' and 'System.out' within a try catch block
 Java: Avoid using 'System.printStackTrace()' outside a try catch block
 Java: Avoid using 'System.printStackTrace()' within a try catch block
 JSP: Pages should use error handling page
 PL-SQL: Use WHEN OTHERS in exception management
 T-SQL: Avoid Functions and Procedures doing an Insert, Update or Delete without including error management
 T-SQL: Avoid Stored Procedures not returning a status value
 VB: Avoid using "On error Resume Next" in the Class event terminate.
 VB: Use a single Error Handling Method

Other OWASP vulnerabilities:

For A4, A5, A7, A10 of OWASP Top Ten 2010, as described in the OWASP document these vulnerabilities as difficult or impossible to check using source code analysis.

CAST AIP Security Health Factor – Standard Security Rules

For a complete list of security rules that CAST AIP checks for, please email us at info@castsoftware.com

About CAST

CAST is a pioneer and world leader in Software Analysis and Measurement, with unique technology resulting from more than \$100 million in R&D investment. CAST introduces fact-based transparency into application development and sourcing to transform it into a management discipline. More than 250 companies across all industry sectors and geographies rely on CAST to prevent business disruption while reducing hard IT costs. CAST is an integral part of software delivery and maintenance at the world's leading IT service providers such as IBM and Capgemini.

Founded in 1990, CAST is listed on NYSE-Euronext (Euronext: CAS) and serves IT intensive enterprises worldwide with a network of offices in North America, Europe and India. For more information, visit www.castsoftware.com.

Questions? Email us at info@castsoftware.com

Europe: 3 rue Marcel Allégot 92190 Meudon - France Phone: +33 1 46 90 21 00

North America: 373 Park Avenue South New York, NY 10016 Phone:+1 212-871-8330