

Short description: The goal of this GSOC project is to have a ModSecurity version that can be used within Java servers (e.g. Tomcat). In order to achieve this, the standalone C code will be wrapped using the JNI framework and the resulting ModSecurity Java project will be used as a module for Tomcat server. Also, we will collaborate with the OWASP WebGoat team in order to integrate ModSecurity for Java into it.

About me

- **Full Name:** Mihai Pitu
- **University:** National University of Ireland, Maynooth
- **Short bio / overview of background:** I am a 23 years Masters student on the Erasmus Mundus programme, studying Dependable Software Systems at National University of Ireland Maynooth. I have a Computer Science Bachelor degree with First Class Honors at the Faculty of Computer Science of University "Alexandru Ioan Cuza" in Iasi, Romania. For my bachelor thesis I had to develop a [system](#) for Automated Image Annotation as part of a participation at [ImageCLEF 2012](#) conference. In 2011 I was an intern for three months at Continental Automotive, where I have configured a version control, revision control server (Subversion), a bug tracker system (Bugzilla), a project wiki (MediaWiki) and I've also developed a notification by e-mail application (in Java) for the users of the system.
- **Email:** mihaitpitu@gmail.com
- **GitHub:** <https://github.com/mihaipitu> (Only code relevant to this project)
- **Skype:** mikep_1989
- **IRC nick:** mihaipitu
- **Mentors:** Ryan Barnett and Breno Silva

Coding skills

- **Current OS:** I have a dual-boot system: Ubuntu 12.04 and Windows 8, I can easily switch between the two
- **Coding platforms:** I am proficient in C/C++ and Java environments (3 years of experience with Java and 4 years with C/C++)
- **Desired knowledge for this project:** I have developed software systems that used JNI (Java Native Interface) before, for example, as part of the [system](#) I've implemented for my Bachelor thesis, I've ported the [TopSurf Image Descriptor](#) to Java using JNI technology. I've successfully configured Apache and Tomcat servers and although I have never developed extensions or modules for them before, I am confident that with my previous experience I will manage to implement the required ModSecurity module with success. In the past few weeks I've got myself familiar with the [ModSecurity Development Guidelines](#) and I'm currently looking at the best possible ways of developing extensions for Tomcat web server.

OWASP Project: ModSecurity CRS - Port to Java

Introduction

ModSecurity is a web application firewall engine capable of preventing security attacks using a set of application protection rules. The project is currently available for integration with Apache, IIS and Nginx servers. We would like to extend ModSecurity availability to the Java platform and to create modules for the most popular Java web servers, for example, Tomcat.

Our approach is to create a JNI wrapper for the existing standalone source code and to port the project to Java web servers.

Project goals

Our main goal is to successfully create a Java version of ModSecurity which will allow us to port the project to the widely used Tomcat web server. As a testing ground, we will use [OWASP WebGoat project](#) (which also runs on Tomcat) and we will collaborate with the development team for ModSecurity integration. Also, we will provide complete documentation of the project and use cases that will help users easily understand how ModSecurity can be integrated with Java servers.

Implementation

ModSecurity is designed as a web application firewall engine, which means that in order to block a suspicious request it has to have access to the client request made to the web server. Initially, ModSecurity was designed for the Apache HTTPD web server, which provides access to the requests and allows ModSecurity to block them.

In the case of Tomcat web server, a [Valve](#) component can be inserted into the request processing pipeline and modules like ModSecurity can block a specific request or response. Tomcat Valves are tightly coupled to Tomcat API and they are working on container level, which means that a Valve will intercept all applications/requests.

Another option is to use [Java Filters](#) which have the same purpose as Tomcat Valves (perform filtering on either the request to a resource or on the response from a resource, or both). Filters are implemented by all compatible web containers, but they are intercepting requests only to a given web application (if the server is maintaining multiple web applications, a filter has to be specified for each application).

The ModSecurity standalone project is designed as a command line tool, independent of the Apache web server but still implemented using the Apache Portable Runtime libraries. We will use this standalone project as a starting point for our Java wrapper and we will port every ModSecurity API function using [JNI](#) (Java Native Interface) technology, thus allowing the Java Virtual Machine to call (from a Tomcat Valve or Filter), or be called by ModSecurity native code.

The main challenge when dealing with the JNI framework is designing Java classes that correspond to APR structures used in the standalone project. For example, the method from the ModSecurity api.h file:

```
int modsecProcessResponse(request_rec *r);
```

uses a `request_rec` APR structure as a parameter. This parameter will have to be initialized from a Java object (for example, if we use Tomcat Valves, the attributes from the [Request](#) object will be used to initialize fields in the `request_rec` structure) that contains the current client request.

Some structures used in the Apache Portable Runtime are already ported to Java in the [APR native library](#) for Tomcat. However, this is not an exhaustive porting of the APR libraries to Java and some extensions will be needed (for example, I wasn't able to find the `request_rec` structure ported to Java).

As a final step, we will need to specify a versatile way to configure ModSecurity from the Tomcat configuration files and to write complete documentation of the project.

Schedule

My Masters studies will finish on 14 July this year, so I will be able to invest 40+ hours a week after this date and around 20-30 hours a week until this date. The schedule for this project is:

- **April – May:** Getting to know the ModSecurity code, APR libraries and Tomcat valves and filters, defining the requirements and specifications of the project by discussing with the mentors and the community
- **June - July:** Develop the Java part of the project, with respect to the ModSecurity API defined in the standalone code
- **July - August:** Develop the C part of the project, which will consist of implementing the JNI interface specified by the Java code
- **August - September:** Refining the implementation, testing, writing the documentation and integration with OWASP WebGoat project

Motivation

Although I have never been involved in the development process in the OWASP's group in the past and I am a new to ModSecurity, I have extensively researched the software, from a user and a developer point of view, in the past couple of weeks. Also, I've [forked](#) the project on GitHub and I am eager to contribute with my own code even after the GSOC program finishes.

I've chosen to participate at the proposed [OWASP project ModSecurity CRS - Port to Java](#) because it suits my current interests and I think that I have the necessary skills in order to successfully port OWASP ModSecurity to the Java platform and to integrate it with current Java

servers. Also, I think that this project is vital for the OWASP ModSecurity CRS because Java technologies are very popular and web servers like Tomcat or Glassfish are getting more and more market share on the web.

I expect that my implementation will meet the desired requirements and more, for example, the porting could be done for [Tomcat](#) and [GlassFish](#) web server also (or other popular Java servers) and we can integrate ModSecurity to the [OWASP WebGoat project](#) for testing purposes and we can design security lessons or tutorials about how to use ModSecurity CRS.

I am confident that I can achieve the goals that I've set as I have the right skill set and a very strong motivation. I think that this project will be a great success both for ModSecurity and for myself and I am eager to become part of the OWASP and ModSecurity community and to start working on this exciting project.