

# PROJECT SPITFIRE – TOWARDS GRID WEB SERVICE DATABASES (ON BEHALF OF THE EU DATAGRID AND GRIDPPPROJECTS)

William H. Bell<sup>1</sup>, Diana Bosio<sup>2</sup>, Wolfgang Hoschek<sup>2</sup>, Peter Kunszt<sup>2</sup>, Gavin McCance<sup>1</sup>, Mika Silander<sup>3</sup>

<sup>1</sup> Department of Physics and Astronomy, University of Glasgow, Scotland.

<sup>2</sup> CERN IT Division – European organisation for Nuclear Research, 1211 Geneva 23, Switzerland.

<sup>3</sup> Helsinki Institute of Physics, Helsinki, Finland.

**Key words to describe the work:** database data-management meta-data grid web-service Axis

**Key Results:** The architecture of the existing version of Spitfire is presented, along with the web services version currently under development; a practical example is given in the paper.

**Motivation (problems addressed):** The motivation is to provide a secure grid enabled database service to permit access to a wide range of relational database systems. Current database solutions are neither grid enabled nor web service enabled. The combination of grid and web service technologies together with databases provides one of the key building blocks upon which many other grid services and grid enabled applications depend. It permits a wide variety of database solutions to be accessed remotely and securely using a uniform service interface.

Many Data Grid services maintain persistent metadata in remote relational databases. However, existing relational database systems are neither grid enabled nor web service enabled, adversely affecting cross-organizational interoperability and reuse. The European Data Grid's Data Management Work Package addresses these issues with Spitfire. The Spitfire service grid-enables a wide range of relational database systems by introducing a uniform service interface, data model, network protocol and security model. These are based on widely accepted standards and neutral with respect to programming language, platform and database product. In this paper we briefly describe the latest stable Spitfire release. We then discuss in detail work in progress towards a web service based architecture, its security model and client APIs, illustrated by practical examples.

The Spitfire middleware is placed in between the client and RDBMS. The Spitfire client accesses the server remotely, the connection being made using HTTPS. There are currently two versions of Spitfire.

The existing released version (v1.1.0) makes use of the Oracle XSQL servlet [1] for handling the interaction to the database. When a client makes an HTTP request, the parameters of this request are used to fill a SQL template on the server; the URL determines which template to use. The resultant SQL statement is executed on the RDBMS and the result-set shipped back to the client in generic XML format. The security layer of Spitfire uses SSL, with the client's certificate or proxy being sent when the

HTTPS request is made. The authentication mechanisms check that the certificate is valid, properly signed, and that the certificate has not been revoked. Spitfire ships with a basic client capable of utilising the user's proxy. The service can also be accessed securely via a web browser.

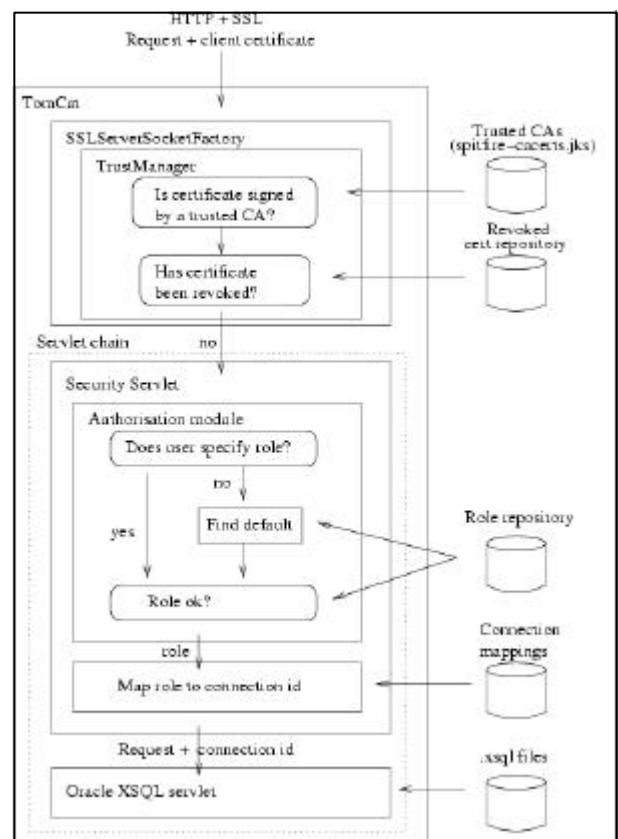


Figure 1: The security model of Spitfire.

The security model of Spitfire is summarised in Figure 1. In order to authorise the client, the certificate's subject name is used to determine the default role upon the database. If a role is explicitly requested, it checks that the certificate is permitted to have this role. Both user-specific and group roles are implemented. The actual role permissions are enforced by the backend database; this removes the need for the Spitfire middleware to parse and check the incoming SQL request.

The new version of Spitfire that is currently under development is based on web services [2]. This will permit the service to be more flexible and powerful, for example, by allowing it to advertise itself to a registry [3].

The service currently runs on the Axis web service engine [4] using the HTTP servlet binding. There is a defined Spitfire API that is implemented using the Remote Procedure Call handler of Axis. This handler is in charge of deserializing the incoming RPC and calling the correct method on the service.

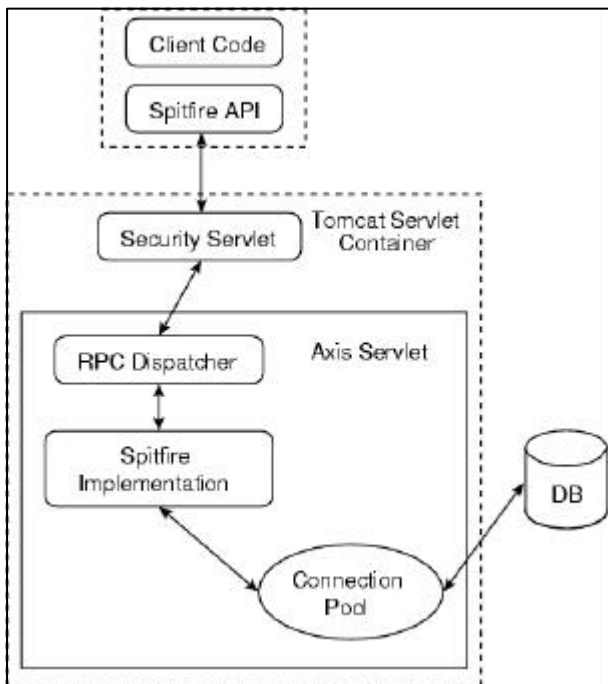


Figure 2: Architecture of the Spitfire web service.

Figure 2 summarises the architecture of the service. The connections to the RDBMS are pooled and handled in an efficient manner in order to limit the overhead for individual calls to the service. The

service can also support longer-term calls, where an open connection from the client is maintained.

The Spitfire API is partitioned into five independent interfaces. A client need only implement one of these to be termed a *Spitfire* client.

- *Grid Database Administration.* This API provides administrative functionality over the Grid enabled databases upon the RDBMS, allowing authorised clients to create, drop and alter databases and tables.
- *User Management.* This permits administrators to remotely edit roles upon the database, the permission levels associated to these roles, and which users or groups are permitted to use the roles.
- *Database Information.* The permits clients to extract information about the database, its associated tables and their schema.
- *Core SQL Functionality.* This implements the common insert, update, delete, and select functionality upon the remote database.
- *High-level functionality.* No higher-level services are defined for the moment. Areas for further work include functionality that will permit replication of database views, distribution and load balancing, and automated clean up services to remove stale meta-data.

The Spitfire software, and full documentation concerning installation and configuration is available for download at <http://cern.ch/hep-proj-spitfire>.

[1] Oracle XSQL servlet (v 9.0.2.0.0.Cb). <http://www.oracle.com>

[2] P. Cauldwell, R. Chawla, Vivek Chopra, Gary Damschen, Chris Dix, Tony Hong, Francis Norton, Uche Ogbuji, Glenn Olander, Mark A. Richman, Kristy Saunders, and Zoran Zaev. *ProfessionalXML Web Services*. Wrox press, 2001.

[3] Wolfgang Hoschek, *A Unified Peer-to-Peer Database Framework for XQueries over Dynamic Distributed Content and its Application for Scalable Service Discovery*. PhD thesis, Technical University of Vienna, March 2002.

[4] Apache Software Foundation. The Axis project. <http://xml.apache.org/axis>

