

## D2K Web Service Deployment: Security Review

Robert E. McGrath, Emily Wu, Doru Marcusi, Andrew Shirk, Terry Fleury, Von Welch

8 March 2006

### 1 Introduction

This note presents a review of security issues for the proposed initial deployment of D2K Web Service, with Evolution Highway (EH) and Phylomat.

We discussed an initial deployment using 3 servers, shared by EH and Phylomat, plus databases on a production oracle server. Figure 1 sketches the deployment. Server 1 runs the servlets that provide the client user interface. This is a Tomcat Web Server with the EH and Phylomat servlets. Server 2 runs the D2K Web Service, which has the itineraries, the jobs, etc.. Server 3 is the compute server that executes the itineraries. We recommend a dual core system for the compute server. Alternatively, several single core systems could be used. The Oracle databases should be hosted separately. We assume that there are Oracle servers already available.

This note discusses the trust model and basic security for an initial deployment. Auditing and accounting issues will be considered in a related document. We also discussed features and requirements that will be needed in future uses.

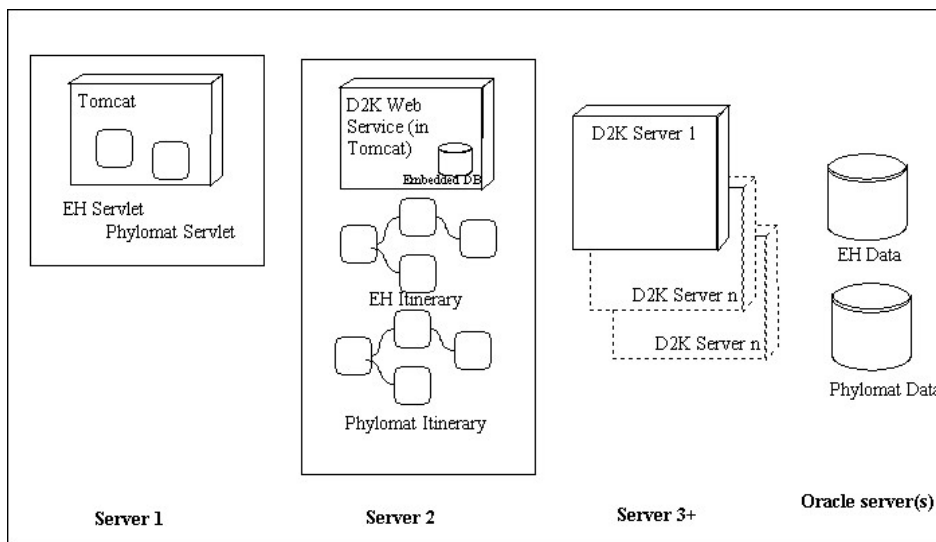


Figure 1. Initial Deployment

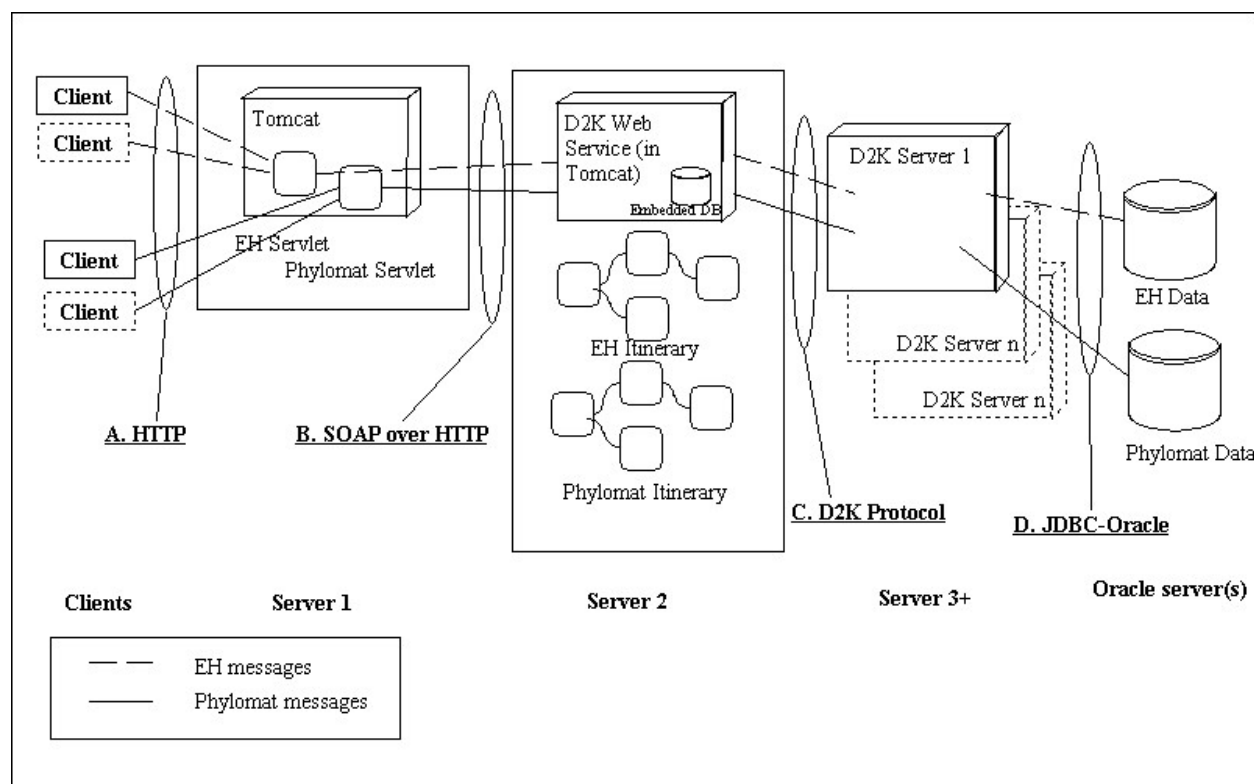
### 2 Security Analysis

This section analyzes the interfaces and messages in the proposed deployment. We define a trust model and sketch how to implement it.

Figure 2 sketches the flow of messages through the system. Clients communicate with the servlets on Server 1. The servlets provide the user interface for each community. Each servlet communicates with the D2K Web Service on Server 2, to launch jobs to run the itineraries associated with the service. The D2K Web Service communicates with one of the D2K Servers in the Server 3 group, passing itinerary and resources to the server to be executed. The executing itinerary communicates with the Oracle server as it runs on Server 3.

While the server resources are shared, each Syberservice has an identifiable flow of messages. Users interact only with the servlets, all the rest of the messages are controlled by the servlets, D2K service, and itinerary.

For example, Evolution Highway users communicate with the Evolution Highway (EH) servlet, which has permission to launch the EH itinerary. The EH servlet acts as the community user for all EH users. The D2K Web Server (Server 3) trusts the D2K Web Service, in that it will run any itinerary sent to it. The itinerary has the URL and login information for the required Oracle data. When the itinerary runs on Server 3, it uses this information to access Oracle through JDBC. Each itinerary has the required access credentials for its own data. In Figure 2, the dashed lines represent communications of EH. The messages flow both directions—results return through the same channels. Phylomat has a similar operation, shown by solid lines.



**Figure 2. Interfaces and messages through the system.**

From these interfaces and messages, we define a trust model:

1. Anyone may use the Syberservice, so there are many users, not limited to current registered NCSA users. Users access a Syberservice through Web browsers or other client software. In the initial deployment, no user authentication is required.
2. Users access the system only through the appropriate servlet (on Server 1) (Figure 2 A). This is the only public interface to the system. The interactions between client and the servlets use HTTP.
3. While this connection uses SOAP over HTTP interface, it is a “private” interface, restricted to use by the EH and Phylomat servlets. The D2K Web Service (Server 2) should be configured to accept connections only from the servlets running on Server 1

(Figure 2 **B**). I.e., in Figure 2, Server 2 should accept connections from the EH servlet and Phylomat servlet. This can be done by making the servlet authenticate as the community user to the D2K Web Service, e.g., with a user name and password for each community. (This authentication requires changes to the existing servlets.)

4. The D2K Servers (Server 3) should be configured to accept connections only from Server 2 (Figure 2 **C**). This can be accomplished using host-based firewalls (e.g. iptables). This is a “private” interface, restricted to use by the D2K Web Service.
5. The Oracle Servers should be configured to accept connections only from the D2K servers (Figure 2 **D**). This can be accomplished using host-based firewalls. Other Oracle security might be used as well. This is a “private” interface, restricted to use by D2K Servers.

We note that there is no strong reason to encrypt resources of the itinerary, nor any need to encrypt the channels between the servers (Server 1 – Server 2, Server 2 – Server 3, Server 3-Oracle). This could be done, but is not essential because:

- all but one of the interfaces are “private”,
- anyone can use the systems, and
- all the data is public.

Therefore, there is minimal risk to the system.

However, it is recommended that the Oracle databases be used as read-only resources, to protect against unintended or malicious modification by the itinerary (or other servers). Since any developer of an itinerary will need access to the databases, we do not anticipate the URL and login information staying private, and so should not grant more rights than minimally required to read the necessary data.

Overall, we conclude that the proposed system is adequate and we can use existing mechanisms to restrict access. No changes are needed in D2K, the itineraries. However, the Web Service should be configured to require a login from each servlet, which will require some change to the servlets.

### 3 Auditing Issues

The model defined in Section 2 should be adequate for protecting the systems. However, it does not necessarily provide adequate auditing, i.e., “who did what when?”. Minimally, it will be necessary to log the activities of the “community user”, EH or Phylomat. This should be possible, although there may be undiscovered requirements.

We noted several uses for accounting and auditing information:

- Auditing (e.g., for security)
- Resource allocation
- Resource accounting (who used all the time?)
- Documenting usage and impact (e.g., to NSF)

These issues will be considered in a second meeting.

## 4 “Phase 2”: Perceived Future Requirements

Beyond the initial deployment, we anticipate additional needs which will require development. We think these issues do not need to be addressed for the initial deployment, but will be important targets for development in the next year.<sup>1</sup>

We need to develop use cases for these features.

### 4.1 *Per User Identification*

Based on other projects and projections of future work, we know that in the future there will be a need for two kinds of per-user identification:

- A user may want to use HPC resources, charging to his own accounts
- The community may want to manage users within its community account.

These requirements are common to most Syberservices, and the technology is under development. It will be important for D2K to support whatever emerges as standard practice.

We noted that these services typically will have broad user bases, not limited to NCSA’s user database. If these users are to be identified, there will need to be some kind of user database. This might be done by greatly expanding the NCSA user list, or it might be done by creating additional databases for “external” users.

This issue will be considered in another meeting.

### 4.2 *Use of HPC Resources*

While the itineraries in the initial deployment do not require HPC resources, it is likely that future deployments will have itineraries that need to access NCSA HPC resources. This will require that the itinerary be able to use appropriate credentials, e.g., from MyProxy.

We discussed two variations of this feature:

- The D2K Server (Server 3) runs on an HPC system
- The itinerary launches jobs on an HPC resource

Either variation requires the ability for the itinerary (on Server 3) to acquire the credentials needed. This might be done by delegation or equivalent procedures, which are yet to be developed.

Again, this requirement is common to most Syberservices, so D2K will need to support standard practice when it develops.

### 4.3 *Uploading, User Administration*

The initial deployment allows users to run itineraries, but they cannot modify or install itineraries. It is likely that future applications will be designed to let users “up load” modules or itineraries, and perform other administrative operations.

---

<sup>1</sup> This is the conclusion from this meeting. Additional discussions may determine that some issues must be dealt with now, e.g., to deal with auditing or other requirements.

The D2K service can support this operation today, but we do not know all the implications for security. One likely approach would be to give a PI or community admin certain rights, while ordinary community members may only run.

A related issue would be found in an itinerary that may update or change the shared databases. The current itineraries only read and analyze the data, updates are performed only by administrators. If the user's jobs can update shared data, there may be a host of issues including synchronization, consistency, and fine grained access control on the data. These requirements might or might not be met by the itinerary and application code, and in any case it may not be clear how to assess the security implications of such actions.

### **4.4 D2K Enhancements**

We identified several enhancements for D2K which are not needed immediately, but may be required or desired in the future.

These issues should be investigated.

#### **4.4.1 Accessing External Resources**

The initial deployment foresees all the resources to be within the NCSA administrative domain. However, we can easily imagine itineraries that want to access data from geographically distributed sources, to launch jobs on external resources (e.g., other HPC sites), or even to send itineraries to D2K Servers at other sites. All of these capabilities can be supported in principle, but the authentication and security issues need to be explored.

#### **4.4.2 Development Environment**

D2K itineraries are developed and debugged in the D2K Toolkit or other environment. Once they are working and demonstrated, they will be installed in the production systems. This installation must not require rewriting the application code, so there must be proper support in the development environment to enable the developers to build the correct authentication and security into the itineraries and modules.

#### **4.4.3 D2K Server Features**

One way to execute D2K on an HPC resource using a specific user ID and account would be to launch an instance of the D2K server owned by the user. This might even be done "on the fly", i.e., the itinerary could launch a D2K Server on the desired resource, and then dispatch the job.

In this and other variations, it would be important to make sure that a particular D2K server should only run certain itineraries from certain sources. This will require options to the D2K Server, and/or might involve identifying particular itineraries (e.g., by a login or a signature).

These features need to be discussed further.

## **5 Conclusion**

We summarized the trust model and basic security for an initial deployment of the D2K web Service to run the Evolution Highway and Phylomat Syberservices. We conclude that the proposed system is adequate and we can use existing mechanisms to restrict access. No changes are needed in D2K, the itineraries.

## Evolution Highway/Phylomat D2K Deployment Security

The D2K web Service should be configured to accept connections only from authorized servlets. The EH and Phylomat servlets should be changed to login to the D2K Web Service.

Auditing and accounting issues will be considered in a related document.

We also discussed features and requirements that will be needed in future uses. These features need to be discussed, to develop use cases and requirements. These will be considered in future meetings.