

## The Learning Matrix and Shibboleth

Dr. Roger Clark, Learning Matrix project manager

### **Acknowledgements**

A number of people have contributed directly to this work including:

Pete Mallinson and John Gilbertson, LSIP project, University of Liverpool,

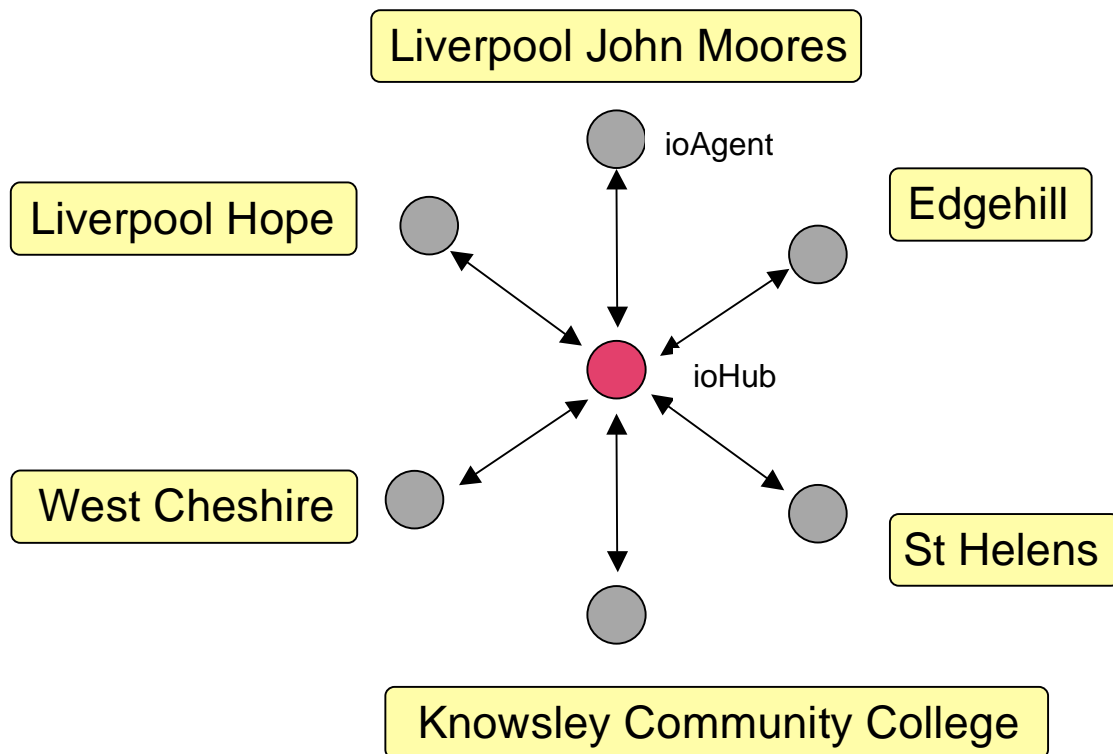
Dimitris Alevizos, Chris, David Hunter, Selwyn Lloyd of Phosphorix Ltd,

Dave Parry, Mark Barratt-Baxendale, Liverpool Hope University,

and a St Helens College representative

### **Background**

The Learning Matrix (LMX) technology substrate consists of a hub/satellite configuration of servers hosting an array of open-source components and services, some specifically written for the “ioNetwork”. The nodes are “ioNodes” (input-output nodes) in the network and pass standards based xml payloads over secure and auditable channels via SOAP/WebServices. The ioNode concept was conceived for the SHELL project by Phosphorix ltd. who continue to develop the notion of an open-source, interoperability network designed as a “black-box”, preconfigured hardware component system.



There are two main interfaces: The Learner Portal, a central arrangement of services, and a private administrative interface to each outlying node's (the ioAgents) main functions in the LMX.

The LMX project pilots a regional service aimed at potential non-traditional Higher Education students. Six HE/FE institutions have ioAgents and offer a selection of short learning packages (LPs) as "tasters" for HE study. Personal Development Planning tools in the Learner Portal are used to support the learner in thinking through and planning their educational future. Through the portal, learners can apply to the institution to take a particular LP, receive communications, and build up a Learner Record automatically from their history of LPs completed. The potential of the technology base is not limited to this particular service and it is expected that there will be other services, quite possibly tracking accumulated credits in a CAT scheme.

Importantly, each institution owns and delivers their own courses using their own resources. A student has to be enrolled and registered on the local systems. There is no sharing of electronic resources between institutions.

### ***The problem space***

The scenario for the future of a set of regional services based on this distributed approach includes a situation for a learner where they may have simultaneous or sequential accounts at several of the local institutions. They may take a course at institution A, complete it and take courses elsewhere and then return to institution A for another course. These episodes of learning imply a short-term enrolment and registration on an institution's system, with possible re-registration at a later time.

Timetable for A Student				
Time	Institution 1	Institution 2	Institution 3	Institution 4
↓				

Key: Part-time, short course study Full-time degree course

The scenario highlights a set of problems for institutions, and for learners. For learners, there is what might be seen as a relatively minor issue around having several sets of credentials to present for access to resources they need for their studies. A well-organised learner may deal with this without problem, but others may have problems with remembering their credentials and which set to present at a particular point. This has knock-on support implications for institutions as well.

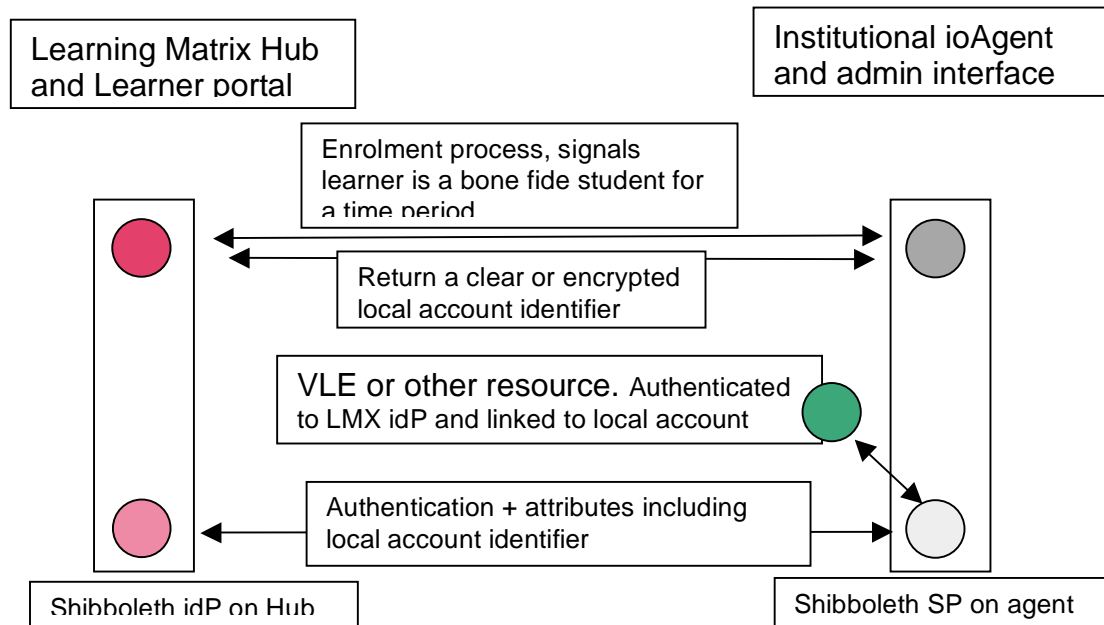
For institutions there is a range of problems around identity management, and the situation of having several separate identities and accounts which actually belong to the same individual, and may need to be aggregated at the point of deciding on awards.

A learning Matrix registered student can use the Learner Portal to locate a course to enrol on at one of the partner institutions that has an ioAgent. There is communication between the ioAgent and the central services located at the Hub, which assists in the process of enrolment, and maintains the record of current and completed courses of study for the student.

Each course of study, even if only for a short period, will usually require access to e-resources, which may include a Virtual Learning Environment (VLE). Currently, the learner would probably have been registered at the institution and have a full account through which they could access such resources. This would mean they have to remember a set of credentials, and recognise when they are appropriate to offer for authentication. They will also have accounts at other institutions where they are studying, and may also take a new course at an institution where they previously studied, but with an “unregistered” period between.

### ***Exploration of a Shibboleth based approach***

The aim of the Shibboleth aspect of the Learning Matrix project is to explore the potential of a shibboleth approach to address at least some of these issues. In this first phase what we would like to achieve is a relatively seamless access to resources for the learner with a single set of credentials, the Learning Matrix credentials.



A guiding principle of the approach is that an institution should be able to join an ioNetwork by installing an ioAgent with its open-source software set appropriately configured, and would then have a complete, or nearly complete, Shibboleth framework to deploy. What we would like to achieve is a solution where:

1. There is a simple out-of-the-box Shibboleth service provider on the ioAgent that can be readily configured to protect any resource on the institution's network, subject to the resource being amenable.
2. An LMX learner can use the route of authenticating to the LMX hub, which authentication is trusted by the resource.
3. The LMX hub "knows" if a learner has an active period of study at that institution and so only authenticates the learner in that case. In addition it will provide authorisation attributes including a token that can identify the learner's local account at the institution.
4. In this solution there is no requirement for an idP on the ioAgent. The LMX project does not require that institutions make their resources available to other partners, or access Shibboleth protected resources elsewhere. However, in the medium to long term there are clear advantages in enabling an idP on the ioAgent, together with easy to use configuration tools.
5. There is no requirement for creating, or membership of, a formal "confederation". A degree of trust exists between the project partners by virtue of their collaboration in the project.

## ***Preliminary investigations: Shibboleth 1.3c versions throughout***

1. IoAgent servers were purchased and software installed and configured on them. The OS used is FreeBSD and there was an attempt made to compile the C++ Shibboleth service provider code onto this platform. This proved difficult and was not achieved completely in the first stages. The steps taken in this attempt are separately documented in the appendix. For testing and exploration one agent was put in place at Liverpool Hope University, who have been the “testbed” for this programme. In the initial stages the software from the Shibboleth Java project was installed on this box.
2. Up-skilling of key personnel was a prerequisite. The only source of expertise in Shibboleth was at the University of Liverpool who were engaged on a separate project, LSIP. Unfortunately, at a late stage it emerged that the personnel involved would not be able to make a significant contribution of their time, as had been planned. They have been able to offer some consultancy and advice which has proven useful. The project manager (in the capacity of technical co-ordinator) took over the detailed management of the Shibboleth work in December 2005. Some time was spent in familiarising with the concepts and software, which has been cascaded to Phosphorix developers and Liverpool Hope technical personnel. Resources on [www.matu.ac.uk](http://www.matu.ac.uk) proved very useful in this.
3. A number of configurations of IdP and SP pairing have been set up as desktop models, for familiarisation and to test possible approaches to achieving project aims, primarily:
  - a. A C++ SP (ShibDaemon) protecting a resource on IIS 6.0, with the standard Tomcat 5.5 based IdP, on WindowsXP sp2 was configured and working to some extent.
  - b. The Shibboleth Java project example configuration was installed, again on WindowsXP. The configuration help documentation here was particularly complete and helpful at providing insight into the shibboleth workings (currently at [http://shibboleth.internet2.edu/downloads/JavaSP/shibboleth\\_eclipse.htm](http://shibboleth.internet2.edu/downloads/JavaSP/shibboleth_eclipse.htm)) . This configuration worked “out of the box” protecting a Tomcat 5.5 resource, a JSP page that displays all supplied headers and so confirming the successful passing of attributes.
  - c. An extension of (b) with Tomcat 5.5 running behind Apache 2.0 connected through Mod\_JK, again on WindowsXP.
  - d. The C++ SP protecting an Apache 2.0 resource, with the standard IdP, on WindowsXP
4. Phosphorix installed the shibboleth IdP on a development server, on FreeBSD (**Apache, Tomcat versions?**). Configuration and testing work eventually produced a working IdP providing attributes provided by the central hub database of Learning Matrix registered students. This was

achieved through simple database connectors rather than LDAP. Further work has been done to integrate shibboleth and LMX authentication such that re-authentication is not required if there is an existing LMX session when a shibboleth-protected resource is accessed. The IdP was initially tested and developed against the remote desktop SPs described above, and then ported to the Liverpool Hope ioAgent.

### ***Possible Options emerging from preliminary investigations;***

The main focus for subsequent work has centred around the problem of allowing the preconfigured shibboleth SP software on the ioAgent to protect web resources on an Institution's LAN. This should be achieved without any need for installing shibboleth components by the institution involved, or significant expertise beyond that normally available in a computer services department running a network.

Five possible approaches have emerged:

1. A consultation with the University of Liverpool staff involved with the LSIP project produced a potential method for using the **Reverse Proxy** mechanism in Apache. Reverse proxy allows a remote webserver's content to be mapped into the web space of a local server. To the client, it appears that the content originates from the local domain. In this scenario, an Apache instance runs on the ioAgent with a shibboleth protected Virtual Server area. The protected area is configured as a reverse proxy for the resource on the institution's LAN with the "proxy pass" and "proxy reverse" apache directives.
  - a. A learner is directed to the protected resource area on the ioAgent's apache webserver.
  - b. A shibboleth authentication is forced, authenticating to LMX hub IdP
  - c. On returning with attributes and a shibboleth session, the learner gains access to the requested resource, which maps the remote web resource to the URL by virtue of the reverse proxy configuration.
2. The "**Lazy Sessions**" mechanism for remotely invoking a shibboleth session. In this scenario, an application on the institution's LAN directs a learner's browser to a particular, shibboleth specific URL on the ioAgent's Apache server. The URL includes a target parameter on the query string, and this is the URL to which the browser is passed after a shibboleth session is established.
3. The **Guanxi** project has developed a distinct Java SP approach specifically designed for protecting resources remote from the host running shibboleth.
4. Using the Java project SP, another approach would be to code a shibboleth protected page to act as a gateway to remote resources.

This could be either as a separate page for each protected remote resource acting as the login page to that resource, or as a generalised mechanism similar to the lazy sessions approach, with a passed target URL.

5. On similar lines to (4), the Java project code works through a Tomcat filter which invokes the shibboleth process but also acts as Resource Manager. This is an exemplar of a Resource Manager and the filter could be recoded to operate differently in order to achieve our intended outcome.

Apart from whether these approaches can be made to work at all, and on what software base, they each have characteristics that need assessing and weighing with respect to a final decision.

### ***Investigation and evaluation of identified possible approaches***

#### **Reverse proxy**

**Java SP:** The Java project SP operates by way of a filter in Tomcat. The filter is configured to intercept certain URLs and to establish a shibboleth session if required in communication with the shibboleth application running at the same host. Once a valid session is established the filter allows the browser through to the requested URL.

Tomcat does not support a reverse proxy mechanism, and searches did uncover any means of adding this functionality.

Tomcat can be added to an Apache instance to provide servlet support, and Apache does support reverse proxy. A configuration of Apache 2.0 with Tomcat 5.5 was set up with Mod\_JK as the connection method, in order to test whether this combination could be used to achieve the result needed (WindowsXP).

**Result:** The reverse proxy setting in Apache intercepts and re-maps the URL, by-passing the Tomcat protected application altogether, so the Shibboleth process is never invoked. A reverse proxy solution is not possible in this arrangement.

**C++ SP:** This SP operates through a separate “shibboleth daemon” application which can be run as a service on Windows XP. A virtual server was configured on Apache 2.0 and set to require shibboleth authentication for access. The virtual server was itself configured with proxy pass and reverse proxy apache directives to map a remote server into the webspace, as required by the project.

**Result:** This configuration works in the way required by the project. The shibboleth process is invoked and upon authenticated access to the virtual

server URL the reverse proxy mechanism operates to present the remote server content to the browser.

**Conclusion:** The C++ version of the SP would be need on the ioAgent for this solution to be used. At the time of the investigation, the problems encountered in trying to compile the C++ SP on a FreeBSD platform had not been solved. The operation of the reverse proxy mechanism might possibly present problems for some resources depending on how relative URLs are constructed. Reverse proxy may be useful combined with other solutions by mapping applications that are not otherwise externally visible into a publicly available server space.

### **“Lazy session” establishment**

“Lazy session” is a mechanism built into the shibboleth project so that an application can itself decide whether a shibboleth session is required for the user and initiate the session by a call to the shibboleth application. This differs from the “normal” method where an attempt to access a protected URL starts off a session. The application has to construct an URL directed at the shibboleth application and containing querystring parameters, one of which being a return URL (the target), which would probably be the calling application. The lazy session mechanism is not well documented, though it is mentioned in the configuration xml files and described in the official shibboleth wiki.

Investigations were undertaken with both the Java SP and the C++ SP, to see whether a session could be initiated and return to a nominated target.

**Result and Conclusion:** The Java SP appears not to support lazy sessions as documented above. Examination of the source code also did not reveal any mention. The C++ SP supports the mechanism which seems to operate as documented and so could be used. Again, the C++ version is a problem on FreeBSD and therefore for deployment on the agent. There is a frequently reported view that there may be security issues with this mechanism, though the substance of this claim is not known.

### **Guanxi SP**

The Guanxi project has built a shibboleth system on a different model to the C++ and Java versions. The Guanxi model separates the resource manager more visibly from the other components. A “Guard” component sits in front of the resource and is configured to communicate with a SAML “Engine” that acts as its representative in dealing with identity providers. Multiple Guards can use a single SAML Engine. Attributes that are passed are eventually passed through to the guard which wraps them and passes them on to the protected application. The Guard operates through a filter intercepting access to a Tomcat application in a similar way to the Java project version.

On inspection of the documentation it seemed that this approach might fit our objectives here, and a software download became available early in 2006.

**Method:** A working pair of SP/iDP was first set up on a Windows XP desktop system to confirm that an iDP was available for the Guanxi test. The software was from the Java project, downloaded as compiled targets, because this system had proved relatively simple and predictable to set up as a desktop system, helped by some full and comprehensible documentation. The compiled guard and engine were then downloaded from the Guanxi project site and the instructions followed to get them to communicate with each other and to the iDP on the desktop system. After configuration, the Engine refused to accept communication from the iDP because of a reported problem with the certificate supplied, despite details of the certificate being added to the Guanxi configuration. As the pairing with the Java SP worked with no problem in this respect, it was not obvious how to rectify this. Consultation with the Guanxi project lead developer, Alistair Young, did not lead to a solution so the desktop system was reconfigured to operate over HTTP instead of HTTPS. This allowed more progress but a different and unresolvable problem emerged. In this case the iDP would not trust the Guanxi engine because it could not locate a metadata entry confirming the link to the Guards assertion consumer. The link had a GUID on the querystring and so changed on every occasion, so making it impossible to add it to the metadata. Further consultation with Alistair Young confirmed that this was a known issue, and that there would be a new version of the software available relatively soon addressing this and other issues. It was also noted that the Guard was designed to protect the Tomcat resource(s) on the same instance of Tomcat, and was not configurable to protect external resources in the way needed by the Learning Matrix.

**Results and Conclusions:** The Guanxi software cannot be used in its current form because it would not be able to communicate with the Learning Matrix identity provider. It also has the same characteristics as the Java project SP in terms of protecting resources held on the particular instance of Tomcat. In the Learning Matrix case resources are on an institution's other servers and ideally the software on the ioAgent's Tomcat would be configurable to protect them and retrieve attributes.

### **Writing purpose built shibboleth protected Tomcat applications for the Java SP**

This approach was taken by the Liverpool Hope team to demonstrate access to a Liverpool Hope resource (an instance of the learning design based SLED player being developed at Liverpool Hope) through shibboleth.

In the supplied configuration of the Java project software, and application called "secure" is protected by the filter which invokes the shibboleth process and eventually passes on the attributes to the secure application. This application consists of a JSP page that displays all headers and other

information about the request, thus helping in testing. Mark Barratt-Baxendale and Dave Parry of Liverpool Hope re-coded the page to collect attributes passed into the headers into a form submitted to the SLED player. A working demonstration has been produced which follows this process:

1. A “learner” attempts to access the secure application on the ioNode’s Java SP implementation
2. Shibboleth is invoked and directs the “learner” to the Learning Matrix hub’s iDP and presents a form asking for Learning Matrix username/password credentials
3. If this is completed with a Learning Matrix identity, the username is eventually passed as an attribute
4. Access is granted to the rewritten “secure” application which packages the username into a form and submits that to the SLED player resource
5. A particular account with that username has been prepared and SLED allows access to that account

(Note: In fact the SLED resource is not public and so the demonstration works from within Liverpool Hope’s network only. The ioNode has an Apache instance running which could be configured as a reverse proxy to the resource which would enable external access.)

The demonstration shows that this approach could be taken, but there are several further steps to take to a complete approach namely:

- The above approach requires a separate “doorway” application to be created on the ioAgent’s Tomcat for each resource. Since the point of this approach is that no Tomcat expertise should be required of an Institution, this presents a problem that could be solved in a number of ways e.g. an interface that autocreates an application page on being supplied with the destination url for the form. The simpler approach of supplying the url on a querystring parameter may not be possible because on our testing, the presence of a querystring breaks the Java project SP.
- To be viable there has to be a system of creating accounts on the resource to correspond to a valid Learning Matrix login. By no means all valid Learning matrix users are authorised to view Liverpool Hope resources. In fact only learners who currently have an ongoing enrolment on a Liverpool Hope course should be given access. In theory, since the Learning Matrix can determine whether this is the case it should only authenticate users for Liverpool Hope who have current enrolments there, thus simplifying the logic required at the institutional end of things. By the same reasoning, the account identification could be supplied by Liverpool Hope as part of the acceptance of a learner on the course and then passed through as an attribute.

## **Rewriting the Java project code**

The Java project's filter invokes shibboleth and eventually maps the supplied attributes onto headers and passes the browser down the chain to the protected application. The filter might be rewritten to pass the learner through to a resource elsewhere. A browser redirect will not have the desired effect because header information will be lost, so the code would have to act as proxy for the browser creating a GET request complete with headers on its behalf. Other approaches are, of course, possible but none were investigated during the project.

## ***Conclusions and recommendations***

A number of approaches to achieving the aim of this project have been investigated. The aim is to be able to provide to Learning Matrix partners a "black box" solution using shibboleth software to ease the problems of access to resources at institutions where a learner has a valid period of study. The solution adopted should require little knowledge of shibboleth or java technologies.

The investigations support the notion that a usable system on these lines is possible, and a demonstration closely approaching this has been produced at Liverpool Hope University. The next stages of LMX development should build on this to take it a step further.

As the investigations show, there are several viable approaches. The most obvious candidate is the "lazy sessions" mechanism because it is built into the official shibboleth C++ release 1.3c for exactly the purpose that we require. Unfortunately there have been difficulties in getting this software to compile on the FreeBSD platform that ioAgents currently use. In terms of configuration tools to support use, this approach needs none at all. All that is necessary is for a protected resource to direct the browser to the shibboleth SP session initiation url and provide a return address.

Second in terms of ease of configuration and absence of programming input is the reverse proxy configuration. In this case an Apache application area on the ioNode is "shibbolised", and is configured as reverse proxy to the remote resource. The remote resource appears to the browser as if it was part of the Apache server's server space. Configuration tools should be simple to create since they would only add directives to Apache config files. Again this requires the C++ SP which does not yet seem to run on FreeBSD.

The remaining solutions have the advantage of being deployable on FreeBSD, but the disadvantage of needing development work and possible creation of more sophisticated configuration tools. The Guanxi software is not possible at the moment, but versions in the pipeline are addressing the problems.

Aside from software issues, the remaining problems and issues are by no means trivial and relate to information flow, security and business processes. The prime question is how a Learning Matrix authenticated learner can be connected to their local user account at an institution where they have a current learning episode. The solutions to these problems may only become articulated by further discussion and investigation.

## ***Appendices:***

### **1. Report on compiling C++ SP onto FreeBSD platform**

#### **Implementation of the reference software as an extension of the ioNode software set (extracted from Shibboleth progress report, 24/08/2005)**

The project will use the software set developed by Phosphorix Ltd to implement the network. The arrangement is a hub/satellite configuration where the outlying nodes are called ioNodes. An ioNode is a FreeBSD platform with an integrated collection of open-source software components, including some specifically developed for the Learning Matrix as well as generic components. The network makes use of SOAP and webservice to route data packages around the network. As part of the Shibboleth component of the project we engaged Phosphorix to add the various Shibboleth software components to the ioNode set, and to the ioHub. This work is clearly of value beyond this particular project, for example to the other Regional Pilot projects that are also using the ioNode approach.

Phosphorix have been extending the collection of open-source components installed as part of their "ioNode" networking concept. The latest build is known as ioNode2 and it was intended that Shibboleth components would be included.

Specifically, the work projected and now (note: as of 24/08/2005) completed is:

- The Shibboleth identity provider (iDP) has been added to ioNode2, installed into the tomcat application server without significant problems.
- A week of work was unsuccessful in installing the c++ build of the Service Provider component (SP) onto the FreeBSD platform of ioNode2. Investigations showed that this software successfully installed onto Slackware and Windows 2000 platforms.
- Other investigations involved configuring ioNode2 onto Red Hat Linux (Enterprise), where both iDP and SP from the Linux RPMs installed with no problems.
- A recently discovered version of the Shibboleth Java SP for Tomcat was found to have problems in the packaging which could require an upgrade of Java from 1.42 to 1.5 onto the ioNode2 platform.

- The latest source code from the Shibboleth CVS has been examined to see if there are ways to mend the binary release for Tomcat (the documentation and libs provided lead to the conclusion that it should work on java 1.42 and Tomcat)
- There have been meetings with members of the Guanxi project, and when their java SP is released it will be looked at.
- Bodington, which has Shibboleth components built in, has been installed.

In summary, then, there are several possible directions for ioNodes if it is to have stable builds of all shibboleth components:

- a) Continue attempts to track down and fix the problems that have arisen on the FreeBSD platform, which has been the OS of choice for ioNodes given consideration of security and maintenance issues.
- b) Abandon FreeBSD and rebuild ioNode2 on Redhat, Slackware, or even Windows.

## **2. List of contributors to the project work**

Dave Parry  
Mark Barrett Baxendale  
Dimitris  
Selwyn  
David  
Chris  
Roger Clark  
Pete Mallinson  
Pete's colleague

## **3. References and resources**