IGE Whitepaper Secure Data Sharing With *AdHoc*

Table of contents

1. Executive summary

In the scientific circles, there is pressing need to form temporary and dynamic collaborations to share diverse resources (e.g. data, an access to services, applications or various instruments). Theoretically, the traditional grid technologies respond to this need with the abstraction of a Virtual Organization (VO). In practice its procedures are characterized by latency, administrative overhead and are inconvenient to its users.

We would like to propose the *Manifesto for Secure Sharing*. The main postulate is that users should be able to share data and resources by themselves without any intervention on the system administrator's side. In addition, operating an intuitive interface does not require IT skills.

AdHoc is a resource sharing interface designed for users willing to share data or computational resources within seconds and almost effortlessly. The *AdHoc* application is built on the top of traditional security frameworks, such as the PKI X.509 certificate scheme, Globus GSI, gLite VOMS and Shibboleth. It enables users rapid and secure collaboration.

2. The pressing need for sharing

A professor, an engineer, and a researcher who have never met before sit down to a conference dinner. One of them has a petabyte database of worldwide historical climate data. The second one owns a weather simulation engine over a cluster of a couple of thousand nodes. The third one has access to a real satellite. During conversation the question comes up: How precisely can we predict today's weather front in Beijing, China? Let's not waste time on discussion, but find out: they open their laptops to form an ad-hoc virtual organization to immediately share their assets. Selected historical data are then being fed to the simulation engine, and the results are compared to the real-time satellite feed. Within minutes, the answer is there for all to see.

Already in the nineties, people clearly understood that they needed to share resources extensively: large datasets, applications or even machines. Long before the pervasive social networks era, the Web 2.0 and the peer-to-peer file sharing mode, this need has already been well-acknowledged in the scientific circles. There, the ad-hoc, cross-institutional, collaborative teams were perceived as conventional. The concept of Virtual Organizations (VO) developed. The VOs were dynamic groups of mutual trust that could be set up immediately[1]. Entities in a VO are able to securely identify and authorize each other, sharing almost everything. As soon as the task is completed, the VO can be disbanded. The first Virtual Organizations were implemented… or were they really?

Not necessarily. The bureaucracy won. It is striking how user-unfriendly the administrative policies could become, effectively fighting productivity. In corporations and universities alike, there is one security bottleneck: the human administrator with his or her procedures. On the pretext of complex technology and fearing responsibility, we have killed productivity and forgotten the promise of on-the-fly and secure yet easy sharing.

As a matter of fact, even though the technology is there, we have not encountered many real Virtual Organizations in action. Instead, we have seen a medical project where, in the absence of an effective sharing mechanism, thousands of anonymous patient records were moved to another institution on a hard drive in a parcel. We have also seen a research center with the highest data protection standards, where a crafty project team has quietly set

1 The Anatomy of the Grid: Enabling Scalable Virtual Organizations, Foster et al, International Journal of High Performance Computing Applications archiveVolume 15 Issue 3, August 2001

up their own Wi-Fi router with a direct Internet cable to bypass the absurd security imposed by their institution. There are numerous other examples where strict regulations are imposed by home institutions which are commonly disobeyed in practice. This situation is far from perfect. Why didn't Virtual Organizations help?

We would like to examine how VOs are implemented today.

The term „resource", in the phrase „sharing resources", can mean: a data set, a remote application, an online service, or physical hardware (a server, a telescope or a microscope). To simplify the discussion, in the following example we focused on the sharing data.

A resource that belongs to Alice (a person, an institution, or a machine) is shared with Bob, provided that Bob can access and use it. The act of sharing will be secured, if no other party (Eve) can access it. What characteristics, what functionality must be present in the secure sharing platform? It needs to have:

- Identity management: prior to communication, Alice and Bob must share some knowledge about a common mechanism of establishing an identity of a dialoguer.
- Authentication: using the identity management scheme, Alice wants to verify that Bob is Bob.
- Authorization: the fact that Bob is Bob does not necessarily mean that Alice wants to share with Bob. Alice will typically need an internal policy or a rule set to decide whether Bob should be granted access (authorized) or not.
- Channel protection (integrity and confidentiality): while the access takes place, both sides want to ensure that no other party (Eve) can intercept or modify the communication, e.g. by eavesdropping, reading the trail of communication, or by acquiring access to Bob's credentials.
- Isolation and local protection: once the communication has taken place, Bob needs to ensure that any confidential data or information he has downloaded remains inaccessible to the third parties accessing his home system. The local protection is typically outside the scope of secure sharing mechanisms and is implemented by the policies of Bob's home institution, but it should not be forgotten. It becomes paramount among the commercial service providers, providing services (such as e-mail server hosting) to numerous clients with the requirement of multitenancy and mutual isolation of data.

A reasonable secure sharing platform should provide mechanisms for identity management, authentication, authorization, and channel protection as well as support for (but typically not implementation of) local protection mechanisms. Do such frameworks exist?

3. The reality

We will examine three approaches: Shibboleth, Globus, and modern document sharing services.

Before we start, we need further clarification on the concept of a Virtual Organization (VO), an idea at the foundation of modern Grid architectures. In brief, a VO is a security domain extended over a set of entities (individuals, services, resources), allowing them to establish and maintain a mutual trust relationship. The VOs are used to interconnect members of different physical institutions and organizations. The VO membership is not constant – it can be enabled and disabled at any time. For example, two entities (i.e. doctor A and hospital B) within the same VO can dynamically establish each others identity and start a secure communication, even if they did not have prior knowledge of each other's existence.

A VO is a group of entities (typically users) that share legitimate need to access similar resources (usually applications or data). The VO membership can be further restricted by other characteristics (attributes) common to the group of entities, such as the role in the organization, or even the membership in informal circles of interest.

3.1. Shibboleth Federated Security

Federated Security is a concept that promises to streamline security when multiple institutions hold onto their treasured resources; but could extract more value out of them, if only they could and would share resources.

Every organization in a federated scheme is responsible for ensuring security, as well as providing authentication and authorization services on its own premises. The organization to which one of the researchers belongs assures his identity. In addition, it can grant resource access to other researchers (whose identity is assured by their home organizations).

No central authority is involved. Obviously, this does require some initial administrative work, possibly signing a form of cooperative agreement between the two institutions; nevertheless the approach addresses the two most crucial problems of security:

a) assuring participants' identity and

b) keeping control of one's resources.

A commonly used federated authentication security framework is Shibboleth, an open-source software from the Internet2 Consortium. Shibboleth is mainly used for protecting Web-accessed resources. Shibboleth enables users to access protected online resources across institutions. Those institutions may differ in methods to authenticate and authorize users. The Shibboleth architecture introduces the following entities: the Identity Provider is an online service that authenticates the user (that service may possibly belong to user's home organization). The resource site runs the Service Provider service, whose role is to protect the resource (typically, the online content that the user wants to access). The single sign-on process (see Diagram 1) works as follows:

1. The user attempts to access the Resource. The Service Provider (SP) issues the Authentication Request and transfers the user to the identity provider.
2. The user is authenticated at the Identity Provider (IDP). The Identity Provider authenticates them (e.g., by prompting for, and checking, a username and a password) and issues the authentication response back to the service provider.
3. The Service Provider checks a response. When the user arrives with the response from the Identity Provider, the Service Provider will validate the response, create a session for the user, and grant the user access to the resource content.

The authentication token will then persist for the follow-up work - the user will not have to enter the password again until their session expires.

An important aspect for Federated Authentication is that Shibboleth supports the situation where the Resource, the Identity Provider, and the user's home institution are at separate systems or institutions. Similarly, it allows to implement scenarios where various users use several Identity Providers, each with their own authentication mechanism.

Shibboleth enables a truly federated and decentralized security model. What it does not fully enable, is the dynamic nature of a Virtual Organization. It would be favourable if the resource owner was able to define resources with some flexibility (e.g. state that each directory is a separate resource), as well as grant and deny access to those resources to various groups of people in the real time. As we shall see later, this is possible to implement on top of Shibboleth, thus quite useful.

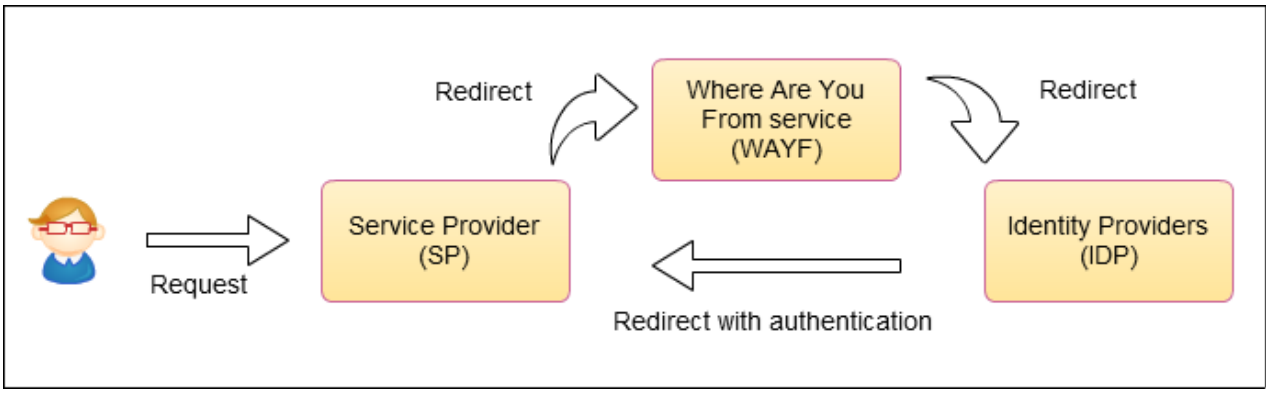

Diagram 1: The Shibboleth's Single Sign-On

3.2. Security in the Globus Toolkit suite

Globus Toolkit is an open-source package consisting of libraries and programs which enable resource sharing within the Grid technology. Its core services and interfaces enable secure resource sharing. The whole is packed into one Globus Toolkit tool pack, but its individual components can be used independently.

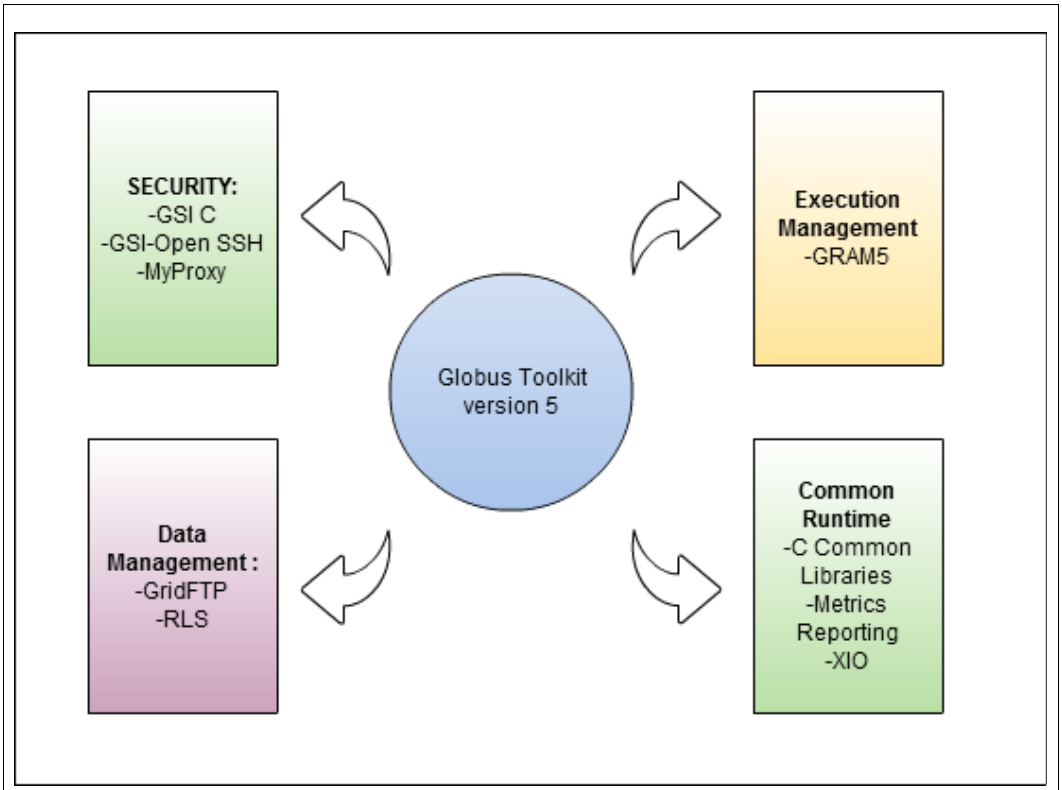

Diagram 2: Modules included in the Globus Toolkit

The security in Globus Toolkit is ensured by **Globus GSI** (Grid Security Infrastructure). GSI is a module designed to support authentication and authorization through X.509 certificates. Each user and host has an issued certificate signed by a trusted Certification Authority (CA), by which each is identified. This certificate contains all necessary information required by the authentication process:

- The name of the entity it represents.
- The public key assigned to the user.
- Certificate Authority (CA) Identity – signing the certificate.
- Digital CA signature.

If we consider the communication between the two resources, we need to be sure that the whole process is secure. If both communicating parties have presented certificates signed by mutually trusted Certificate Authorities, the communication between them may occur on the basis of mutual identification. The GSI uses the Secure Sockets Layer (SSL) for its protocol of mutual identification. The process is called authentication.

For the authentication to take place, both parties must have the copy of CA which signed the certificates. In order to perform authentication, PersonA establishes a connection with PersonB – PersonA sends her/his public certificate to PersonB. At the beginning, PersonB must be assured that she/he can trust PersonA. She/he verifies the validity of the certificate by checking the digital CA signature (the validity and integrity of the signature).

Then, PersonB must be ensured whether it is the right person by sending a random message to PersonA and asking to encrypt the response. PersonA encrypts the message with her/his private key and sends back the message to PersonB who decrypts the message using the public key. If the messages can be decrypted, PersonB can trust the identity of PersonA. Similarly, the operation takes place in the opposite direction to make it mutual.

Furthermore, the GSI supports the idea of a Single Sing-On through the use of proxies (grid-proxy-init command). The proxy is actually a new certificate (with the new public and private key). It contains slightly modified certificate content indicating that it is a proxy as well as information about the time after which it should not be accepted. Such generated certificate is signed by the proxy owner, not by the CA.

Having elaborated on proxies in the GSI authorization, the mapping can be defined. Each user who wants to access a resource on the Globus grid has to be authorized by the GSI and mapped onto a local UNIX user to be able to use resources. It is done by the grid-mapfile file. In its traditional form, this file contains mappings of user's DN (found in her/his certificate) to a UNIX account created and maintained by the UNIX administrator of the resource. A typical scenario assumes a user who wants to run a job on a grid. When the job Manager tries to access the computational nodes GSI (globus-gatekeeper), it tests the user's credentials against the grid-mapfile entries and if successful, it grants access to the resource for job execution. If no entry for the specified user is defined or the proxy has expired, the user is denied to access the resource. This scenario slightly differs when using the VOMS and LCAS/LCMAPS framework.

EMI VOMS (Virtual Organization Management Service) is a service used for the management of the Virtual Organizations and its members. It consists of a MySQL database of the Virtual Organizations, Users, User-VO affiliations and privileges of a given user in the Virtual Organizations. The database is frequently maintained by a VOMS administrator through a Web interface. Registration to the VOMS as well as any other operation on either users or the Virtual Organizations invariably involves the VOMS administrator's participation. The VOMS introduces a new command “voms-proxy-init” as opposed to traditional “grid-proxy-init” that enables users retrieving the VOMS VO affiliations details in a proxy along with standard proxy information. This functionality allows the GSI to use this information for authorization and mapping through a framework created specifically for this by the Nikhef University, which is called LCAS/LCMAPS.

LCAS/LCMAPS (created by Nikhef university) is a framework that expands the mechanism of the Globus Toolkit's GSI (authentication and authorization). It was designed in order to use fully the capabilities of the VOMS attributes in a proxy. It consists of two main components:

1. LCAS (Local Centre Authorization Service) – a component responsible for authorization of a user based on special authorization files (allowed_users and banned_users). Whether a person is in one of these files, depends on the decision to grant her/him access to the resource.
2. LCMAPS (Local Credential Mapping Service) – a component responsible for mapping of users or group of users to a UNIX account or a group of accounts.

The framework's most important functionality is to allow advanced mappings in the grid-mapfile file, such as the possibility to map a Virtual Organization to a pool of UNIX accounts based on UNIX group. In detail, an exemplary grid-mapfile entry might consist of the mapping below:

```
/lcmapsVO/lcmapsGroup/* .lcmaps
```

This entry means that each user has the VOMS group “/lcmapsVO/lcmapsGroup/” affiliation in their proxy, will be mapped to a user from a “lcmaps” UNIX group. She/he will be assigned a username of a pattern “lcmapsXXX”, where “XXX” is the next free user number of the “lcmaps” group. Before it can be used, several users have to be created with their home directories ready to use e.g. lcmaps001, lcmaps002 and lcmaps003. This kind of a change of behavior of authorization and mapping of the GSI allows more flexible authorization management. Several people might be added to the group “lcmapsGroup” in the VOMS and all of them will be automatically mapped to a UNIX user. In a traditional approach, the DN of each of users would have to be added to the grid-mapfile file and all of them would need to have their unique UNIX accounts - this involves a lot of the UNIX administrator's attention. It is important to mention that the administrator's attention is still needed to add the aforementioned line in the grid-mapfile for the mapping to take effect.

Taking all these technologies into account, a typical scenario of the Globus, VOMS and LCAS/LCMAPS suite resembles the diagram below. Users A and B want to use a resource that belongs to user B. They want to use the Virtual Organization approach because in the future there might be another person joining them who would also like to use the resource automatically. It can be done by creating the Virtual Organization in the VOMS and based on this organization, grant access to a pool of UNIX accounts.

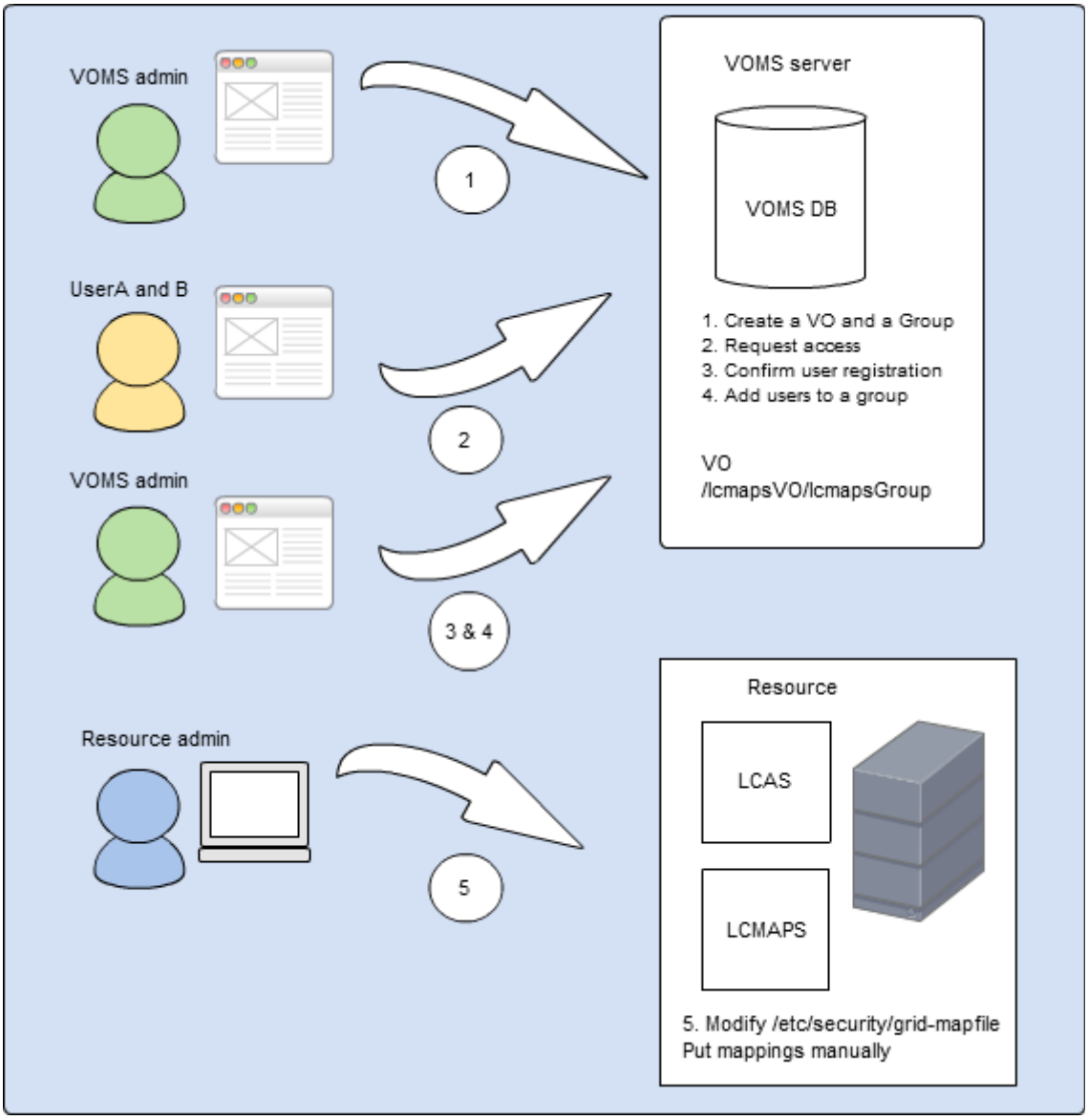

Diagram 3: Standard approach to VOMS and LCAS/LCMAPS

At the beginning, two users express a wish to have a separate Virtual Organization for their sharing purposes.

1. The VOMS administrator creates the Virtual Organization called lcmapsVO and a group within which the VO is called lcmapsGroup. This step cannot be done by anyone else due to the credentials and security issues with the VOMS. Only a person that has access to execute commands in the UNIX (voms-admin command) is eligible to do this.
2. Users A and B request access to a newly created VO and a group. They login to the VOMS server web page and fill the registration form. After that they have to confirm their e-mail address by clicking a link in an e-mail they received.

3. The VOMS administrator receives information that two users want to register to the VO. She/he accepts the registration for each user.
4. The VOMS administrator adds two new users to a group in a newly created VO and fixes their privileges.
5. In the end, the resource administrator logs in through a UNIX console to a resource and modifies the grid-mapfile file manually to add mapping from a newly created VO to a pool of accounts that was created for that purpose.

In none of these steps, neither User A nor User B was able to do the work by themselves without the knowledge of the VOMS or UNIX administration.

3.3. Common document sharing frameworks

Since the advent of Grid computing and the resource sharing in the late 1990's, the IT world became more advanced, and – in many cases – simpler. The rapid development of the Web industry demonstrated the predominant drive towards simplicity, and showed that many things can be made simpler than we thought. The same applies to sharing of data and resources. Solutions commonly used include tools such as Dropbox or Rapidshare, where users can share any files, GoogleDocs, where real-time, remote collaboration can take place, Flickr or Picasa, where photos but also diagrams can be shared, BitTorrent or Gnutella to exchange gigabytes, Skype, WebEx or Jabber for conferencing, or Basecamp, MindMeister, Bugzilla and numerous online collaboration tools that share characteristics of social networks with sharing capacity. In fact, for simple sharing scenarios there is a multitude of choice of platforms today.

Simplicity is a strong argument for using one or more of these frameworks, and the academic community will – sooner or later – adopt them informally, often breaking the rigid non-sharing regulations imposed by their home institutions. So why not doing so formally and officially?Today there are two main reasons why commonplace sharing and collaboration tools are not widely adopted in academia:.

1. They are often not integrated with the security scheme in force at a given institution, such as PKI, LDAP, ActiveDirectory etc.
2. They work well with simple resources such as files, but they do not work well with advanced resources, such as servers, jobs and databases.

3.4. Summary

The example given at the beginning of this paper shows that the VO is a very powerful concept. Over the past decade, most grid installations have implemented VO's using derivatives of the Grid Security Infrastructure (GSI) package, which originated from the Globus project, while many other universities adopted Shibboleth. The GSI-based VOs integrate well with local certificate-based security schemes, but generally miss the dynamic nature that is at the core of the VO concept.

Undoubtedly, the IT industry and Computer Science has evolved over the past ten years. For today's scientific community familiar with one-click sharing via Dropbox, the Grid sharing model may appear unattractive and old-fashioned due to its inflexibility, command-line approach and inability to achieve anything without a system administrator. If the Grid community is to gain traction and clientele, it needs to keep up to date with modern sharing paradigms.

What is therefore needed is a sharing mechanism that is simple to use, quick, yet fully integrated with the institutional security schemes such as the PKI/GSI derivatives.

4. To the rescue: AdHoc sharing

4.1. The history and origin of AdHoc

In the end, we can observe how the concept of the Virtual Organization underwent changes from the original idea. To match the administrative needs of the home institutions, the VO generally became a static structure managed by the system administrator, where changes could only be applied if requested in writing. As the result, when users met with the need of immediate sharing, they ended up finding creative ways to bypass the security domain, which in the end is not that difficult (Dropbox, Google docs, pendrive, or omitting the security scheme altogether by providing one's password to a colleague).

Is it rational to protect all the data to the highest degree? In the case of mission-critical data, we agree. But we have grown mature enough to understand that there are various degrees of confidentiality in various data. In many cases, imposing an equally strict security regime on all data is absurd. As an example, we would like to elaborate on one of many collaborative projects that span two institutions, where putting together sections of experimental data could yield interesting results. What is the risk for data leakage? Indeed, negative consequences are possible: a competing project team getting hold of the data and publishing their results faster. Is this critical? Sometimes yes, but frequently - not (because the data is not that difficult to obtain, or the data quality is questionable, or simply the data is too complex for an external person to use properly). Hence: it should be up to the researcher to decide how strong security should be put in place.

AdHoc is the software based on these assumptions. *AdHoc* was developed in 2007-2009 in Virolab, a European consortium of 14 universities jointly performing research on the HIV virus. They needed to share significant amount of data, databases and applications, all distributed among eight countries. With the current state-of-the-art HIV science, clinical treatment of patients is more effective if supported by the historical data from many thousand past patient cases, and this data can only be collected from numerous hospitals in many countries. *AdHoc* was created to make this sharing easier.

4.2. The Manifesto for Secure Data Sharing

Virolab is a great example of how effective data sharing could not only add value to the research, but even lead to saving lives. In this project, we proclaimed the *Manifesto for the Secure Data Sharing* with the following postulates:

1. **Free the ordinary users! Let they themselves decide on sharing.**

   Today, institutions leave this decision to an expert or an administrator. But that's not practical. Everyone owns data these days. Sharing data is not an IT concept any more, it is our daily bread. Social networks, Web 2.0 and ubiquitous P2P file shares, whether we want them or not, demonstrate it best. So: administrators should guard the data critical for the enterprise. Temporary data of a research project should be owned and managed directly by the team.

2. **User-friendly data sharing interface should not require IT skills.**

   People tend to confuse skills of a Security Officer with skills of a sysadmin. To take ownership of data and to make conscious sharing decision, one needs to be mature, responsible and aware of consequences, but it should not be necessarily to be an IT expert. Because sharing is not an expert action, it should not need an expert's involvement. Optimally, it should be a drag-and-drop GUI.

3. **Data sharing must be easy, efficient and take seconds.**

   In some institutions it can take a significant time to grant someone access, mainly due to the involvement of administration. We know cases when the procedure to add members to a group can take hours or even days. Multiple levels of human, manual authorization may be required by the home institution. Technology should enable, but not enforce such model. Technology should also enable sharing in seconds, if that is the choice of the resource administrator.

AdHoc has been created as a security framework that follows the philosophy above. *AdHoc* enables a creation of truly dynamic Virtual Organizations. In *AdHoc*, anyone can create and administer their own VO, as the *AdHoc* VO becomes again what it should be: a truly dynamic “group of friends of Alice”, to which Alice can add (or remove) Bob at any moment in time. Next, she can grant (or deny) the VO access to any resource she owns, again in real time. Decisions are effective instantly, a fraction of a second after the mouse click.

With *AdHoc*, Alice can execute sharing in three steps:

1. Create a group (a Virtual Organization),
2. Add Bob to the group,
3. Add resource (a server, a database) to the group.

At this point, Bob has access to the resource. All the three steps that Alice makes are drag-and-drop actions in the *AdHoc* GUI, and sharing is executed within seconds. What is more, after Bob is done using the resource, Alice can equally easily remove the resource from the group, which automatically blocks further access to the VO members.

Importantly, *AdHoc* does not reinvent the wheel and does not involve any new security paradigms. *AdHoc* implements the above scenario on top of the existing security frameworks. To demonstrate how it is possible, we have an implementation of *AdHoc* on top of Shibboleth, and a separate one on top of PKI, Globus GSI, glite and VOMS. Think of *AdHoc* as a clever GUI sitting above of one of those traditional security frameworks. But it is not just about using the GUI instead of a command line. It is about dynamic sharing. Suddenly, sharing becomes simple, rapid - and fun.

5. AdHoc scenarios

5.1. How AdHoc sharing was made possible with Shibboleth

In the aforementioned Virolab project, *AdHoc* was integrated with the Shibboleth authentication in order to enable the virologists from various hospitals and universities to work together. Also, resources of various types were instrumented to work with *AdHoc* and Shibboleth. The virologists would often work with resources such as SubVersion (the popular version control repository), and various open-source databases such as MySQL.

In a typical scenario, two virologists from different institutions (here: *virologistA* and *virologistB*) want to execute scientific experiments in order to search for a new HIV treatment. To perform all the necessary experiments, scientists require access to two geographically separated resources: ShibSVN that contains predefined experiments sources, and the DRS database that contains patients data. To enable this, the resource owner (administrator) opens the intuitive *AdHoc* GUI which looks as follows, and performs three steps described below.

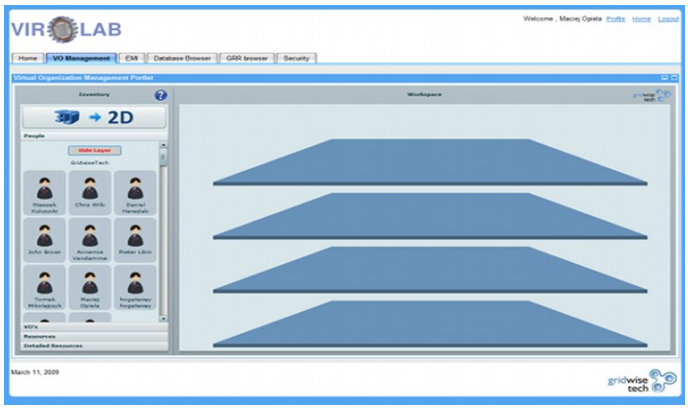

Diagram 4: AdHoc without sharing established

1. The resource owner (administrator) uses the *AdHoc* online GUI to create a new Virtual Organization.
2. Next, the administrator adds two virologists from remote institutions (here: *virologistA* from *homeOrganizationA* and *virologistB* from *homeOrganizationB*) to newly created VO.
3. Finally, she/he also adds two ViroLab resources: *ShibSVN* and the *DRS database* that are needed to conduct the experiment. She/he connects each component with the VO by drawing a line between them.

At this point, the *AdHoc* GUI looks as follows.

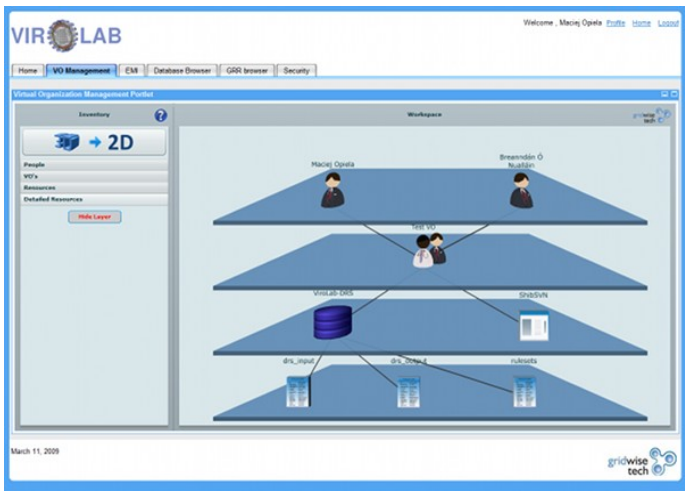

Diagram 5: AdHoc with sharing established

As soon as these persons are connected with resources forming the new VO, changes in access policies take effect almost immediately. VirologistA or VirologistB can access both resources instantly for SVN operations (e.g. *checkout*) or DRS operations (e.g. *update table schema*). After all the results from experiments have been generated and administrator has been notified by the Virologists that they had finished their work, the administrator removes the VO, effectively revoking the users' permissions to these resources.

It is important to note that all this has been implemented in accordance with the Shibboleth paradigm, using the typical Shibboleth authentication/authorization mechanisms. How was it made possible? In fact each of these simple drag-and-drop actions implement quite a complex Shibboleth activity that happens underneath:

1. It is the user's home organization that stores information about the VO, its owner (administrator) and its members.
2. The membership of a person in a certain VO is also implemented inside that person's home organization, and stored as an LDAP parameter associated with the given user's distinguished name. If a VO administrator adds the two Virologists to the club, their LDAP entries receive additional parameters (roles).
3. When a resource (SVN or database) is made available to the VO, it is its local access policy that is modified. However, the local access policy never stores access rules about the individual users. Using role-based access control (RBAC) approach, the policy is made aware that users hold a particular role (belonging to a certain VO) should be granted access.
4. When a user attempts to contact the resource presenting the Shibboleth handle, it will inquire with that user's home organization about the user's VO membership.

The result? The complex security activity indeed happens underneath, but what the user experiences is an intuitive drag-and-drop sharing that does not involve the IT know-how skills.

5.2. How AdHoc sharing is possible with VOMS

In the Globus world, *AdHoc* has been integrated with widely used Globus tools and its authorization / authentication frameworks. There are two main technologies involved, namely the VOMS server and the LCAS/LCMAPS security framework. Security is built on the top of GSI and X.509 certificates. The diagram below shows the main components of this integration with *AdHoc*.

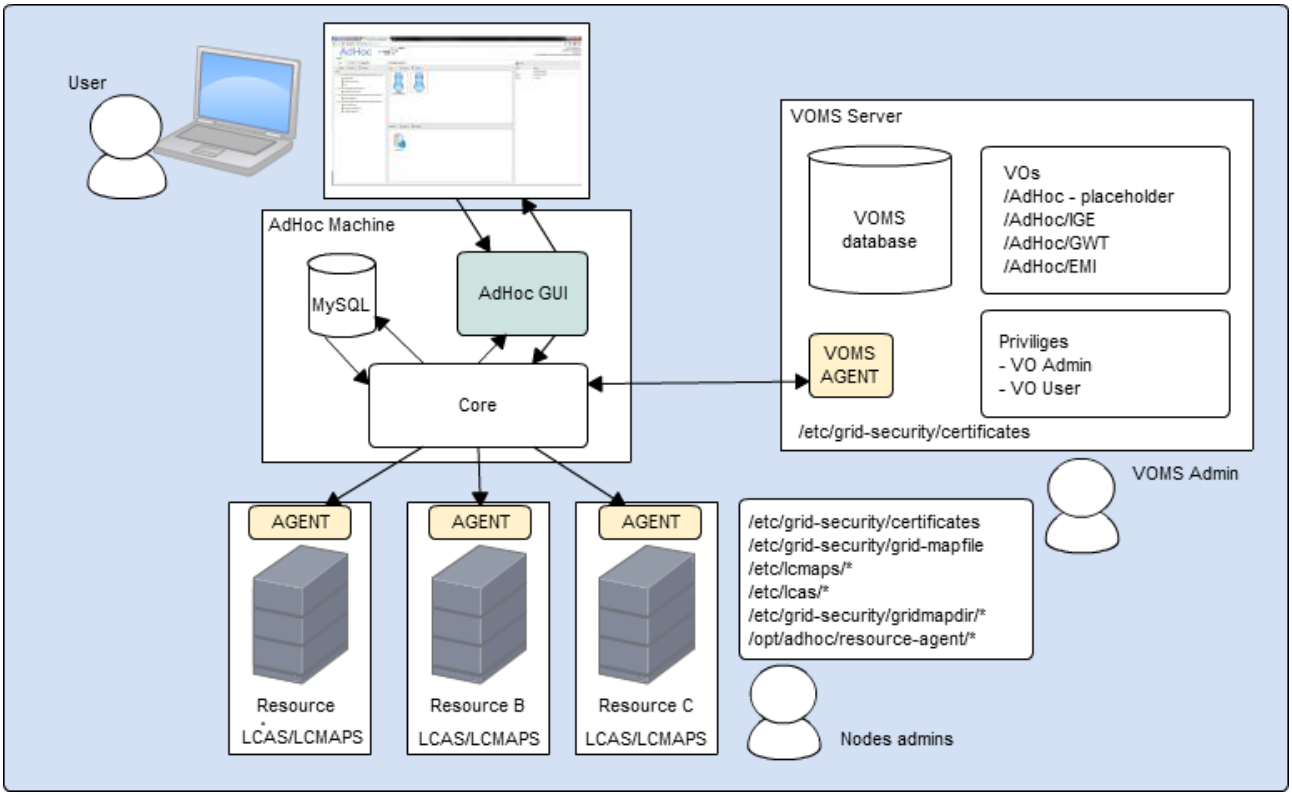

Diagram 6: AdHoc approach to sharing with VOMS and LCAS/LCMAPS

The Virtual Organizations that are the basis of the concept are set up and maintained in the VOMS server. *AdHoc* connects to the VOMS server through an API which enables *AdHoc* to execute basic and advanced operations on the VOMS server remotely (without the VOMS administrator involved). To enumerate some of the capabilities of the API, it can be used to create / delete the Virtual Organizations (VOMS groups), add / remove users to / from the VO, grant / revoke privileges to users in the VO, but also to register a user in the VO. These operations are possible due to a behind-the-scenes VOMS user, which is the VOMS VO administrator who is executing all the operations. It is also worth mentioning that there has to be one VO that the behind-the-scenes user is an administrator of. All the so-called AdHoc VOs are groups of this VO in terms of the VOMS nomenclature (/AdHoc/IGE, /AdHoc/GWT, AdHoc/EMI). A typical scenario of this behavior has been depicted below:

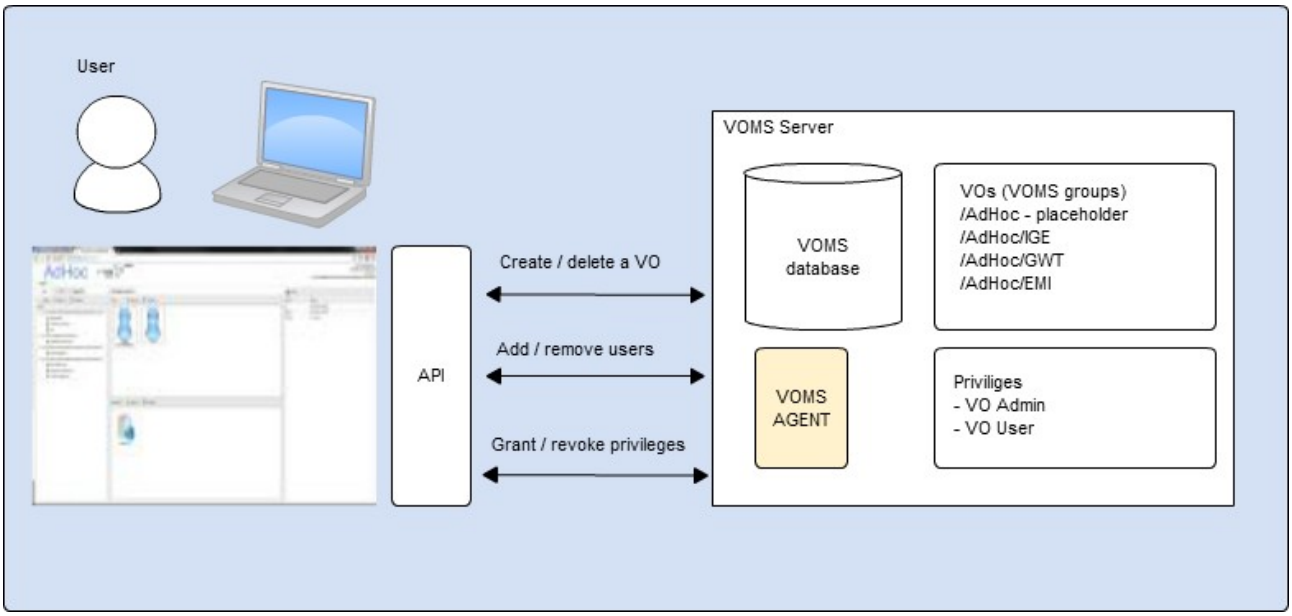

Diagram 7: AdHoc approach to sharing with VOMS

Resources in the Globus environment are the computational nodes and storage. Over the years, the Globus community has developed and maintained the GSI and security frameworks to grant access to these resources using X.509 certificates. *AdHoc* exploits these technologies to integrate resources with the VOMS Virtual Organizations. More explicitly, *AdHoc* connects to the grid-mapfile, the file responsible for mapping unique users' DNs to Linux accounts. The combination of VOMS (EMI) and LCAS/LCMAPS (NIKHEF) grants a unique capability to the authorization process, which is the mapping of VOMS VOs, groups and roles to Linux user accounts and a group of users. This enables the administrator to map a group of people to a pool of Linux accounts which is far less time-consuming in terms of maintenance. What *AdHoc* does in this case is to modify the grid-mapfile to reflect the VO affiliations made by resource admins. This functionality was depicted below.

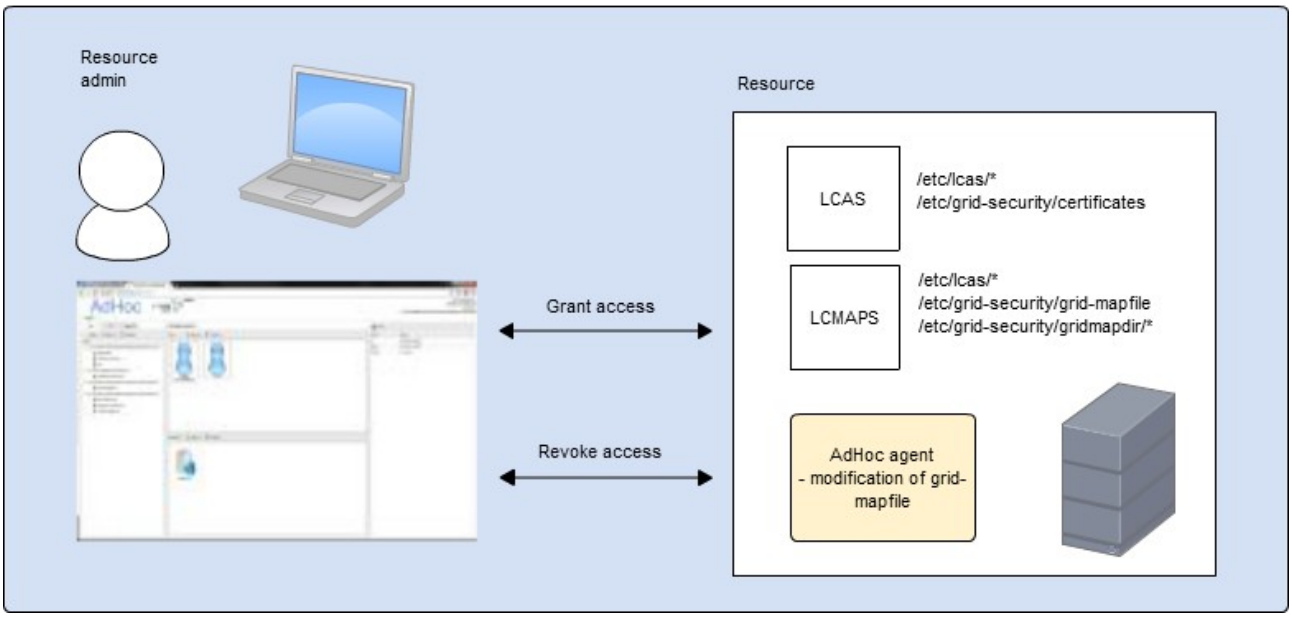

Diagram 8: AdHoc approach to sharing with LCAS/LCMAPS

A typical scenario of integration of all the above technologies can be viewed below. UserA wants to use resources owned by UserB. For this purpose, she/he creates a Virtual Organization and adds UserB as a member. UserB adds her resources to this newly created Virtual Organization which enables both UserA and UserB to user UserB's resource.

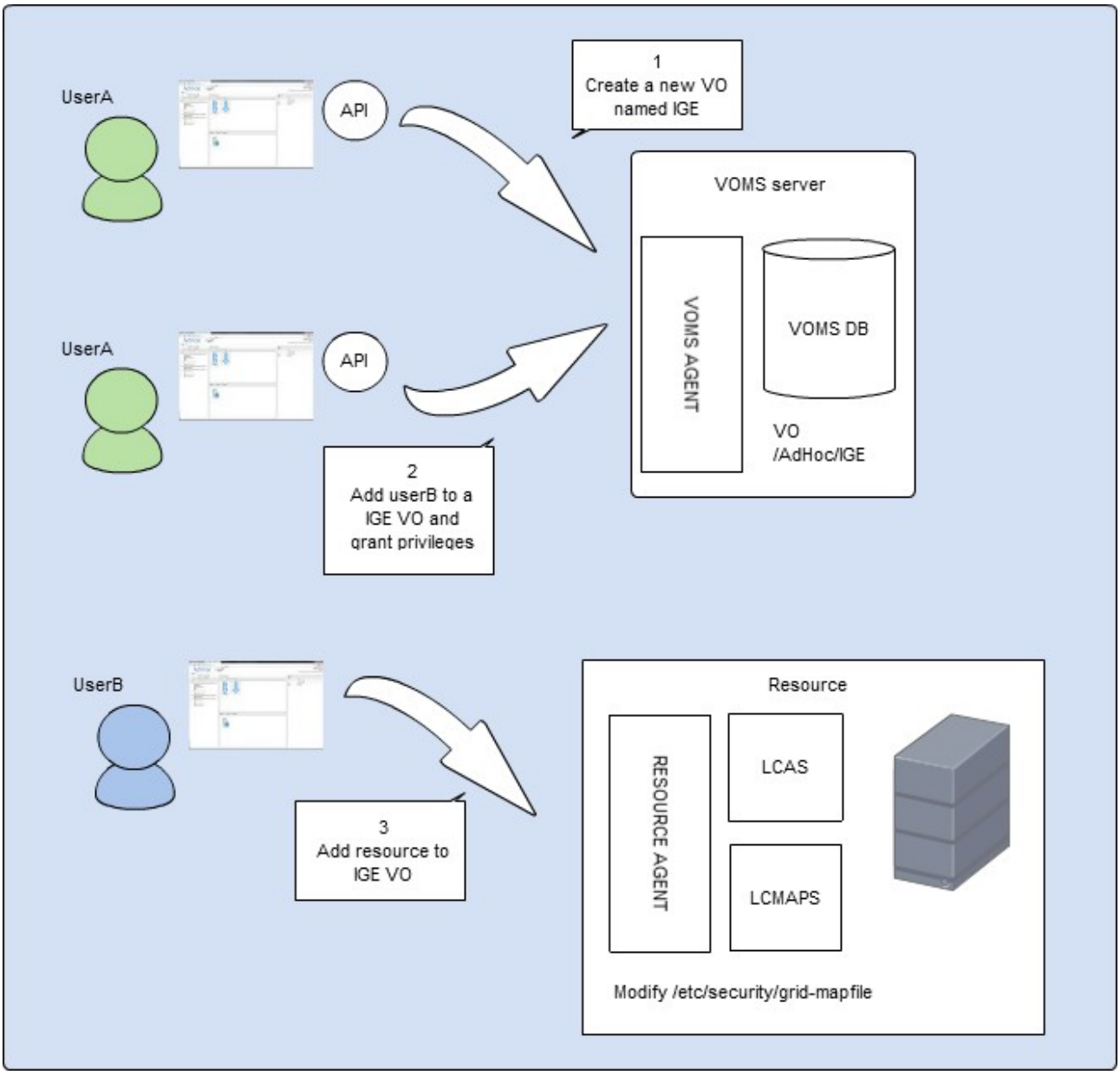

Diagram 9: AdHoc approach to sharing with VOMS and LCAS/LCMAPS

1. UserA logs into *AdHoc* and creates an IGE VO in the *AdHoc* GUI. Simultaneously, if she/he has not been yet registered with the predefined / AdHoc Virtual Organization, she/he is automatically added to this VO. The VOMS group in / AdHoc VO is created, this group is called the VO in AdHoc. All calls to the VOMS are made via the VOMS agent and API calls to the VOMS server services. Along with a group creation, UserA becomes an administrator of an IGE group, which grants her/him a possibility to grant privileges and add users to this group.
2. UserA adds UserB to a Virtual Organization through the *AdHoc* GUI using drag-and-drop functionality. In this step UserA as an administrator makes UserB a user of a Virtual Organization. This call is also done

using the VOMS agent and API calls to the VOMS server services.

3. UserB logs into *AdHoc* GUI and adds her/his resource to the IGE Virtual Organization. At this point, a call from *AdHoc* to a resource agent is made to modify /etc/grid-security/grid-mapfile mapping file the VO to resource affiliations are maintained in an internal *AdHoc* database. No call to VOMS server is made.
4. In the end, UserA only has to generate a proxy from a VOMS server and she/he can use UserB's resources.

6. Conclusion

The IT industry and Computer Science has evolved over the past ten years. For the scientific community used to drag-and-drop sharing in Dropbox, the Grid resource sharing model may appear unattractive and old-fashioned due to its lack of flexibility, command-line approach and inability to act without system administrator's assistance. If the Grid community is to gain traction and clientele, it needs to keep up to speed with the modern sharing approach.

AdHoc is the framework which follows the *Manifesto for the Secure Sharing*:

1. *AdHoc* enables easy sharing within seconds, without a system administrator's intervention,
2. *AdHoc* does not require IT skills, replacing command line with a simple GUI,
3. underneath, *AdHoc* uses commonplace security paradigms, including Shibboleth and PKI/GSI/VOMS, thus can easily be used in most grid infrastructures worldwide.

AdHoc still requires substantial work when it comes to tighter integration with several types of resources, and this will be our goal for the upcoming period. In general however, the two existing implementations described in this paper have proven that fast and easy sharing in “grid” environment is possible and practical.

Institutions and users interested in using *AdHoc* or introducing other form of cooperation are encouraged to contact GridwiseTech.

7. Credits

Here we wish to give huge credit to our partners, friends and sponsors, without whom this work would have never been possible. In the first place, we wish to thank to the members of the two consortia who helped to both fund and develop the project, and provided invaluable assistance in areas such as user feedback, testing, marketing, community reach and more.

AdHoc's integration with GSI and VOMS has been developed for the European Commission, and partially funded by Seventh Framework Programme (FP7) within the Initiative for Globus in Europe (IGE) in the years 2010 -2013. IGE is the consortium of the following institutions:

- Leibniz-Rechenzentrum, Bayerische Akademie der Wissenschaften in Germany (the Project Leader)
- University of Southampton in United Kingdom
- Technische Universität Dortmund in Germany
- Universitatea Tehnica Cluj-Napoca in Romania
- Universidad Complutense de Madrid in Spain
- Poznan Supercomputing and Networking Center in Poland
- Uppsala Universitet in Sweden
- University of Edinburgh – Edinburgh Parallel Computing Centre in United Kingdom
- Stichting voor Fundamenteel Onderzoek der Materie – Institute for Subatomic Physics in Netherlands
- GridwiseTech in Poland
- University of Chicago in United States of America

Adhoc is a continuation of a project carried out for ViroLab consortium which was funded through European Commission's Sixth Framework Programme (FP6-IST) in years 2006-2009, where we worked together with other established institutions:

- University of Amsterdam in Netherlands (the Project Leader)
- University of Stuttgart in Germany
- Erasmus MC - University Medical Centre Rotterdam in The Netherlands
- Catholic University of Rome in Italy
- IRSICAIXA Foundation in Spain
- University of Brescia in Italy
- Catholic University of Leuven in Belgium
- Eotvos Lorand University, ELTE in Hungary
- ACK Cyfronet in Poland
- University College London, Dep. of Chemistry (UCL) in the United Kingdom

Additionally, we want acknowledge the special effort of Tim Parkinson from University of Southampton who helped us to thoroughly review of this paper.

GridwiseTech

www.gridwisetech.com

contact@gridwisetech.com

Europe:
ul. Chrobrego 28/4
31-428 Kraków, Poland
Phone: +48 12 2947120

USA:
1133 Broadway, Suite 708
New York, NY 10010, USA
Tel: +1 646 7559884