

A VO-friendly, Community-based Authorization Framework

Part 1: Use Cases, Requirements, and Approach

Ray Plante and Bruce Loftis

NCSA

Version 0.2 (March 7, 2005)

Abstract

The era of massive surveys like LSST are driving the increasing necessity for astronomical research that is network-based and features remote data access and remote data analysis. With the development of the Virtual Observatory (VO) come the tools to make remote science easier. The VO community is large—thousands of potential users, and traditional authorization models based on individuals will simply not scale. In traditional models, authorization policies are enforced solely using permissions associated with login accounts; thus, a user or group must exist for every set of authorizations. This three-part document proposes an application of the Globus Community Authorization Service (CAS) to centralize authorization policies that must be enforced by a number of resources, minimizing the authorization intelligence needed by the resources. To ease the burden on users, we introduce the concept of *weak certificates* that enable a sufficient level of access control common in many existing web based applications today but which is compatible with stricter grid security practices. In part 1, we describe three general use cases that we aim to address, list requirements, and summarize our approach using the Globus CAS. In part 2, we describe how the CAS-based model can be applied to a single organization that manages many distributed services and users within a single administration domain. In part 3, we extend the model for use in VO applications that span across administration domains; in this model, VO users can establish a single login that can be used with any compliant portal or service.

1 Use Cases

1.1 PI-driven Observatory with a Proprietary Access Policy (NOAO)

The NOAO Science Archive manages access to a number of ground-based optical and infrared telescopes. These facilities are largely operated under a private investigator (PI) driven model. That is, a group of astronomers led by a principal investigator applies for time on a telescope to collect data for a particular scientific investigation. The observatory grants the investigators some number of nights on the telescope. Once the raw data are taken, the observatory grants the group exclusive access rights for a period

of 18 months; afterwards, the data are made publicly available. The observatory may automatically create processed data products from the raw data; these would generally have the same access rights associated with them.

The observatory typically delegates access authorization power to the PI; that is, the observatory provides the PI with exclusive access to the data, and the PI is responsible for sharing the data with his/her co-investigators, including additional researchers (e.g. students) that were not necessarily on the original proposal. In principle, however, all of the investigators that appeared on the original proposal have equal access rights to the data. The proposal information and process by which they are submitted and approved establish sufficient trust to grant use of the telescope.

NOAO serves data to hundreds of users around the world. Many who use NOAO facilities have participated in many (e.g., tens of) proposals over time, each with a different combination of collaborators.

1.2 A Survey-driven Observatory Generating Public Data (LSST)

The Large Synoptic Survey Telescope will generate very large datasets, totaling some 5 TB per night. In principle, all data are immediately made public; however, in practice, these data are held for a short period while integrity is verified. For raw data, this holding period will likely be short (a day or less) - just long enough to ensure proper tagging, packaging, and archiving. For higher level products that result from processing, the holding time may need to be longer, perhaps a week or more, depending on the processing involved.

In general, we envision three classes of authorized access. The first is public, non-restricted access. Next would be "curator" access which would allow observatory staff to review as yet unreleased data; this access would likely be largely read-only with some writing ability to annotate data or schedule reprocessing. Third, with the broadest permissions, would be "administrator" access, which would have the ability to remove data and manage access permissions by curators.

1.3 A Portal Supporting a Virtual Work Space (NVO)

Virtual Observatory (VO) applications will increasingly rely on the use of virtual work spaces to store new datasets and state generated by VO applications. An institution that supports a number of VO services (e.g. an observatory like NOAO) will create, for example, a virtual storage area where users can store their newly created datasets. In general, the user that generates the new data will have exclusive access rights to those data indefinitely; however, they will often wish to share them with their collaborators (much like the team that shares observational data described in section 1.2) or with the general public.

The International Virtual Observatory Alliance is developing the standards that allow VO services to interoperate. One such standard now in development, called VOStore, seeks to develop a common API for managing and using remote read/write storage. Interoperability between compliant VOStore sites will allow users to manage a virtual workspace that spans across many sites. In this environment, users will have a single

login for access anywhere in the space. They will be able to participate in many groups, each with authorization to different sets of data and services.

1.4 Commonalities: Groups and Virtual Work Spaces

The above scenarios share two important themes. First is that authorization for access to restricted resources is, in general, granted to groups of users. At some level, each scenario requires that users be added to (and deleted from) groups dynamically. Being able to manage authorization on the group level is much easier because there are typically fewer groups than users and because permissions assigned to groups change less often than permissions associated with users.

The other important theme is that of the virtual work space. Although this is highlighted in the VO example (1.3), it applies to the other examples as well. Very large collections like LSST can not rely effectively on the traditional modes of data distribution: users will not likely be able to download a nights worth of data to their local workstation for analysis. Instead, they will rely more heavily on archive services that filter and analyze data for them. This analysis will likely be complex enough to require that LSST run their own VOStore for its users. Furthermore, the LSST project is considering the supporting user-provided code; this has important implications for security management. Data created from such uploaded codes will have access restricted to the group that uploaded the code.

For an organization like NOAO, archive-based research will become increasingly important. They, too, plan to provide remote processing and analysis services to make such research easier. The variety of data (coming from the different facilities managed by NOAO) and the complexity of the processing will quickly necessitate managing their own VOStore.

2 Requirements

2.1 Logins/Sign-ons

2.1-R1: It must be as easy for a user to create an identity (i.e. login) for oneself as it is for any typical commercial or community web site featuring a personal workspace (e.g. Travelocity, community blogs). Once a user has filled out a registration form, he/she should be able to use that identity immediately (i.e. without delay for human approval).

Comment: First, we should remember that most astronomers are impatient with technology and will not invest a lot of effort into hoop-jumping that is not producing science results. Second, we should realize that for many applications that must deal with authorization, it is not critical that we guarantee that the person logging is indeed who they say they are; rather, we only need assurance that the person who is logged in is the one that created the account. This is certainly true of most VOStore applications currently being considered. In the case of proprietary access to telescope data, we need assurance that the person accessing the data is among the co-I's that submitted the proposal that generated

the data; this assurance can be backed by the observatory proposal process already in place.

It is this requirement, then, that underlies the need for "weak" certificates that are issued by authorities that should be by their nature less trustworthy than one that is more thorough in its verification of identity.

2.1-R2: It should be as easy to login in with an established identity as any common web site requiring a login.

2.1-R3: It should be simple for an automated software agent to establish its identity without requiring a human to supply a password. In particular, it should be possible to launch an authenticated agent automatically at boot-time, from a command scheduler (e.g. `cron` and `at` on UNIX machines), or from a long-running process.

2.2 Groups

2.2-R1: Authorization policies should fundamentally be assignable to logical groups.

2.2-R2: Individuals can be members of many groups at one time.

2.2-R3: Certain members of a group must be able to add and delete users from the group. Group membership will be highly dynamic.

2.2-R4: Groups may be permanent or expire after sometime.

2.2-R5: Applications and services should not have to worry about who is in what group at any given time.

2.2-R6: Any organization should be able to define and manage groups and assign their authorization policies.

2.3 Certificates

Comment: This section specifically assumes a authentication model based on certificates. See section 3 for more information on this model.

2.3-R1: A framework that allows for the creation of weak certificates must accept or otherwise be compatible with pre-existing, strong certificates.

2.4-R2: Users should be allowed to "upgrade" a weak certificate to a strong one without loss of access to their data.

3 A VO-friendly Model

3.1 Authorization Tools: CAS and Shibboleth

The model presented here is based on the [Globus](#) model for authorization management based on the Community Authorization Service [ref]. An alternative model is the [Shibboleth](#) framework. The latter approach has the advantage of being a system that is currently used by many access-restricted resources in academia. We currently believe

that the Globus model is better suited to the use cases described in section 1 for the following reasons:

- The Globus model is simpler and appears to have lower communication overhead for resolving authentication and authorization. It is likely to be easier to deploy and perform better.
- The Globus model appears more easily adapted for automated, non-interactive software agents. Typical deployments of the Shibboleth framework explicitly include interactive components (WAYF and login) into its use.
- Our use cases allow users to *create* resources with restricted access—e.g., virtual disk space—as well as new groups to access the resources; thus, personal identity, which is hidden in the Shibboleth framework, is important.
- The Globus model is more readily integrated with the grid. In particular, the Globus model and the use of MyProxy repositories provide a mechanism for credential delegation important for utilizing a chain of grid resources.
- The Shibboleth authentication model relies on a pre-existing federation of the users' sponsoring organizations. In the VO, not all users will necessarily have sponsoring organizations that can practically participate in such a federation (e.g. pre-college students, general public, students at foreign universities).

These assertions are *not* backed up by an in-depth, comparative study of Shibboleth; thus, it may well be straight-forward to create a Shibboleth-based infrastructure for the VO community. The Shibboleth and Globus approaches are very similar and contain components that share the same role; thus, a Shibboleth-based model may look very similar to the one described in this series of documents.

3.2 The User's View of Restricted Access

The requirements in section 1.1 recall the ease of logging into many portals in common use today. In our model, we aim to make logging into a VO-enabled portal indistinguishable from other common portals. That is, the first time the user uses a VO-enabled portal, they create a login by filling out a registration form; immediately upon submitting the form, the user is given a login, and he or she may begin using services with restricted access. Upon the next visit, user simply logs in with the username and password set during the registration process, and again the user can access restricted services.

The gained advantage of a VO-enabled portal is that once the user has created a login it can re-use that login at other VO-enabled portals. For example, if the user created a login via a portal run by NOAO, it can later use that same login to log into the portal at Space Telescope. Furthermore (and perhaps less obvious to the user), once the user is logged into a portal, the user can access any VO-compatible service with restricted access, even those beyond the administration realm of the portal. For example, a user that logs into the NOAO portal will be able to access his/her proprietary data from the Space Telescope archive without having to re-authenticate.

3.3 Weak and Strong Certificates

As discussed above in section 2 (see **Comment** for requirement 2.1-R1), there are a number of applications that require the use of restricted-access services that do not need the full security that would require users to present a traditional certificate signed by a well-known authority (e.g. Verisign, DOE, NCSA). The evidence for this is demonstrated by the large number of portals currently out there on the Web that allow users to “register” and create a login completely on-line and in an automated way. In these types of applications, it is not necessary that the portal ensure that the person that registers an account is actually who he says he is. Rather, it really only needs to ensure that the person accessing some personal space or state is the person who created the account. To make it easy to support these types of applications, we introduce a distinction between “strong” and “weak” certificates.

A *weak certificate* is one that does not attempt to ensure the owner of the certificate is the person identified in its distinguished name. A weak certificate, therefore, can be generated quickly and automatically without a human in the loop. A *strong certificate*, on the other hand, ensures that the owner of the certificate *is* the person identified by the distinguished name. The traditional methods for ensuring this, in which an applicant must present additional proof of identity, would be used. Typically, a human is in the approval loop, and, thus, there would be some latency between registration and the issuing of the certificate that would prevent its immediate use. Strong certificates would be used for accessing resources that require more careful allocation and accountability, such as those that offer computational services.

One could imagine a model in which, instead of using weak certificates, one would simply use traditional user tracking mechanisms supported by current web servers or content management systems. However, the weak certificate model has some important advantages for VO applications:

- A single authorization management system can be used for both high-security services (requiring strong certificates) and lower-security services.
- Certificates provide a mechanism for securely crossing administration domains (the focus of Part III of this document).
- Certificates provide “authenticated currency” that allows to a portal client to break away from the portal to access restricted services directly without re-authenticating.

3.4 Overview of Model components

In the next two parts, we describe two variations of our authorization model. Both, of course, share common components.

- *Separate Certificate Authorities (CAs) for creating strong and weak certificates.* The key difference between these two types of certificates is the level of trust. Applications typically make the choice as to whether to trust a certificate based on the distinguished name of the signing CA. Thus, applications that require strong certificates would choose not to trust the weak CA. We recommend that the word

“Weak” be included in the name of a weak CA to make more obvious to users its limitations.

- *A registration service* for creating a login to a portal.
- *A login service* that a user employs to log into a secure environment, be it a portal or a web of restricted-access resources distributed across the VO.
- *A MyProxy service* to hold copies of certificates in order to create proxy certificates for use by certificate owner. For most weak certificate users, this will be the sole location of their certificates.
- *A Community Authorization Server (CAS)* that issues community credentials with personal authorization policies embedded inside.
- *CAS-enabled services*: data services that can accept and use CAS credentials to restricted access.
- *Credential-enabled client applications*: in some scenarios, client applications can be helpful for interacting with CAS-enabled portals; it may be necessary for some applications to be able to download and use CAS credentials for direct client-side access to data services.

4 Applications of the Model

The next two parts of this three-part document describes how a CAS-based authorization model would be applied to two scenarios. In part 2, we address how a single organization can manage a number of distributed services within a single administration domain. We take as an example the case of NOAO as described in section 1.1. In this example, we show how by centralizing authorization policy through an organization-wide CAS server, we remove the amount of authorization intelligence required to be embedded into service implementations as well as eliminate the problem of propagating this information uniformly to services that need it. This application does not attempt to provide any interoperability with other portals; rather, it assumes its management of authentication and authorization is completely independent of any other portal. This application can be thought of as a transition toward the more general application described in the part 3.

Part 3 broadens the application to that of the VO, as described in section 1.3. In this model, authentication is handled by a shared community entity like the NVO. However, the management of authorization policies is still handled by each individual portal or sub-community (like NOAO). In this model, it becomes possible for portals to reach across administration domains to access restricted services. In particular, it makes it possible to build a VOSpace in which a user can manage multiple VOStores provided by multiple portals across the VO.