Enabling the Provisioning and Management of a Federated Grid Trust Fabric

Stephen Langella[1], Scott Oster[1], Shannon Hastings[1], Frank Siebenlist[2], Tahsin Kurc[1], Joel Saltz[1]

[1]*Department of Biomedical Informatics*
Ohio State University
Columbus, OH 43210
{langella,oster,hastings,kurc,jsaltz}@bmi.osu.edu

[2]*Mathematics and Computer Science Division*
Argonne National Laboratory
Argonne, IL 60439
franks@mcs.anl.gov

Abstract

In order to authenticate and authorize users and other peer-services, Grid services need to maintain a list of authorities that they trust as a source for issuing credentials. Grids inherently span multiple institutional administration domains and aim to support the sharing of applications, data, and computational resources in a collaborative environment. In this environment there may exist hundreds of certificate authorities, each issuing hundreds if not thousands of certificates. In such a dynamic multi-institutional environment with tens of thousands of users, credentials will be issued and revoked frequently, and new authorities will be added regularly. Clearly a Grid-wide mechanism is needed for maintaining and provisioning trusted certificate authorities, such that Grid services and users may make authentication and authorizations decisions against the most up-to-date trust information. In this paper we present the design and implementation of the Grid Trust Service (GTS), a federated framework for creating and managing a Grid trust fabric, enabling the provisioning of certificate authority information.

1. Introduction

As Grid computing technologies gain acceptance and adoption, the transition from highly specialized Grids with only a few institutional participants to a Grid environment with hundreds of institutions is becoming a reality. Security is of primary importance in the Grid and the support for secure communication, authentication, and authorization is a critical requirement, specifically in settings where sensitive data (e.g., patient medical information) must be accessed and exchanged. Also needed are mechanisms to establish and manage "trust" in the Grid so that asserted identities and privileges can be verified and validated with the required level of confidence. Within a collaboration, it is clear that the different institutions will have tiered levels of confidence in the users and service management policies of the various other institutions. While generally all institutions want to collaborate in some fashion, they will have services with varying security policy enforcement requirements. The interconnections, between clients and services that are able to securely communicate in the larger Grid, form conceptual overlays of trust, which we herein refer to as the "trust fabric" of the Grid. *Figure 1* shows an example trust fabric composed of four trust groups (Trust Groups A-D), over a worldwide Grid. The establishment, provisioning, and management of the trust fabric are critical to the scalability, maintenance and security of the Grid and other web service environments. This paper is concerned with the design and implementation of the Grid Trust Service (GTS) framework to facilitate the provisioning and management of a Grid trust fabric.

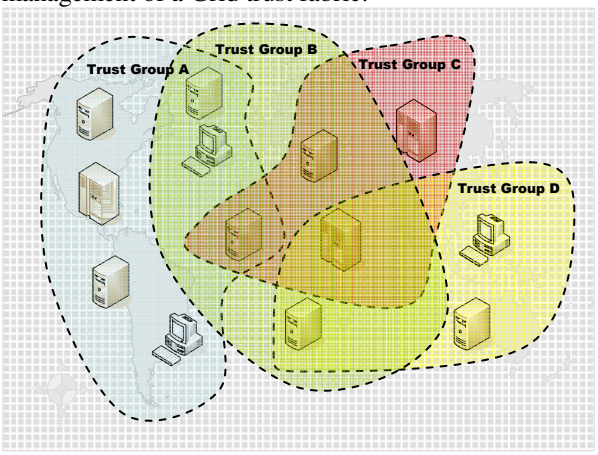

Figure 1 Conceptual Trust Fabric example with different trust groups

Many components of the Grid rely on having trust agreements in place. For example, when a user wants to access a service, she is authenticated based on an identity assigned to her. In the Grid, clients and services authenticate with one another using X.509 identity certificates. Grid Identities are assigned to users by authorities. When a grid-identity is asserted by an authority in the form of an X.509 identity certificate, it is digitally signed by that authority. One of the criteria relying parties use in making authentication decisions is determining whether or not the certificate presented is signed by a trusted certificate authority.

Thus, authentication requires a trust agreement between the consumers of X.509 identity certificates and the certificate authorities that issue them.

In a Grid environment, there may exist tens or even hundreds of certificate authorities, each issuing hundreds if not thousands of certificates. To make matters more complex, in a dynamic multi-institutional environment, the status of identities may be updated frequently. Identities and credentials can be revoked; suspended, reinstated, or new identities can be created. In addition, the list of trusted authorities may change. In such settings, certificate authorities will frequently publish Certificate Revocation Lists (CRL), which specify "black listed" certificates that the authority once issued but no longer accredits. For the security and integrity of the Grid, it is critical to be able to perform authentication and validate a given identity against the most up-to-date information about the list of trusted certificate authorities and their corresponding CRLs.

Each institution normally manages its own security infrastructure with its own CAs, and all client and services within such an administrative domain needs to be configured to trust the local trust roots. If collaborations span administrative domains, then participating entities have to be configured to trust the trust roots defined in the different organizations within the limits of their own local policies. The required trust root configurations to participate in such Virtual Organizations (VO) are complex, error prone and security policy sensitive. By centralizing the configuration management and provisioning collaborating clients and services "on demand", one can ensure that the correct and up-to-date trust-root information is made available. In this scenario, the central provisioning server becomes a trusted entity itself, and clients need to be configured to trust its provisioning information. In order to facilitate the trust in the provisioning servers, they should be locally known to the clients, which requires local provision servers to aggregate and to front-end remote ones.

The Grid Trust Service (GTS) is a Web Services Resource Framework (WSRF) [9] compliant federated infrastructure enabling the provisioning and management of a grid trust fabric. The salient features of the GTS can be summarized as follows:

- It provides a complete Grid enabled federated solution for registering and managing certificate authority certificates and CRLs, facilitating the enforcement of the most recent trust agreements.
- It allows the definition and management of trust levels, such that certificate authorities may be grouped and discovered by the level of trust that is acceptable to the consumer.
- The federated nature of the GTS, coupled with its ability to create and manage arbitrary arrangements of authorities into trust levels, allows it to facilitate the curation of numerous independent trust overlays across the same physical Grid.
- The GTS can also perform validation for a client, allowing a client to submit a certificate and trust requirements in exchange for a validation decision, which allows for a centralized certificate verification and validation.

2. Motivation

Our work is driven mainly by the requirements that are derived from the Grid security usage scenarios gathered from the Cancer Biomedical Informatics Grid (caBIG™) [1] program. This program, funded by the National Cancer Institute (NCI), was launched to provide a coordinated approach to the informatics requirements of basic and clinical cancer research and multi-institutional studies. The goal is to accelerate the delivery of innovative approaches for the prevention and treatment of cancer by facilitating sharing, discovery, and integration of distributed information and analytic resources. Although the caBIG effort started relatively recently, it is expected that the caBIG community will grow to comprise hundreds of organizations and many thousands of cancer-research participants from geographically dispersed medical centers, universities, government agencies, and commercial companies.

Given the sensitivity of the medically related data and the number of institutions involved, security has quickly become a high priority issue in caBIG. In order to articulate the security requirements of caBIG and evaluate existing technologies, a Security Technology Evaluation White Paper [8] has been developed. This white paper serves as one of the motivating influences for the GTS work. A key security issue is to implement an effective mechanism of managing the Grid-wide identities and privileges of large numbers of users across multiple organizations. Another key issue is to be able to authenticate and authorize users to caBIG services based on this information, such that access to resources can be restricted to individual users or a group of users. Considering the scale of caBIG, these issues become challenging.

The GTS framework has been implemented in the context of the caGrid [2, 30], which is the Grid software infrastructure of caBIG. caGrid is a service-oriented architecture and implementation that provides core services, toolkits and wizards for the development and deployment of community provided services. One of the primary design principles of caGrid is to leverage the open Grid standards [3, 4]. The caGrid infrastructure is built on top of the Globus Toolkit [5], the most widely used reference implementation of the Grid standards. The Globus Toolkit implements support for security via its Grid Security Infrastructure (GSI) [6, 7]. GSI requires the use of X.509 Identity Certificates for identifying a user. An X.509 certificate with its corresponding private key constitutes a unique credential or so-called "Grid credential" that is used to authenticate both users and services within the Grid. Under the current Globus release (4.0.3), the authentication process ensures that the X.509 Identity Certificate provided by the peer was issued by a trusted certificate authority. However, one limiting issue with the current mechanisms is that trusted certificate authorities (CAs) and their CRLs are maintained locally on the file system of each Globus installation. When a client authenticates with a service, Globus locates the root CA and CRL of the client's Identity Certificate on the local file system. Once located, the Globus runtime validates the Identity Certificate against the CA certificate and CRLs. Although this approach is effective, it is difficult to provision CA certificates and CRLs in a large multi-institutional environment, as one has to ensure that all CA and CRL information must be copied to every installation and kept current with the dynamically changing environment. Given the sensitivity of the data in caBIG, it is critical that services authenticate clients against the most up-to-date list of CA certificates and CRLs. This requirement is one of the primary motivations behind the design and development of the Grid Trust Service (GTS).

3. Background and Trust Fabric Profiles

The XML Key Management Specification (XKMS) [10] specifies models and protocols for distributing and managing public key credentials. XKMS specifies a tiered service model allowing applications to leverage the level of functionality that meets their requirements. The design of the GTS mimics this tiered service model, and for the purpose of this paper we will describe the relevant tiers in terms of the locations of CA certificates and where certificate validation occurs.

Tier 0, also referred to here as the *Locally Stored, Locally Validated* (LSLV) profile, specifies that CA certificates are stored and accessed locally; certificate validation is done against the locally stored CA certificates. Although the deployment and maintenance of the LSLV profile seems simple, the realization of such a profile in a large and dynamic Grid environment faces several problems. First, the LSLV profile makes it difficult to provision CA certificates and their corresponding CRLs. Under the LSLV approach, every local environment is required to update its local trust store each time a new CA certificate becomes available or each time a CA publishes a new CRL. Moreover, the LSLV profile could pose a potentially serious security risk because it requires users to maintain their own trust fabric. A simple error by an inexperienced user could easily introduce a security hole in the environment.

Tier 1, also referred to here as the *Remotely Retrieved, Locally Validated* (RRLV) profile, specifies that CA certificates are retrieved remotely from a Trust Service; certificate validation is done locally against the remotely retrieved CA certificates. When deployed in a large Grid environment, the RRLV profile significantly improves upon the LSLV profile. Retrieving the latest CA certificates and corresponding CRLs remotely allows Grid services to perform validation against the latest trust fabric. It also moves the management of the trust fabric from the hands of users to the hands of administrators, who have more expertise and experience in managing trust. Performing the validation locally also allows services and applications to enforce local validation policies that may go beyond those specified in X.509 certificate validation. At the same time local validation can be a potential security risk, if the local validation process is not effectively enforced.

Tier 2, also referred to as the *Remotely Stored, Remotely Validated* (RSRV) profile, specifies that CA certificates are stored remotely; all certificate validation with exception to validating the certificate of the remote validation service is done remotely against the remotely stored CA certificates. The RRLV profile and the RSRV profile are similar in that validation is done against the latest trust fabric. They differ in where the validation is done. Under the RSRV profile validation is done remotely, which removes the validation from the hands of the local providers minimizing potential security exposure when validation is not strongly enforced locally. However, it introduces a potential performance problem in that a local service

is required to contact a remote validation service every time validation is required.

4. Grid Trust Service (GTS)

The Grid Trust Service (GTS) is a WSRF compliant Grid Service framework for creating, managing, and provisioning of a federated Grid trust fabric. Establishing trust in the Grid is rooted in the problem of determining whether or not to trust a given certificate authority. Through its service interface, the GTS provides the ability to register and manage certificate authorities. Using the GTS, Grid entities (services and clients) can discover the certificate authorities in the environment, decide whether or not to trust a certificate authority, and determine the levels of trust assigned to a certificate authority.

In implementing a trust fabric in a Grid environment, we envision that the trust fabric will consist of Grid users and administrators, Grid services, multiple CAs, and multiple GTS instances. The flexibility of GTS allows many possible deployment scenarios. For example, an institution can set up a local CA and GTS instance. Alternately, a group of organizations may all share a common CA for certificates and a GTS to maintain the list of trusted external CAs. In any deployment, each Grid user will be given a certificate, signed by a CA that can be used by services to authenticate the user. Similarly, each Grid service will be given a certificate, also signed by a CA, so that a client application, user, or other service can check the integrity of the service. Since GTS instances are Grid Services, they should be assigned certificates as well.

As deployments leveraging the GTS to maintain the trust fabric are effectively delegating this responsibility to the GTS, it is imperative the GTS instance(s) can be trusted. Traditionally a trust "bootstrapping" approach is adopted wherein clients and services communicating with the GTS are manually configured to trust its CA. Additionally, by default, the GTS clients perform host authorization against the specific GTS with which they are communicating. This ensures the service providing the information about the trust fabric (the GTS) has a certificate signed by a trust authority, and that it is provably the specific instance the client intended to communicate with. There are multiple possible deployment options for assigning certificates to GTS instances. A possible way is that each GTS instance has a self-signed certificate (i.e., serving as its own CA). In such a deployment, clients and services are manually configured to trust the self-signed certificates of the GTS instances they intend to interact with. Alternatively, there can be one (or a few) *trusted root-CA*, which will be used to assign the certificates to each GTS instance. Installations in the Grid are then bootstrapped to trust this authority or small set of authorities. Note that even if the clients are pre-configured with the trusted CAs, the GTS infrastructure can be used as a distribution mechanism of the CA's CRLs.

An advantage of the deployment with self-signed GTS certificates is that it does not require a root-CA to exist. If clients and services will only interact with a few GTS instance, this could be a preferred way of deployment. However, GTS certificates cannot be revoked in such a deployment. An advantage of the deployment with root-CA is that if the root-CA monitors integrity of GTSs, it can revoke GTS certificates and publish CRLs, which will include the revoked GTS certificates, in case of a security breach (with the acknowledged issue of communicating the revoked GTS-certificate to the GTS-clients when the distribution mechanism relies on the GTS).

While GTS facilitates management of certificate authority lists, the trust establishment with a CA and setting its trust level is a manual process. That is, the administrator of a GTS instance is expected to exchange correspondence with the owner of the CA to be added to the list of CAs managed by the GTS instance. Once a trust level, or set of trust levels, has been established, the CA can be added to the list of CAs so that it can be discovered by users and services. The trust levels of the CA can be used by a client or service to determine whether or not to trust the CA. (We will describe the management of trust levels in greater detail in Section 4.2.) In the trust fabric setting, a CA is responsible for publishing its CRL so that local CRLs and CRLs maintained by GTS instances can be updated. In addition to the level of trust, each GTS maintains a status value for each CA in its list of CAs. By changing the status of a CA (e.g. from "trusted" to "suspended"), the GTS can report to a relying parties that any certificates from the CA in question should not be trusted at this time, this can be valuable if there is a security breach.

In a large Grid environment, it is desirable to have a federated trust fabric for redundancy and scalability, and for the integration of multiple trust overlays. A possible way of federating GTS instances is to create a hierarchical structure, in which there are *authority GTS instances* and *subordinate GTS instances.* The

authority GTS instances maintain lists of trusted CAs and CRLs and synchronize with CAs for updates. The subordinate GTS instances can be designed to synchronize with one or more authority GTS instances. In this way, when the state of the trust fabric changes (e.g., because of publishing a new CRL), the updates need not be broadcast to all GTS instances individually. The approach for federation of GTS instances is discussed in Section 4.5.

4.1. Profiles Supported by GTS

Out of the box, the Globus Toolkit supports the LSLV profile, and the GTS adds the support for both the RSRV profile and the RRLV profile. Our use cases for enabling trust between identity providers and consumers of identities and between attribute authorities and consumers of attributes call for the use of Remotely Stored Remotely Validated (RSRV) profile. It is also anticipated that future releases of Globus will support callouts to remote validation services. The GTS supports this profile by providing a validation operation through its service interface. In this manner, the GTS is a validation service allowing clients to submit validation criteria and an X.509 certificate chain for validation.

Figure 2 illustrates an example usage scenario of the RSRV profile. In this example, a GTS instance is used to manage and validate client certificates in caGrid. This example has a Dorian [11] instance serving as a CA and proxy certificate generator. Dorian is a Grid service infrastructure for the management of Grid user accounts. Dorian provides an integration point between external security domains and the Grid security domain, enabling users to obtain Grid credentials using their locally provided authentication mechanism. In *Figure 2*, an Ohio State University (OSU) user authenticates to the OSU authentication system using her local user name and password. Upon successfully authenticating, the user is given a signed SAML assertion, which can be given to Dorian in exchange for a Grid proxy or Grid credentials. In the example, a trust agreement has been established between the GTS instance and the Dorian instance wherein the Dorian instance's CA is listed as a trusted authority in the GTS instance. Trust and the trust level(s) between the GTS and Dorian instances can be established by the administrator of Dorian and the administrator of GTS through the exchange of information such as certificate policy and certification process statements. When the user presents the Grid proxy certificate to a Grid service, the grid service first performs a check to ensure that the user is the holder of the private key associated with the proxy certificate, the proxy certificate is then sent to the GTS instance for validation (Step 6: "Is Proxy Trusted?" in *Figure 2*). The GTS instance can respond back to the service with a "yes" or "no" answer. If the response is "no", the service prevents the user from accessing its resources. If the response is "yes", the service can take additional steps to authorize the user and control her access to the service's functionality based on the authenticated user's privileges.

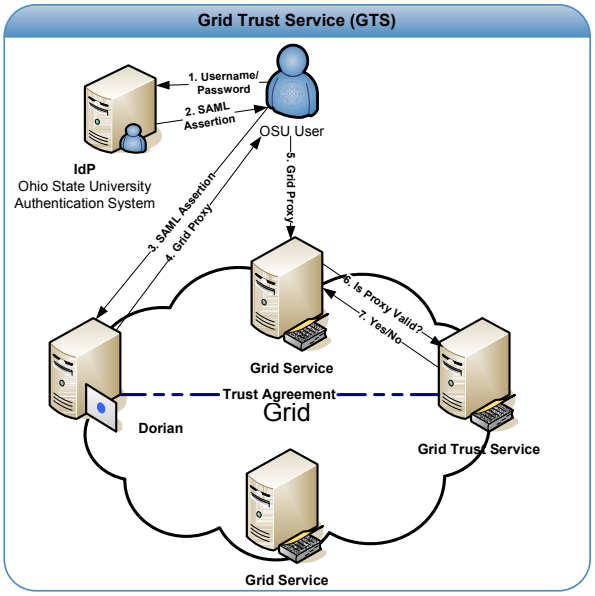

Figure 2 Trust Service (GTS) RSRV Profile

The GTS also supports the RRLV profile for accessing the trust fabric. This is because the current release of Globus Toolkit (Globus 4.0.3) does not provide a means of supporting remote validation of credentials; rather validation must be done against a locally maintained trust fabric. To enable Globus to authenticate users by validating their credentials against the trust fabric maintained in the GTS, the RRLV profile is employed. The GTS provides a framework called SyncGTS, which is embedded in the Globus runtime to automatically synchronize the local trust certificate store with the latest trust fabric maintained in the GTS. The SyncGTS functionality is presented in more detail in Section 5.

Figure 3 illustrates how authentication and certificate validation can be performed with the RRLV profile in GTS. When a Grid service is invoked, Globus authenticates the client by validating that the Grid proxy provided is signed by a trusted certificate

authority. The certificate is validated against the local trusted certificates directory as is seen in the figure. In the example in *Figure 3*, the Dorian certificate authority has been registered with the GTS as a trusted certificate authority and Globus has been configured to synchronize its local trusted certificate store with the GTS. Thus when the OSU user invokes a Grid service using her Dorian-obtained proxy, she will be successfully authenticated by Globus. Future versions of Globus will support a credential-validation callout, at which time the RSRV GTS profile, illustrated in *Figure 2* can be employed.

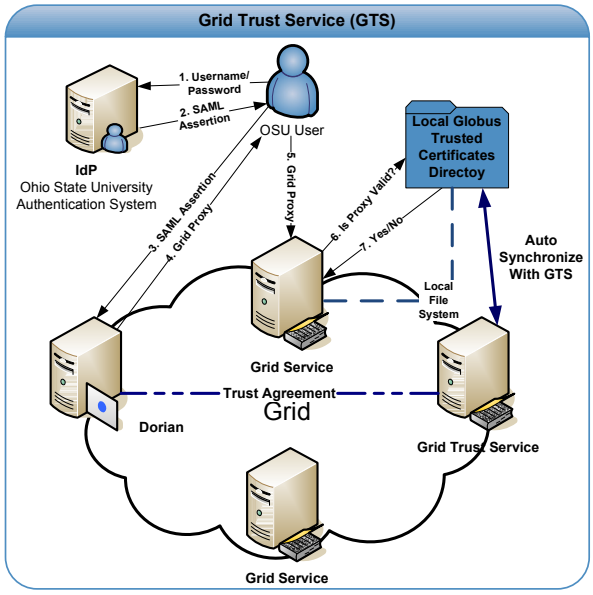

Figure 3 Grid Trust Service (GTS) RRLV Profile

4.2. Managing Trust Levels in GTS

The trust level specifies the level of confidence with which a given certificate authority is trusted in the fabric in which it is deployed. In the Grid, one can assume that certificate authorities will be trusted with different levels of confidence.[1] There will be multiple types and instances of certificate authorities. Some authorities may be used to assert identities; other authorities may be used to assert digitally signed documents. Even certificate authorities asserting the same thing may have differing levels of trust associated with them, as they may employ different policies for issuing and validating identities. For example, a certificate authority may require that anyone applying for a certificate present official documentation about their real identity. The CA issues a certificate to the applicant after these documents are reviewed by the CA staff. Another certificate authority may automatically issue certificates based on an online application submitted by the applicant -- the applicant may have been requested to log on to the system using a user id and password. In these cases, the first certificate authority has a stricter policy for issuing certificates; thus, it is reasonable to expect that the first certificate authority should be trusted more than the second certificate authority . More specific examples of trust levels are CAs that are accredited by the International Grid Trust Federation (IGTF) [31] or CAs whom use a Hardware Security Module (HSM) for storing their private key.

In order to model different levels of trust in the trust fabric, the GTS provides a mechanism for its administrators to define and manage trust levels. When certificate authorities are registered into the trust fabric they are assigned one or more trust levels. Clients can specify the level of trust that they require when discovering trusted CAs or when requesting validation. Trust levels in the GTS each consist of a unique name (value) and description. The unique name is used to implicitly bind a certificate authority to a trust level. The description is used as a human readable method of understanding what a specific trust level represents. Through its Grid service interface, the GTS provides several operations for accessing and administrating trust levels. The *getTrustLevels()* operation is a publicly accessible operation which provides a list of all the trust levels existing in a GTS. The *addTrustLevel(), updateTrustLevel(),* and *removeTrustLevel()* operations are accessible to GTS administrators, providing them with a method of administering trust levels. *Figure 4* illustrates the trust level management interface in the GTS administrative user interface (admin UI).

On initial deployment, the GTS does not come pre-configured with a set of trust levels, at the time of development it was determined that it was desirable to allow Grid administrators to define trust levels based on the security policy defined by their Grid. The GTS provides the tools for administrators to assign and manage trust levels to CAs. The system is flexible in that each GTS instance can have its own trust levels. It

[1] The trust level concept in the Grid is similar to obtaining an identity card from different institutions. Obtaining a passport will require that the application provide more documentation and a more thorough background check be performed. On the other hand, getting a library card will require a less strict background check and less documentation

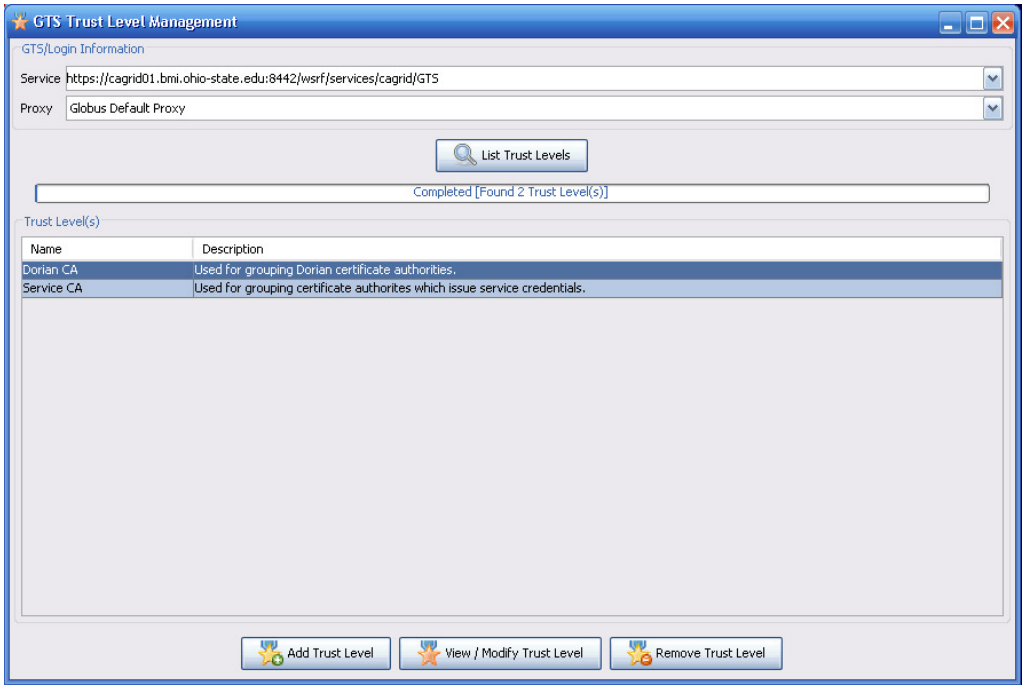

Figure 4 Admin UI for trust level management

may be desirable to have a common set of trust levels across multiple institutions, however, so that a client or a service can interpret the trust level associated with a CA correctly. In that case, a standard for trust levels should be accepted by the community and made available in a common repository or distributed to each GTS instance. Note that having a common set of trust levels does not require each GTS instance to assign the same trust level to the same CA. A GTS instance may assign a higher level of trust to a CA than another GTS instance. However, a common set of trust levels allows a client to interpret the trust level assigned by each GTS instance correctly and determine whether or not to trust the corresponding CA.

4.3. Managing Certificate Authorities

The GTS service interface provides several operations for registering and managing trusted certificate authorities. Registration of a certificate authority requires the specification of the CA's root certificate, a set of trust levels, a status, and an optional CRL. The CA's root certificate is required for validating certificates. The set of trust levels specifies the level of trust associated with the CA. The status specifies the current state of the certificate authority; the status can be set to "trusted" or "suspended". Setting the status of a certificate authority allows it to be temporarily added and removed from the trust fabric. For instance, if the security of a CA has been compromised, its status can be set to "suspended" to quickly inform the relying parties not to trust certificates issued and signed by the CA. For each trusted certificate authority, the GTS maintains a Certificate Revocation List (CRL). The CRL contains a list of certificates that have been revoked by the CA.

The GTS makes several operations available to administrators for managing trusted certificate authorities, these include *addTrustedAuthority()*, *updateTrustedAuthority()*, *removeTrustedAuthority()*, and *updateCRL()*. The *updateCRL()* operation provides a mechanism for CAs to publish their CRL immediately after it is locally updated. For example, Dorian revokes a user's certificate and adds an entry to its CRL, every time a user's account is suspended. Upon updating its CRL, Dorian immediate publishes it to the GTS. The *updateCRL()* operation can be invoked by GTS administrators and individual trusted certificate authority administrators. The GTS provides a mechanism of assigning and maintaining a list of individual trusted certificate authority administrators. *Figure 5* illustrates the trusted certificate authority management interface in the GTS admin UI.

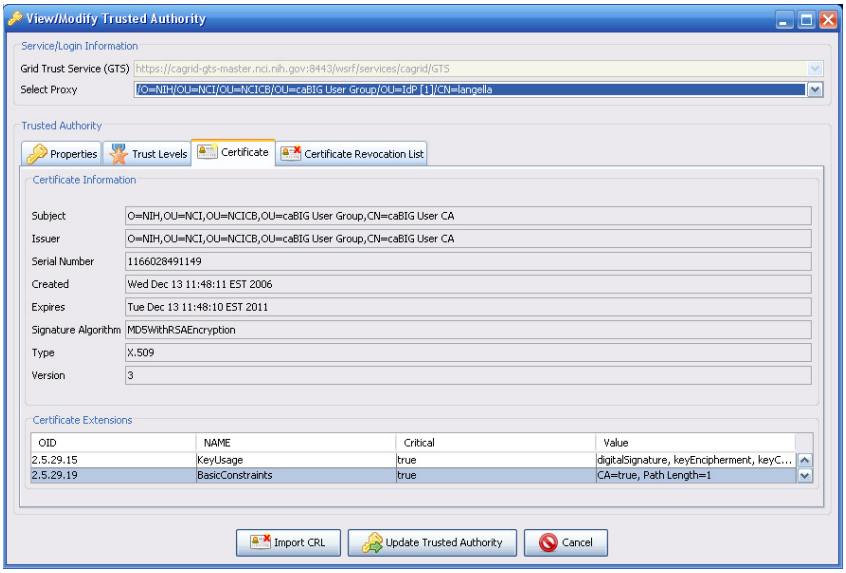

Figure 5 Admin UI for managing a trusted CA

4.4. Managing Administrators

Many of the operations provided by the GTS provide a means of administrating the trust fabric and are therefore restricted to GTS administrators. The GTS provides three operations for managing the assignment of administrative roles to users: *addPermission(), revokePermission(),* and *findPermissions().* These operations are, themselves, obviously restricted to GTS administrators.

The GTS allows for the assignment of two types of permissions; GTS Administrators, and Trusted CA Administrators. GTS Administrators are "super users" and can perform any operation on a GTS (i.e. manage certificate authorities, manage trust levels, manage permissions, etc). A Trusted CA Administrator permission corresponds to a specific CA, giving a user the ability to update the CRL for the corresponding CA.

4.5. Federation

Redundancy and scalability are critical properties of a federated trust fabric. Serious performance implications will occur if all entities in the Grid are discovering and performing validation against a trust fabric maintained in a central GTS. In order to enable a federated trust fabric, each GTS can be administered to synchronize with a set of authoritative GTSs. GTSs can inherit both trust levels and trusted certificate authorities from its authority GTSs. Registering an authority GTS requires the specification of the following properties: service's uniform resource identifier (URI), priority, whether or not to synchronize the trust levels, time to live, whether or not to perform authorization, and the authority service's identity. The priority property is used for resolving conflicts between authority GTSs, for example if two authority GTSs have a listing for the same certificate authority, the authority GTS with the highest priority will be used for obtaining that certificate authority, and its corresponding information (e.g. its CRL). If contact to an authoritative GTS is lost for a significant amount of time, the trust fabric within the subordinate GTS may become significantly out of date; this could be a potential security risk. The time to live property specifies how long certificate authorities obtained from authoritative GTSs will be valid for in the subordinate GTS. The time to live on a given certificate authority record is reset after each synchronization with the authority GTS. If contact with an authority GTS is lost, the time to live will expire and the certificate authority will be removed from the subordinate's trust fabric.

Figure 6 illustrates an example of how multiple GTSs can be deployed to create and manage a federated trust fabric. In the example there are five GTSs: caGrid GTS, TeraGrid GTS, OSU GTS, caGrid/TeraGrid GTS, and UT GTS. The caGrid GTS has no authority GTSs, it manages the certificate authorities A and S. The TeraGrid GTS has no authority GTSs, and it manages the certificate authorities X and S. The OSU GTS has one authority GTS, the caGrid GTS. The OSU GTS inherits the certificate authorities A and S from its authority the caGrid GTS. The OSU GTS

manages an additional certificate authority B. The OSU GTS is an example of how the global trust fabric can be extended to include local trusted certificate authorities, in this case and the additional certificate authority CA B, which is trusted by OSU. The caGrid/TeraGrid GTS has two authority GTSs, the caGrid GTS and the TeraGrid GTS. The TeraGrid GTS inherits CA A from the caGrid GTS and CA X from the TeraGrid GTS, since the caGrid GTS has a higher priority then the TeraGrid GTS, it inherits CA S from the caGrid GTS. The caGrid/TeraGrid GTS is an example of how two existing trust fabrics from two different Grids can be joined together. Finally the UT GTS has one authority GTS, the TeraGrid GTS. The UT GTS inherits CA X and CA S from the TeraGrid GTS. The UT GTS is an example of standing up a GTS for better redundancy and scalability.

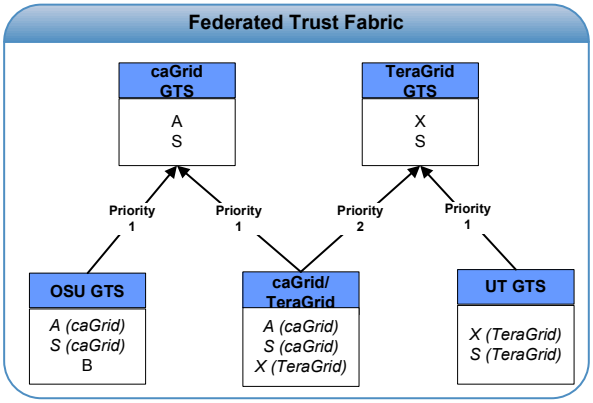

Figure 6 Example federated GTS deployment

Supporting a federated trust fabric across GTSs introduces additional metadata to be associated with trust levels and certificate authorities. This metadata includes the "Source GTS", "Authority GTS", and "Time to Live". The "Source GTS" specifies the service URI of the GTS in which the trust level or certificate authority was inherited from. The "Authority GTS" specifies the service URI of the GTS that is the authority of trust level or certificate authority. "Time to Live" specifies the date until which the certificate authority entry is valid.

Through its service interface, the GTS provides administrative operations for managing the federation. The trust fabric is managed by specifying authority GTSs for a given GTS. To support this, the GTS service interface provides the following operations: *addAuthority()*, *updateAuthority()*, *removeAuthority()*, and *updateAuthorityPriorities()*. *Figure 7* illustrates the authority GTS management interface in the GTS admin UI.

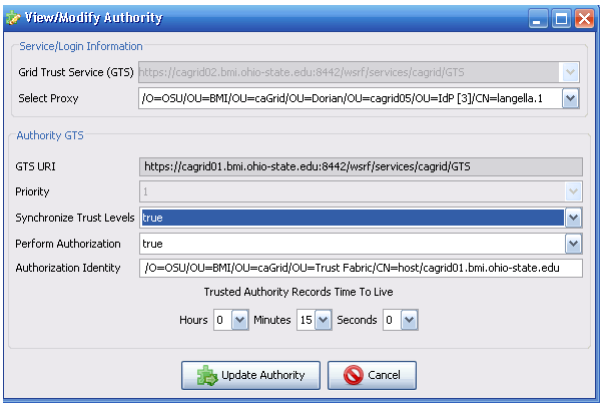

Figure 7 Admin UI for GTS Authority Management

4.6. Discovery and Remote Validation

In order to support the RRLV profile, local validation processes will need a method of discovering and obtaining trusted CAs from the pre-configured GTS. The GTS provides a publicly available operation, *findTrustedAuthorities()* through its service interface. In discovering trusted certificate authorities the GTS allows the specification of search criteria. The search criteria include the name of the certificate authority, the trust level, the status, the lifetime, the source GTS, and the authority GTS. Using this operation, validators implementing the RRLV profile can discover a list of trusted certificate authorities based on the trust level. Additionally, trust fabric administrators may leverage this operation for discovering the trust fabric such that they may administer it. *Figure 8* illustrates the discovery interface in the GTS admin UI.

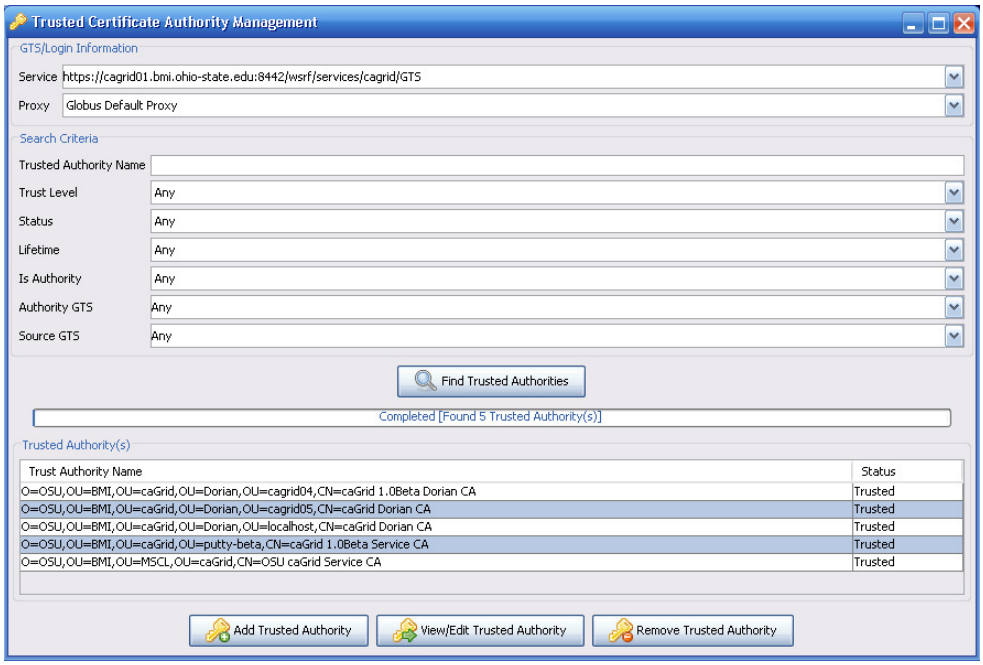

Figure 8 Admin UI for Trust Fabric Discovery

To support the Remotely Stored, Remotely Validated (RSRV) model, the GTS takes on the role of a validation service. The GTS *validate()* operation enables clients to submit, for validation, a certificate chain and validation criteria. The GTS uses the X.509 validation specifications for validating the certificate chain. Additionally, it enforces the X.509 Proxy extensions for validating the certificate chain, if an X.509 proxy certificate exists in the chain. The GTS uses the validation criteria to identify a set of certificate authorities to validate the specified certificate chain against. The validation criteria are similar to discovery criteria, optionally allowing the set of certificate authorities to validate against to be limited based on the following: the name of the certificate authority, the trust level, the status, the lifetime, the source GTS, and the authority GTS.

5. SyncGTS

As mentioned, the current Globus Toolkit release (version 4.0.3) supports the LSLV profile for certificate validation. To meet the security requirements of caBIG, we needed to ensure that certificate validation is performed against the latest trust fabric. To this end, without having to modify the Globus Toolkit we created SyncGTS. SyncGTS is a plugin for the Globus Toolkit enabling it to support the RRLV profile. Globus performs authentication, or certificate validation, against a trusted certificates directory on the local file system. SyncGTS, shown in *Figure 9* maintains the local trusted certificates directory for Globus. When SyncGTS synchronizes with a set of GTS instances, it copies the CA certificates and CRLs to the local trusted certificates directory using proper Globus naming conventions, purging all certificate and CRLs that existed from the previous synchronizations. SyncGTS is configured with a synchronization description, which describes the synchronization criteria. Each synchronization criterion specifies a GTS to synchronize with and a set of search criteria for enforcing restrictions (trust levels, etc.) on the resulting certificate authority set. When SyncGTS is executed, it connects to all the GTS instances specified in the synchronization description and obtains a list of trusted certificate authorities and corresponding CRLs matching the specified search criteria.

For auditing purposes SyncGTS maintains a reporting model, *SyncReport*, describing each synchronization point. A report is generated and written to the local file system, each time SyncGTS synchronizes with a set of GTSs. The report describes when the synchronization occurred, the synchronization description executed, and the outcome of the synchronization. The outcome details which GTS a certificate authority and CRL came from and where on the file system the CA information and CRL were written. It also describes any certificate authorities and CRLs that were removed,

because they were remnants of a previous synchronization.

SyncGTS can support several deployment options. It can be embedded directly into a Globus container as a Grid service, enabling all services running in that container to perform authentication against the GTS maintained trust fabric. SyncGTS can also be embedded directly in client-side applications by leveraging the SyncGTS Java API. In addition, SyncGTS provides command line utilities, which can be leveraged on both the client and service sides.

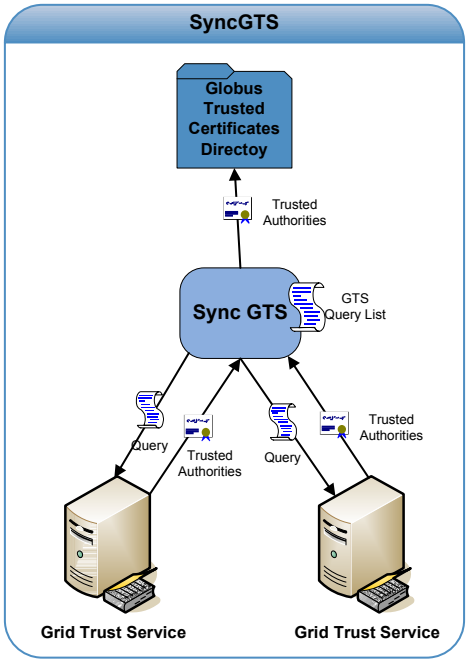

Figure 9 Conceptual View of SyncGTS

6. Related Work

Grid-wide management of security related information (such as certificates, credentials, user accounts, and authorization information) is a critical and challenging issue. The challenge stems from the fact that the Grid consists of autonomous end-points with their own security policies and infrastructure and it is a dynamic environment. A number of toolkits and service architectures have been developed, for example, to manage user credentials and user accounts. The MyProxy Credential Management Service [22, 23] is an open source project for managing X.509 Public Key Infrastructure. MyProxy provides the ability to manage private keys and certificates for Grid users, and also has an option to provision the trusted CAs and CRLs to the users. The current client implementation for the provisioning is C-based, and does not deploy webservice compliant protocols. Furthermore, it lacks automatic synchronization capabilities and possible aggregation mechanisms of provisioning information.

The Portal-based User Registration System (PURSe) [24] implements a user interface for users to register and access their Grid credentials. It uses SimpleCA and MyProxy as the backend system for management of Grid credentials. GAMA [25] is a GSI-based infrastructure that provides a backend server for creating and managing X.509 credentials for users and a suite of portals that serve as interfaces for users and administrators to access GAMA's functionality. Dorian [11] provides a Grid service infrastructure, based on the use of public key certificates and SAML assertions, for managing and federating user identities in the Grid. It facilitates combined use of SAML and Grid certificates to authenticate users to the Grid environment through their institution's authentication mechanism. The GTS differs from these systems in that it addresses the complementary problem of federating trusted identity authorities (i.e., Certificate Authorities) with different trust levels and validation of user certificates against this federated environment.

Management of trust is recognized as an important component of security in distributed environments [6, 12-21]. Manchala [12] describes trust models and metrics in e-commerce applications and discusses how risk can be analyzed under different models. Azzedin and Maheswaran [13] present a trust model for a Grid environment. Their approach models trust based on behavior and reputation of entities that interact with others. They describe techniques for computing this type of behavior trust, how it evolves in an environment, and how it can be managed in a Grid setting. GridAdmin, proposed by Quillinan, Clayton, Foley, [14] is a system that provides support for automatic handling of requests for administrative actions and resource allocations. The system incorporates trust metrics in responding to and ranking such requests. Weaver et. al. [17] discuss trust-sharing agreements and an IT infrastructure for federated security in distributed healthcare applications. Hwang and Tanachaiwiwat [18] propose a trust model and systems using this trust model for dynamic resource allocations. Grandison and Sloman [26] present a toolkit that provides support for specifying and monitoring trust relationships for Internet applications. Ahsant et. al. [27] discuss how business trust relationships can be propagated to the Grid environment and how these relationships can be

federated dynamically. Li, Zhu, and Lam [28] propose a two-level trust model, in which the first level defines trust relationships between virtual organizations and the second level (lower level) specifies trust relationships within a domain. Park, Moon, and Sohn [20] propose an approach and a service for validation of certificates in the Grid using XKMS. Their system implements support for realizing trust relationships in a dynamic environment. Basney et. al. [29] describes extensions to the basic Grid security architecture in order to support negotiation and dynamic establishment trust relationships between entities in the Grid. Thompson et. al. [21] discusses trust and trust models based on CAs for the Grid environment. Our work complements the previous work on trust management in that earlier work focused on specification of trust and establishment and management of trust between entities. The GTS, on the other hand, enables Grid-wide management of trusted Certificate Authorities with different trust levels.

7. Conclusions and Future Work

We have shown the need for secure multi-institutional Grids and have demonstrated current approaches for enabling access to trusted CA certificates as well as CRLs. We have outlined and analyzed the problems that are incurred when attempting to scale Grid security across multiple institutions, which each has its own set of authorized users as well as local security constraints. We have presented GTS, a design and implementation for providing a Grid-wide trust fabric that solves issues in curating a Grid-wide trust network. The GTS' ability to store and manage large networks of trusted CA certificates and their CRLs will enable simple and safe scalability in leveraging secure and trusted certificates in a Grid. This CA certificate trust network and corresponding CRLs provided by GTS as a Grid service will enable validating users and services from any institution against a known set of trusted certificate authorities. In conjunction with SyncGTS, GTS will enable the local caching of the trust network so that local user validation can occur. The combination of these tools, along with other Grid security measures such as federated identity management, not only will enable Grid security to be less cumbersome to manage for administrators but also will increase the ability to create much larger, more secure, and disparate multi-institutional Grids.

Although we use the tiered approach of the XKMS, our current implementation does not employ the XKMS defined protocols and interfaces in the GTS infrastructure. We plan to incorporate this into GTS in a future release. Moreover, we plan to extend GTS such that it can provision On-line Certificate Status Protocol (OCSP) responder information. We expect the Globus Toolkit to support OCSP in its future release and will ensure that GTS can provide a mechanism for OCSP-enabled clients and services to contact and trust the right authorities. Similarly we also plan on exploring adding support for the Server-based Certificate Validation Protocol (SCVP) [32]. Another planned future extension to GTS is the support for the provisioning of attribute and authorization authorities. With the latter information, client and service will be configured to allow them to query the relevant and trusted attribute and authorization services.

In the future we would like to collaborate with the IGTF to define a standard set of trust levels. We would also like to work with the IGTF to investigate how the GTS can be used to distribute IGTF accredited trust roots and whether Grids using the IGTF would be interested in leveraging our framework.

Acknowledgements:

This work was supported in part by the National Cancer Institute's Center for Biomedical Informatics, under the caBIG™ program (#79077CBS10), by the National Science Foundation Grant ANI-0330612, by the Ohio Board of Regents BRTTC program (#BRTT02-0003 and ODOD AGMT TECH 04-049), and by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Dept. of Energy, under Contract DE-AC02-06CH11357.

References

[1] Cancer Biomedical Informatics Grid (caBIG™), https://cabig.nci.nih.gov.

[2] The caGrid infrastructure, https://cabig.nci.nih.gov/workspaces/Architecture/caGrid/

[3] I. Foster, C. Kesselman, and S. Tuecke., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International J. Supercomputer Applications, 15(3), 2001.

[4] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", Technical Report,

Globus Project; http://www.globus.org/research/papers/ogsa.pdf, June 2002.

[5] The Globus Toolkit, http://www.globus.org.

[6] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, and S. Tuecke, "Security for Grid Services", Twelfth International Symposium on High Performance Distributed Computing (HPDC-12), IEEE Press, June 2003.

[7] I. Foster, C. Kesselman, S. Tuecke, J. Volmer, V. Welch, R. Butler, and D. Engert, "A National-Scale Authentication Infrastructure.", IEEE Computer, 33(12):60-66, 2000.

[8] Ken Lin and Gary Daemer "caBIG™ Security Technology Evaluation White Paper". https://cabig.nci.nih.gov/workspaces/Architecture/Security_Tech_Eval_White_Paper_Provisional, January 2006.

[9] Web Services Resource Framework (WSRF), http://www.oasis-open.org/committees/wsrf, 2006.

[10] XML Key Management Specification (XKMS) http://www.w3.org/TR/xkms/

[11] S. Langella, S. Oster, S. Hastings, F. Siebenlist, T. Kurc, and J. Saltz, "Dorian: Grid Service Infrastructure for Identity Management and Federation," The 19th IEEE Symposium on Computer-Based Medical Systems, Special Track: Grids for Biomedical Informatics, Salt Lake City, Utah., 2006.

[12] D.W. Manchala, "E-Commerce Trust Metrics and Models". IEEE Internet Computing 4(2): 36-44, 2000.

[13] F. Azzedin and M. Maheswaran, "Evolving and managing trust in grid computing systems," In the Proceedings of Canadian Conference on Electrical and Computer Engineering, vol.3, pp. 1424- 1429, 2002.

[14] T.B. Quillinan, B.C. Clayton, and S.N, Foley, "GridAdmin: Decentralising Grid administration using trust management". Third International Symposium on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks, pp. 184- 192, July 2004.

[15] M.H. Hanif Durad and C. Yuanda, "A vision for the trust managed grid". The Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops, 2006.

[16] Y. Demchenko, C. de Laat, and V. Ciaschini, "VO-based Dynamic Security Associations in Collaborative Grid Environment," International Symposium on Collaborative Technologies and Systems (CTS 2006), pp. 38- 47, May 2006.

[17] A.C. Weaver, S.J. Dwyer, A.M. Snyder, J. Van Dyke, J. Hu, X. Chen, T. Mulholland, and A. Marshall, "Federated, secure trust networks for distributed healthcare IT services," In Proceedings of IEEE International Conference on Industrial Informatics (INDIN 2003), pp. 162- 169, Aug. 2003.

[18] K. Hwang and S. Tanachaiwiwat, "Trust Models and NetShield Architecture for Securing Grid Computing". Journal of Grid Computing, 2003.

[19] N. Nagaratnam et. al., "Security Architecture for Open Grid Services". Global Grid Forum Working Draft. 2003.

[20] N. Park, K. Moon, and S. Sohn, "Certificate validation service using XKMS for computational grid". In Proceedings of the 2003 ACM Workshop on XML Security (XMLSEC '03), ACM Press, New York, NY, Oct 2003.

[21] M.R. Thompson, D. Olson, R. Cowles, S. Mullen, and M. Helm. "CA-based Trust Model for Grid Authentication and Identity Delegation", Grid Certificate Policy Working Group, Global Grid Forum, Oct, 2002.

[22] J. Novotny, S. Tuecke, and V. Welch., "An Online Credential Repository for the Grid: MyProxy". Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), pp. 104-111, IEEE Press, Aug 2001.

[23] J. Basney, M. Humphrey, and V. Welch., "The MyProxy Online Credential Repository". Software: Practice and Experience, Volume 35, Issue 9, July 2005

[24] PURSe: Portal-Based User Registration Service, http://www.grids-center.org/solutions/purse/

[25] K. Bhatia, S. Chandra, and K. Mueller, "GAMA: Grid Account Management Architecture," E-Science '05, pp. 413-420, 2005.

[26] T. Grandison and M. Sloman, "Trust Management Tools for Internet Applications". The First International Conference on Trust Management. Springer Verlag, 2003.

[27] M. Ahsant, M. Surridge, T.A. Leonard, A. Krishna, and O. Mulmo, "Dynamic Trust Federation in Grids". In Proceedings of the 4th International Conference on Trust Management, Pisa, Tuscany, Italy, 2006.

[28] T.-Y. Li, H. Zhu, and K.-Y. Lam, "A Novel Two-Level Trust Model for Grid". International Conference on Information and Communications Security (ICICS 2003), pp. 214-225, 2003.

[29] J. Basney, W. Nejdl, D. Olmedilla, V. Welch, and M. Winslett, "Negotiating trust on the Grid". The 2nd Workshop on Semantics in P2P and Grid Computing, New York, 2004.

[30] Joel H. Saltz, Scott Oster, Shannon L. Hastings, Stephen Langella, William Sanchez, Manav Kher, Peter A. Covitz, "caGrid: design and implementation of the core architecture of the cancer biomedical informatics grid", Ed. Alvis

Brazma, Bioinformatics, Vol. 22, No. 15, 2006, pp. 1910-1916.

[31] International Grid Trust Federation (IGTF), http://www.gridpma.org/

[32] T. Freeman, R. Housley, A. Malpani, D. Cooper, W. Polk. “Server-based Certificate Validation Protocol (SCVP)”, IETF Draft