A Design for Authentication and Authorization Infrastructure On the LIGO Data Grid

The Authentication and Authorization Subcommittee
of the LIGO Computing Committee

February 24, 2010

LIGO DCC LIGO-T0900516-v4

Abstract

LIGO wishes to remove the burden from users of requesting, retrieving, and managing X.509 digital certificates and the associated private keys. This new authentication and authorization infrastructure design for the LIGO Data Grid (LDG) includes deploying short-lived credential services (SLCS) using the MyProxy server at all LIGO computing sites. In this design LIGO and Virgo collaboration members use their LIGO credentials (Kerberos principal and password) to authenticate to the MyProxy server and obtain a proxy credential suitable for authentication to LIGO Data Grid services and resources. Users no longer have to request, retrieve, renew, or manage X.509 credentials. Additionally this design includes infrastructure for the automatic generation and deployment of grid-mapfiles to remove from LIGO administrators the burden of keeping by hand access control lists current. Lastly, this design includes details and specifications for a LIGO Root CA, a subordinate CA for signing host and service certificate requests, and subordinate CAs to be used with the LIGO SLCS.

Contents

1 Current authentication and authorization infrastructure

Today in order to authenticate to LIGO Data Grid (LDG) services users must first obtain a long-lived X.509 digital certificate from the DOEGrids Certificate Authority (CA) [1] if based in the United States or from one of the many CAs accredited by the EUGridPMA [2] and the APGridPMA [3] if based in Europe or Asia Pacific. Typically the long-lived X.509 certificate is valid for one year.

A LIGO user typically obtains a long-lived X.509 credential by first installing the LDG Client Package [4] or a similar set of tools. He then runs a command to generate a certificate request and an associated private key. The certificate request is then automatically uploaded to the DOEGrids CA or other CA infrastructure and, after a vetting and verification procedure, the certificate request is signed by the CA. The private key material is never transmitted over the network. The user is notified that the signed request is available and runs another script to download and put into place on his computer in a standard location the signed long-lived X.509 certificate.

Some CAs instead require a user to browse to a web page and fill out a web form. After filling out the web form the user clicks a button that causes the user's web browser to generate a private key and a certificate request. The request is uploaded by the browser to the CA infrastructure. Later, after vetting and verification, the request is signed by the CA and the user is notified that the request is available for download. He browses to a similar web page and clicks a button to have the signed certificate downloaded into his web browser repository. He then needs to follow a detailed procedure to export the signed certificate and key out of the browser and convert it into the correct format suitable for use with LDG tools and infrastructure.

After obtaining the long-lived X.509 certificate the user is responsible for managing the plain text files that store the certificate and associated private key. Before the certificate expires the user is required to renew the certificate. If the certificate is expired it cannot be used to authenticate to LDG services. If the user fails to renew his certificate he must obtain a new certificate, and he loses access to LDG resources until his new certificate with a new subject is entered into the grid-mapfiles by LDG administrators.

Another LIGO document details problems [5] LIGO users often encounter when requesting, retrieving, and renewing long-lived certificates issued by the DOEGrids CA and similar CAs. Please refer to that document for motivation on why it is desirable to remove the burden from LIGO users of requesting, retrieving, renewing, and managing long-lived X.509 credentials.

After a LIGO user obtains his long-lived X.509 credential he must fill out the LIGO VO Computer Resource Request Form [6], which is part of the LIGO Account Management System (LAMS), to request access to LDG resources (this form also allows all LSC users to request and receive access to resources offered by the Open Science Grid [7]). When the request is approved an email that includes the subject of the user's X.509 credential is sent to local LDG site administrators so that the administrator can create the necessary local accounts and authorize the user to access the local LDG resources. The mechanism for authorization is discussed in detail below. The afore mentioned document details the latency involved in this request[5]. Please see that document for motivation on reducing the latency.

Before he is able to authenticate to LDG services a LIGO user must run the command `grid-proxy-init` which generates from his long-lived X.509 credential a short-lived RFC 3820 proxy credential [8]. The proxy credential is used to authenticate to LDG services and delegate to those services the ability to authenticate on the user's behalf to other LDG services and resources. By default a proxy credential is valid for 12 hours. Using the proxy credential a LIGO user may authenticate to LDG services, but also may authenticate to services hosted by the Open Science Grid (OSG). Some LIGO users may also use their proxy credential to access certain EU grid resources (apart from the LDG sites in Europe).

A proxy credential derived from a long-lived X.509 credential signed by the DOEGrids CA or similar enables a LIGO user to authenticate to LDG services but he must also be authorized to access the service or resource. Currently LDG services authorize users by comparing the subject name of the proxy credential to a list of authorized subject names kept in a grid-mapfile. Some services require that subject names be mapped to local UNIX accounts and for those services the subject name appears next to one or more local UNIX account or login names. A typical entry in a grid-mapfile used on the LDG for authorizing access to LDG services looks like this:

```
"/DC=org/DC=doegrids/OU=People/CN=Scott Koranda 212488" skoranda
```

Although some LDG sites have created "once-off" local solutions to help manage grid-mapfiles, no elegant solution is currently deployed grid-wide and most often each grid-mapfile at an LDG site must be kept up to date by hand.

2 Short-lived credential service

This proposed design for the authentication and authorization infrastructure for the LDG centers on a short-lived credential service (SLCS) [9]. In this design a LIGO user must still present an RFC 3820 proxy credential to authenticate to LDG resources, but he obtains the proxy credential by first authenticating to a SLCS using his LIGO credential (currently a Kerberos principal in the @LIGO.ORG realm). After successfully authenticating to the SLCS using his LIGO credential the user receives from the service a short-lived X.509 certificate. The short-lived X.509 certificate is downloaded to the his computer and then used to automatically generate on his computer an RFC 3820 proxy credential (the generation of a RFC 3820 proxy credential from the short-lived X.509 credential is not strictly necessary since all services that require a RFC 3820 proxy credential will also accept the end-entity short-lived credential, but since LIGO users are accustomed to running `grid-proxy-info` to check on the status and lifetime of their credential we want to support that behavior). The LIGO user can use this proxy credential to authenticate to LDG resources in exactly the same way he uses a proxy credential generated today using `grid-proxy-init`.

2.1 Anatomy of the credentialing process

Figure 1 shows the steps during the credentialing process in this design. The steps are as follows:

1. The user runs in a shell the command `ligo-login` (The first time with his LIGO username as the argument; on subsequent invocations the LIGO username is retrieved from a cache file on disk if it does not appear on the command line. Note that `ligo-login` supports all of the appropriate command-line arguments for the tools it wraps, namely `kinit`, `myproxy-logon`, and `grid-proxy-init`).

2. Control passes from the `ligo-login` wrapper script to the `kinit` command and the user is prompted for his LIGO password. For example

```
$ ligo-login
Password for scott.koranda@LIGO.ORG:
```

3. The `kinit` tool contacts the Kerberos KDC.

4. Provided the user has entered his correct password a ticket for the user in the @LIGO.ORG realm is issued (the issued ticket is a Kerberos ticket granting ticket or TGT).

5. The `ligo-login` script passes control to the `myproxy-logon` command.

6. The `myproxy-logon` tool uses the Kerberos ticket [1] to authenticate to the SLCS and begin a session with the credentialing service. The session is encrypted using SSL/TLS.

7. A public and private key pair are generated on the user's computer.

8. The public key is sent to the SLCS. The private key does not go out over the network and is kept securely on the local computer. The private key is not encrypted.

9. The SLCS uses the public key and its own configuration to generate a X.509 certificate request. The common name or CN field of the subject in the request is in this design the unique Kerberos principal for the LIGO user. For example the subject in the request might look like

```
/DC=org/DC=ligo/O=LIGO/OU=People/CN=scott.koranda@LIGO.ORG
```

10. The embedded CA in the SLCS signs the certificate request. The new X.509 certificate is a short-lived certificate. The default lifetime for this short-lived certificate is 12 hours. The maximum possible lifetime is a matter of policy to be decided. The short-lived X.509 certificate is sent back to the user's computer. The SLCS caches a copy of the short-lived certificate (but not the associated private key, since it is never sent over the network to the SLCS) locally for auditing and similar purposes.

11. The `ligo-login` script passes control to the `grid-proxy-init` command.

[1] Not shown in detail in the diagram is the exchange between the myproxy-logon tool and the KDC to retrieve using the TGT from the KDC a ticket specifically for authenticating to the MyProxy SLCS server.

12. The `ligo-login` tool uses the short-lived X.509 certificate and the unencrypted private key to generate an RFC 3820 proxy credential. This proxy credential is completely equivalent to a proxy credential generated from a long-lived X.509 credential except that it is based on the shorter life credential. The lifetime of the proxy credential cannot be longer than the lifetime of the short-lived X.509 credential obtained from the SLCS. The proxy credential is located on the user's computer in the standard location so that standard grid tools can find and use it to authenticate to grid services.
13. The `ligo-login` tool then destroys the unencrypted private key and the short-lived X.509 certificate since they are no longer needed during the lifetime of the proxy credential.
14. The credentialing process is complete. The user may use the proxy credential to authenticate to LDG resources until the proxy credential expires.

Note that it is not strictly necessary to generate an RFC 3820 proxy credential from the short-lived X.509 credential and the associated private key. The short-lived X.509 credential can be used directly with all Globus GSI client tools and libraries (such as the GSI-enabled SSH clients `ssh` and `scp`, the GridFTP client `globus-url-copy`, and the GRAM clients `globusrun` and `globusrun-ws`) to authenticate to grid services in the exact same way as an RFC 3820 proxy credential is used. Nor is it strictly necessary for the X.509 credential and associated private key to be destroyed once a proxy credential is created from it. In this design, however, we prefer to have a proxy credential created automatically for the user to preserve the LIGO user's experience with the `grid-proxy-info` tool. And if the proxy credential is generated automatically then we prefer to destroy the short-lived X.509 certificate and private key so that they are not files about which a LIGO user needs to be concerned.

2.2 User experience

The LIGO user only needs to run the `ligo-login` tool and enter his associated password in order to generate the proxy credential as detailed in the previous section (the first time the `ligo-login` command is run the LIGO username is cached on disk for use with later invocations if it is not present on the command line). This is the same LIGO name and password used for authenticating to LIGO web resources in the new LIGO authentication and authorization infrastructure. The LIGO user does not need to remember any other login or password combination. The proxy credential obtained is equivalent in function to the proxy credential obtained currently by running `grid-proxy-init`.

In detail the LIGO user experience changes in these ways:

- The user does not have to generate a certificate request and send it to the DOEGrids or similar CA. By virtue of joining the LIGO/Virgo collaboration and being entered into the LIGO Roster the user obtains a LIGO username and password and is automatically and with low latency (less than 30 minutes) able to obtain a proxy credential.
- As with the current process the LIGO user must install some software before he can generate a proxy credential. The `ligo-login` script must be installed and configured. In this design that script and its (small collection of) software dependencies is available via native packaging including RPMs, Debian packages, Mac ports, and Solaris packages (details below). Below we explain that in this design we use the MyProxy server for the SLCS. MyProxy includes an option that enables the root certificates for CAs to be downloaded and automatically installed on a user's computer during the session. This further simplifies the deployment of the necessary software and configurations.
- The user no longer needs to manage the `$HOME/.globus/user[cert|key].pem` files containing a long-lived certificate and associated private key. There are no equivalents in this design.
- The user in this design does not need to ever renew a certificate. When the proxy credential expires the user simply runs `ligo-login` again (unless the user still has a valid Kerberos TGT, in which case the `ligo-login` tool will obtain a new short-lived proxy credential automatically for the user).
- The user can run `ligo-login` and obtain a proxy credential on any computer that has the `ligo-login` tool installed. There is no need to move around and manage a certificate and key file.

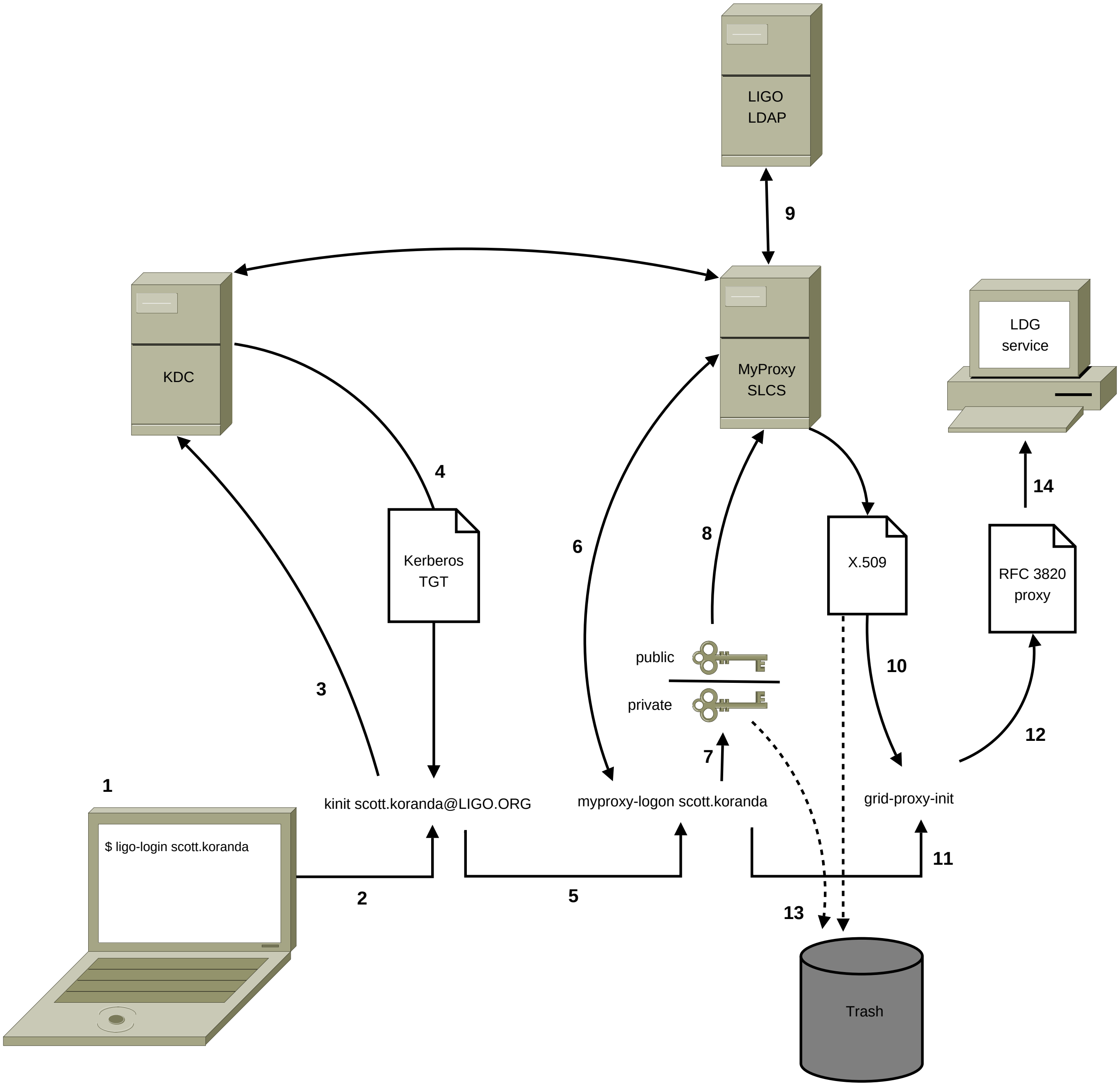

Figure 1: The anatomy of the credentialing process. See the main text for a description of each step. Note that the user only runs the `ligo-login` command and enters his LIGO password. The rest of the steps are transparent to the user and the result is a proxy credential familiar to LIGO Data Grid users. The user does not have to manage a X.509 certificate and private key.

2.3 MyProxy as a SLCS

This design uses the MyProxy Credential Management Service [10] as the SLCS. The MyProxy server includes the following functionality that makes it especially suited for this design:

- MyProxy provides a set of flexible authentication and authorization mechanisms for controlling access to credentials. Specifically it supports Kerberos authentication, either directly with GSSAPI or via PAM.
- The MyProxy server includes the ability to act as a Certificate Authority (CA) (which may be subordinate to a root CA), signing certificates with a configured CA key on request for authenticated users. Users can obtain a certificate from the MyProxy CA when and where needed, without needing to store long-lived keys and certificates in the MyProxy repository or elsewhere.
- MyProxy supports mapping of MyProxy usernames (the name used when authenticating to the server, in this design the LIGO username or Kerberos principal) via plain text grid-mapfiles, a general purpose call-out interface, and querying via LDAP.
- The MyProxy distribution includes Java, Python, Perl, PHP, and JAAS APIs as well as a rich set of command line tools that are straightforward to wrap. Additionally the MyProxy protocol is well documented.

We also note that the National Center for Supercomputing Applications (NCSA) runs a SLCS based on MyProxy and it has been fully accredited by TAGPMA [11].

2.4 LIGO deployment of MyProxy

In this design a MyProxy server configured as a SLCS is deployed at each LDG site. Deploying a MyProxy server at each site eliminates a centralized single point of failure and makes each site robust against network outages that cut the site off from the rest of the internet. Even if a site is cut off from the rest of the internet local users may authenticate using LIGO credentials (obtained from a local Kerberos KDC replica) to the local MyProxy server and obtain a proxy credential to be used to access local LDG services.

Each of the MyProxy servers will host its own CA to be used for signing the certificate requests. Each of these CAs will be subordinate to the LIGO root CA (see below). Each of these CAs will have a DN of the form

```
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=<SITE>/CN=MyProxy <N>
```

where `<SITE>` indicates the LDG site that is running the SLCS service and `N` is an integer indicating the ordering of the service at each site. For example the first SLCS service deployed at Caltech might have the DN

```
/DC=org/DC=ligo/OU=Certificate Authorities/OU=CIT/CN=MyProxy 1
```

The proposed set of CNs for the first set of deployed servers across the LDG is

```
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=CIT/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=MIT/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=LHO/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=LLO/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=SYR/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=UWM/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=AEI GLM/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=AEI HAN/CN=MyProxy 1
/DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=Cardiff/CN=MyProxy 1
```

Although each of the MyProxy SLCS deployed throughout the LDG will have a unique CN and will be based on a unique subordinate LIGO CA, each will issue short-lived X.509 certificates into the same name space. *The short-lived X.509 certificate and the RFC 3820 proxy certificate derived from it will be unique for each LIGO user across the entire federation of MyProxy servers, no matter from which MyProxy server the credential is obtained.* The MyProxy servers will issue short-lived X.509 certificates with DNs of the form

CN	*starting serial number*
OU=CIT/CN=MyProxy 1	100,000,000
OU=MIT/CN=MyProxy 1	200,000,000
OU=LHO/CN=MyProxy 1	300,000,000
OU=LLO/CN=MyProxy 1	400,000,000
OU=PSU/CN=MyProxy 1	500,000,000
OU=SYR/CN=MyProxy 1	600,000,000
OU=UWM/CN=MyProxy 1	700,000,000
OU=AEI GLM/CN=MyProxy 1	800,000,000
OU=AEI HAN/CN=MyProxy 1	900,000,000
OU=Cardiff/CN=MyProxy 1	1,000,000,000

Table 1: Starting serial numbers for LIGO subordinate CAs issuing short-lived X.509 certificates via MyProxy.

```
/DC=org/DC=ligo/O=LIGO/OU=People/CN=<KRB5 PRINCIPAL>
```

where `<KRB5 PRINCIPAL>` is the LIGO user's unique Kerberos prinicpal as held in the LIGO roster. For example:

```
/DC=org/DC=ligo/O=LIGO/OU=People/CN=scott.koranda@LIGO.ORG
```

Each of the subordinate CAs used by the MyProxy server will be configured to use its own range of serial numbers. The range for each server will span 100 million, enabling each server to issue up to 100 million short-lived certificates without resetting the counter. The proposed starting serial number for each of the first set of deployed servers across the LDG is shown in table 2.4.

2.5 Authorization with grid-mapfiles and LIGOmapper

The LIGOmapper tool [12] enables the automatic generation of grid-mapfiles based on the discovery of group membership and identity in an LDAP server. A detailed design of LIGOmapper is not presented here, but the general idea is based on the GUMS tool [12]. We expect LIGOmapper, since it runs as a client tool and directly queries LDAP, to be simpler than GUMS. In this design each LDG site deploys LIGOmapper on machines that host services requiring grid-mapfiles and the tool is configured to query the site's local LIGO LDAP server replica.

The LIGOmapper tool is configured to generate grid-mapfiles on a per host and per service granularity. For example, the grid-mapfile generated for the `globus-gridftp-server` on the host `ldas-cit.ligo.caltech.edu` may be distinct from the grid-mapfile generated for the `globus-gridftp-server` on the host `ldas-grid.ligo.caltech.edu`. Likewise, the grid-mapfile generated for the GSI-enabled `sshd` on the host `ldas-cit.ligo.caltech.edu` may be distinct from the grid-mapfile generated for the `globus-gridftp-server` on the same host. Wildcards in subnets are supported so that a grid-mapfile for all hosts in a subnet may be the same (again with per service granularity if so desired).

The grid-mapfiles are generated by the LIGOmapper tool running a script on the host on which the grid service runs that is to have authorization controlled by the generated grid-mapfile. The periodicity should be such that when changes occur in the LIGO roster, and hence the LIGO LDAP server network, the changes propagate and are reflected in the grid-mapfiles in an appropriately timely matter. This design calls for the cron job that generates each grid-mapfile to run with a periodicity of five (5) minutes.

Each mapping between host/service combinations and groups may include multiple groups so that it is straightforward to configure LIGOmapper to create grid-mapfiles that enable authorizations for multiple groups as defined in the LIGO LDAP server. Of course LIGOmapper can be configured to use a generic group such as "All users" (provided that group is defined in the LIGO LDAP database) in order to simplify the configuration for authorization to services to which all LDG users should have access. Further, by using wildcards LIGOmapper can be configured to produce the same grid-mapfile for multiple hosts and service combinations if so desired.

The design of groups within the LIGO LDAP database and the detailed base design of the LIGOmapper tool be presented in a later document.

2.6 Transition from the current infrastructure

LIGO administrators can smoothly transition LDG services with this design by using the LIGOmapper tool to splice into the grid-mapfile the current static grid-mapfile maintained by hand at each site. As users transition from using existing credentials issued by the DOEGrids and similar CAs to the credentials issued by the LIGO MyProxy SLCS the entries in the static grid-mapfile that is spliced in may be removed.

On the user's computer the transition is also straightforward since the `ligo-login` tool will not be made available for installation until the LIGO MyProxy SLCS are available for production use. Because the installation and use of `ligo-login` will not interfere or be impacted by the installation and existence of current tools no abrupt transition for the users is necessary.

2.7 Use of existing X.509 credentials by Virgo and other users

Users wishing to continue to use X.509 credentials issued by the DOEGrids CA or similar, including Virgo users issued credentials signed by the Italian and French national CAs, can be enabled to do so by setting the appropriate LIGO LDAP database attribute for the user (presently the 'description' attribute) to contain the DN for the certificate she wishes to use. For example if the LDAP database contains the entry

```
dn: employeeNumber=1232,ou=people,dc=ligo,dc=org
    uid: frederique.marion
    cn: Frederique Marion
    description:/O=GRID-FR/C=FR/O=CNRS/OU=LAPP/CN=Frederique Marion
```

then the grid-mapfile for the user will contain the line

```
"/O=GRID-FR/C=FR/O=CNRS/OU=LAPP/CN=Frederique Marion" frederique.marion
```

The cost for this flexibility is the user must first use his LIGO.ORG login and password to first authenticate to a web page and fill out a form to indicate the certificate subject that he wishes to have placed in the description attribute in the LIGO LDAP server. This is because although the infrastructure is configured to trust that the X.509 certificate signed by the French national CA was issued to the user with proper name Frederique Marion, there is no mechanism in the infrastructure to map that proper name and certificate name to a LIGO credential (Kerberos principal) since the subject name on the certificate is not generated automatically by the infrastructure as it is with the short-lived certificate issued by the LIGO SLCS. So the user must take the extra step to seed the infrastructure with the mapping from his long-lived certificate to his LIGO login name.

2.8 Packaging and distribution

In this design the following tools and software are provided using native packaging such as RPMs, Debian packages, Mac ports, and Solaris packages:

- User tools
  - `ligo-login` script and configuration
  - MyProxy user tools
  - Globus GSI user tools (i.e. grid-proxy-info)
  - GSI- and GSSAPI-enabled ssh client tools
  - Common Globus user tools (i.e. globus-url-copy)
  - LIGO Kerberos configuration file
- Services
  - MyProxy server and appropriate LIGO configuration
  - LIGO Kerberos configuration file
  - LIGOmapper client tool for generating grid-mapfiles
  - Root certificates for all LIGO CAs

It is hoped that the VDT [13] team will be able to act as an upstream packaging provider and provide packaged source distributions of many of these tools ready for simple reconfiguration and distribution by LIGO.

3 LIGO certificate authorities

The motivation for deploying LIGO certificate authorities is discussed in two other documents [5][14]. Please refer to those two documents for the motivations for LIGO deploying a root CA and a subordinate CAs.

3.1 LIGO root CA

In this design the LIGO root CA serves only to sign the certificates for LIGO subordinate CAs. The root CA will not sign any end entity, host, service, or other certificate requests.

This design includes the following details for the LIGO Root CA (a full listing of all configuration and policy related details will need to be developed in the certificate policy and practice document prepared for the CA, and all details in that document will supersede any in this design document):

- The CA root certificate will be a self-signed certificate with the subject

  ```
  /DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/CN=LIGO Root CA 1
  ```

- The CA will only sign certificate requests with subject names matching the following pattern:

  ```
  /DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities
  ```

- The private key for the LIGO Root CA must be kept encrypted on all storage media, including the computer's disk(s).
- The passphrase for decrypting the LIGO Root CA private key must not be written down or stored anywhere other than on off-line media in secure places where access is controlled.
- The CA computer, where the signing of certificates for subordinate CAs will take place, must not be connected to any network and the exchange of certificate requests and signed certificates MUST ONLY happen using physical media such as memory tokens, DVDs or CDs, or similar.
- The CA computer must be located in a secure environment where access is limited to specific trained personnel.
- The CA signing key must have an acceptable minimum length at the time the key is created.
- Copies of the encrypted signing key must be kept on off-line media in secure places where access is controlled.
- The CA signing certificate shall have a lifetime of 30 years.
- The CA root certificate must be pubslished on the internet as a root of trust.
- The CA must issue and publish a CRL.
- A certificate policy and practice statement, modeled as much as possible on authentication profiles published by the IGTF will be developed and published with a globally unique object identified (OID) based on the LIGO PEN number. This document will detail further the policies laid out above.

The process for generating and signing a subordinate CA certificate using the LIGO Root CA is:

1. An individual who will be responsible for the secure operation of the subordinate CA will be identified.
2. The certificate request for the subordinate CA and the associated private key are generated on the computer on which the subordinate CA will be hosted. The private key must not ever be sent over a network.
3. The certificate request is sent to the administrator of the LIGO Root CA, along with the contact information for the individual responsible for the subordinate CA.
4. The administrator of the LIGO Root CA must verify via a secure channel that the subject name on the request is the same as requested by the administrator submitting the request, and that the contact information for the individual responsible for the subordinate CA is correct and current.

5. The administrator of the LIGO Root CA must via a secure channel obtain explicit permission from the chair of the LIGO Computing Committee to sign the request for the subordinate CA.

6. The administrator places a copy of the request onto a secure physical medium such as a memory stick, DVD, CD, or similar and transfers the request to the computer for the LIGO Root CA.

7. The administrator signs the request and places a copy of the signed certificate onto a physical medium such as a memory stick, DVD, CD, or similar and transfer the request to another computer so that it can be delivered to the administrator making the request. The signed certificate is a public document so it need not be securely communicated back to the administrator making the request, but again the computer hosting the LIGO Root CA must never be connected to a network.

3.2 LIGO subordinate CA for host and service credentials

In this design a *single* LIGO CA, subordinate to the LIGO root CA, is available for signing host and service certificates. This CA will not sign any end entity certificate requests representing people. The CA will sign requests for "robot" certificates used by services acting in the role of client to authenticate to other remote services. The LIGO Data Replicator (LDR) LDRTransfer daemon is an example of a service using robot certificates.

This design includes the following details for the LIGO subordinate CA (a full listing of all configuration and policy related details will need to be developed in the certificate policy and practice document prepared for the CA, and all details in that document will supersede any in this design document):

- The CA certificate will be signed by the LIGO Root CA certificate and will have the subject

  ```
  /DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=Services/CN=LIGO CA 1
  ```

- The CA will only sign certificate requests with subject names matching the following pattern:

  ```
  /DC=org/DC=ligo/O=LIGO/OU=Services
  ```

  The appropriate signing policy file to help enforce that requirement must be created and distributed with the certificate for the CA.

- The private key for the CA must be kept encrypted on all storage media, including the computer's disk(s).

- The passphrase for decrypting the LIGO private key must only be kept where access is controlled.

- The CA computer, where the signing of service certificates will take place, MUST be connected to a highly protected and monitored network which will be accessible from the internet.

- The CA computer must be located in a secure environment where access is limited to specific trained personnel.

- The CA signing key must have an acceptable minimum length at the time the key is created.

- Copies of the encrypted signing key must be kept on off-line media in secure places where access is controlled.

- The CA signing certificate shall have a lifetime of 20 years.

- The CA certificate must be pubslished on the internet as a root of trust.

- The CA must issue and publish a CRL.

- A certificate policy and practice statement, modeled as much as possible on authentication profiles published by the IGTF will be developed and published with a globally unique object identified (OID) based on the LIGO PEN number.

- The process for requesting and obtaining service certificates from the CA will be published in a conspicuous place available on the internet to all LIGO/Virgo collaboration members.

During a first phase the following process will be used for requesting and obtaining host/service certificates from the CA:

1. The host/service certificate request and associated private key are generated by the requesting administrator. The request and private key may be generated on a computer different from the one on which it will be deployed, but that practice is discouraged. If the private key must be generated on a computer different from the one on which it will be deployed the private key must only be transferred in a secure way. If the private key is transferred over a network it must be done over an encrypted channel (for example using `scp`). The original copy on the original host must be properly destroyed (for example using the `shred` tool).

2. The host/service certificate request is emailed to `certrequest@ligo.org` along with the name and contact information of the LIGO administrator requesting the certificate.

3. The administrator of the LIGO CA that signs host/service certificates must verify via a secure channel that the administrator named in the email did indeed submit the request, and that the contact information for the requesting adminstrator is correct.

4. The administrator of the LIGO CA must verify that the requesting administrator is authorized to request host/service certificates. The chair of the LIGO Computing Committee has the final authority to determine which LIGO administrators are authorized to submit host/service certificate requests.

5. The administrator of the LIGO CA signs the request and emails the signed request to the requesting administrator.

During a second phase a web form for submitting host and service certificates requests to be signed by the LIGO CA will be available. Only authorized administrators may authenticate using LIGO credentials and use the web form to submit the request. After the request is successfully submitted using the form a script will automatically sign the request and issue the signed certificate. The signed certificate will be displayed in plain text so that it can be easily downloaded. Another option will be available to have the signed certificate emailed to the @LIGO.ORG email address for the requesting administrator.

3.3 LIGO subordinate CAs for SLCS servers

In this design a LIGO CA, each subordinate to the LIGO Root CA, is available at each LIGO Data Grid site for signing short-lived end-entity certificates as part of a SLCS as detailed above. The CAs used for the SLCS will not sign any requests for host or service certificates.

This design includes the following details for the LIGO subordinate CAs used for the SLCS (a full listing of all configuration and policy related details will need to be developed in the certificate policy and practice document prepared for the CA, and all details in that document will supersede any in this design document):

- The CA certificate will be signed by the LIGO Root CA certificate and will have a unique subject for each site of the form

  ```
  /DC=org/DC=ligo/O=LIGO/OU=Certificate Authorities/OU=<SITE>/CN=MyProxy <N>
  ```

  as detailed above.

- The CA will only sign certificate requests with subject names matching the following pattern:

  ```
  /DC=org/DC=ligo/O=LIGO/OU=People
  ```

  The appropriate signing policy file to help enforce that requirement must be created and distributed with the certificates for the CAs.

- The private key for the CA will be kept unencrypted so that the MyProxy server may automatically sign requests after appropriate authentication to the MyProxy server.

- The CA computer hosting the SLCS must provide a high level of protection and must not depend on the network environment for any protection. It must not run any other services and have any accounts other than for those people who administer the machine. It cannot have any remote root login and the root password must be different from that of any other hosts.

- The CA computer must be located in a secure environment where access is limited to specific trained personnel.
- The CA signing key must have an acceptable minimum length at the time the key is created.
- Copies of the encrypted signing key must be kept on off-line media in secure places where access is controlled.
- The CA signing certificate shall have a lifetime of 20 years.
- The CA will NOT issue certificates with a lifetime of more than one (1) million seconds.
- The CA must issue and publish a CRL.
- A certificate policy and practice statement, modeled as much as possible on authentication profiles published by the IGTF will be developed and published with a globally unique object identified (OID) based on the LIGO PEN number.

3.4 Packaging and distribution

In this design the certificate for the LIGO Root CA and all of its subordinate CAs are available via a number of formats so that they may easily be obtained for use with server and client grid tools:

1. Downloadable as plain text from a public web server.
2. Downloadable as a single compressed tarball containing all of the CA certs.
3. Installable using native packaging including RPMs, Debian packages, Mac ports, and Solaris packages.

3.5 Accreditation

The properly deployed LIGO Root CA and the subordinate CA for signing host and service certificates should be able to obtain accreditation by TAGPMA since they are to be based on authentication profiles from IGTF that have numerous instances in production throughout the world.

The design for the LIGO SLCS service is also based on an authentication profile from the IGTF. While the authentication profile only mentions a single SLCS, NCSA maintains a production deployement of multiple, coordinated SLCS based on MyProxy in much the same way as the design proposed here. The NCSA deployment is accredited by the TAGPMA. The primary difference between the accredited NCSA production deployment and the proposed design is the scale–ultimately we expect as many as 10 LIGO SLCS working together issuing credentials into the same but managed namespace.

References

[1] http://www.doegrids.org

[2] https://www.eugridpma.org/

[3] http://www.apgridpma.org/

[4] http://www.lsc-group.phys.uwm.edu/lscdatagrid

[5] LIGO document T080174-00-Z

[6] http://ligo.aset.psu.edu/users/account.shtml

[7] http://www.opensciencegrid.org

[8] http://www.ietf.org/rfc/rfc3820.txt

[9] http://www.tagpma.org/authn_profiles/slcs

[10] http://grid.ncsa.uiuc.edu/myproxy/

[11] http://security.ncsa.uiuc.edu/CA

[12] https://www.racf.bnl.gov/Facility/GUMS

[13] http://vdt.cs.wisc.edu

[14] In development.