GFD.18

Grid Certificate Policy WG

T. Goss-Walter, Deutscher Wetterdienst
R. Letz, Deutscher Wetterdienst
Dr. T. Kentemich, Pallas GmbH
H.-C. Hoppe, Pallas GmbH
P. Wieder, Forschungszentrum Jülich

July 2003

An Analysis of the UNICORE Security Model

Status of This Memo

This memo provides information to the Grid community.
Distribution is unlimited.

Document Version: 1.06

Abstract

This document provides information about the UNICORE security model. It summarizes the current architecture of the UNICORE PKI, describes the certificate generation process and the range of application of certificates within UNICORE.

A key feature of the UNICORE security model is job authentication and secure transmission of data. The security model supports both job signing and data encryption, which protects remote users against data theft and data manipulation. It also offers the HPC centers a high level of assurance against illegal usage as well as jobs containing malicious code.

The focus of this document is on the UNICORE Public Key Infrastructure (PKI). It outlines the hierarchy of Certifying Authorities (CA) and Registration Authorities (RA) and describes the different kinds of certificates.

The UNICORE PKI will be restructured before UNICORE becomes fully operational in 2003. The first section provides a general overview of the UNICORE architecture, components, and operational model. The second ection summarizes the current UNICORE PKI architecture. The third section analyzes the PKI architecture and outlines alternative solutions to overcome limitations of the current model.

The fourth section discusses the UNICORE certificate policy and compares it to the common CP reference model of the Grid Certificate Policy working group of the Global Grid Forum.

Table of Contents

1 UNICORE Functionality

1.1 End–User Functionality

The end–users of UNICORE interact with all sites and systems in a UNICORE Grid via a graphical Client that runs on a wide variety of Java–enabled systems, including Unix and Windows platforms. Regardless of the target site for a job, the connection process is always the same:

- The user starts the UNICORE client.
- The user unlocks the UNICORE certificate (by entering the keystore password).
- The Client loads any plugins that the user has configured in the plugin directory, and makes the plugin functions available to the end–user.
- The Client automatically connects to the UNICORE Grid sites from a user or system–defined list. It retrieves the list of available resources for each site, including information about available execution systems, special software & hardware support, archive and storage systems etc.
- The user can now load and/or construct new jobs, submit them to the available computing resources, inspect the status of submitted jobs, and finally retrieve results for jobs that have been (partially) completed.

UNICORE supports nomadic use, in that it doesn't matter from which location or platform an end–user is connecting from, as long as her certificate is available. The user authentication is performed by validating the UNICORE X.509 certificate that has been presented by the end–user. Once connected to UNICORE, different tasks can be performed:

- Construct or modify a UNICORE (batch) job and submit it for execution.
- Inspect the queued, running or terminated UNICORE jobs and retrieve results.
- Perform data transfer or file management functions.
- Access functions of the loaded plugins.
- Manage the local keystore – add/modify user certificates, select an identity from several certificates, add trusted certification authorities for Gateways or plugins.

1.1.1 Job Construction and Submission

UNICORE jobs are constructed as a directed a-cyclic graph (DAG) of actions reflecting the time–order in which to execute the actions. Actions can be either indivisible tasks, or they can in turn be composed in a recursive manner of a DAG of actions. Normally, the execution of a UNICORE jobs stops whenever one of the actions does not complete successfully. Special flow–control tasks allow the user to specify an alternative execution path in this case, or more generally enable the user to construct if–then–else branches and loops depending on runtime conditions.

A job is constructed using *abstract* terms, substituting the platform–dependent commands and parameters for the resource requirements, data transfer etc. by a platform–independent representation. The UNICORE Network Job Supervisor (NJS) server will translate from this abstract definition to a *concrete* sequence of commands and options suitable for the selected execution platform. The end–user doesn't have to know the platform–specific details, and a UNICORE job can be easily re–targeted to a different site or execution system.

The end user specifies on which system a job should run; it is also possible to specify different execution systems for different parts of the job, thus creating a job that will run on

multiple systems possibly belonging to multiple sites. Data transfer is handled transparently by the UNICORE system, given that the end user has specified which files are needed and which are produced by each action within the job.

For each task, the end–user can specify the *resources* required to run this task; supported resources include CPU or node count, computing time, memory size etc. The resource model is designed to be extensible, so that other required software or hardware resources can be defined, provided by a site and requested by a task. Each of the UNICORE sites defines the resources made available by each of their systems, and the client performs a check on whether the required resources are actually available.

UNICORE jobs can be saved to disk on the machine running the client, and later may be loaded to make modifications or submit the job again. Both a proprietary binary and an open XML–based job store format are supported.

When an end–user is satisfied with the constructed job, she can submit (or *consign* in UNICORE–speak) the job to the target Virtual Site (Vsite). The complete job definition is then transmitted as an *abstract job object* (AJO) using the *Unicore protocol layer (*UPL layer) to the UNICORE NJS server running at the target site. This server sends an acknowledge reply, and after that the job status can be controlled by the monitoring functions described in the section below.

1.1.2 Job Monitoring

Once a job is submitted, the UNICORE NJS server tracks its status, and by using the monitoring functions of the Client, the status can be monitored, and tasks or job groups can be killed or put on hold. Whenever an action (task or sub–job) is terminated (either normally, by job monitor commands, or because a failure), the user can retrieve the results (standard output, standard error, result files) and transmit them to the local workstation. After all results have been retrieved, the job information can be purged from the UNICORE system.

A UNICORE job is actually run under control of an execution agent on the target system. Normally, this will be a batch system, or an interactive shell on systems that provide “near–interactive” access. As far as the UNICORE system is concerned, the following job states (both for job groups and for tasks) are supported:

- SUCCESSFUL: job or task has been run successfully. For a job, this means that all components have run successfully.
- FAILED: execution of the job group or task has failed. For a job, this means that at least one of the components has failed, while other components may have run successfully. The end–user can inspect the status of the components to find out which have failed. For a task, this means that the task has failed. A message indicating the reason of failure can be displayed.
- PENDING: job or task is queued within the UNICORE system. This happens if a predecessor task has not yet been executed, or if the whole job has not yet been started at all.
- QUEUED: job or task is queued in the target batch system. This happens if the resources on the target system are used already by other job which may or may not be UNICORE jobs.
- EXECUTING: job or task is currently executing, that is running in the target batch system and is not yet completed. For a job group, this is indicated if at least one component is being executed.

To enable flow control of UNICORE jobs, a task can be marked with an `IgnoreFailure` flag that prevents the whole job from being terminated as the result of a that task failing. The flow control tasks work on the basis of the return codes from previously completed actions.

1.1.3 File and Data Management

The UNICORE system includes functions to

- Transmit files between the local user workstation and file systems or archives that can be accessed from the target systems or the UNICORE site.
- Perform Unix–style file management functions (copy, move, delete, chmod etc.) on files residing on file systems or archives that can be accessed from the target systems or the UNICORE site, and generate directory listings.
- Transfer files to and from file systems or archives that are accessible.

1.2 Computer Center Functionality

A computer center running the UNICORE gateway and servers and thus providing a UNICORE Grid site (a U-Site) can benefit from the improved end–user interfaces and the simplified way to access its systems for everyday work by non–expert end–users. Furthermore, users are encouraged to use all of the available systems, by the mere fact that moving a UNICORE job to a new system is a very easy task, once that system is supported by UNICORE. This will lead to better utilization of resources, and at the same time to lower training and support requirements.

The UNICORE functionality relevant for the computer centers comprises

- Provision of a strong user authentication mechanism (X.509 certificates)
- Extensibility by site–specific user authentication methods (like SecurID cards, SKEY one–time passwords etc.)
- Compatibility to the center's authorization mechanisms and policy (mapping of UNICORE userIDs to local Unix userIDs, accounting, disk quotas etc.)
- Site– and system–specific incarnation of UNICORE jobs driven by a declarative *Incarnation Database* that can be adapted to the center's needs.
- Declarative description of available resources, both traditional *capacity resources* (like processor count, computation time, memory size) and *capability* resources (like available software packages and special hardware capabilities). Extensibility to support dynamically changing resources in the future will be provided.
- Special support for important applications that simplifies the entry and definition of jobs, or provides steering or control functionality.

2 The UNICORE PKI

The UNICORE Public Key Infrastructure (U-PKI) as set up within the UNICORE Plus project is based on a centralized PKI architecture with a single CA and multiple RAs utilizing X.509 certificates. It consists of the following entities:

- Trusted Root CA (currently the CA of Deutsches Forschungsnetz e.V. (DFN)),
- UNICORE CA (currently hosted by Leibniz Rechenzentrum in Munich),
- Certificate Revocation Lists (CRLs) of Root CA and U-CA,
- UNICORE RAs,
- Gateway Certificates on each UNICORE Gateway,
- NJS certificates for distributing sub-jobs to different V-Sites,
- Client (or user) Certificates for all accepted UNICORE users and
- Client Certificates for UNICORE developers used for code signing.

This section summarizes the role of the individual components and outlines the certification process. We also provide a process flow diagram showing where and how both user and server certificates are used.

2.1 An Overview about the UNICORE PKI

UNICORE is an integrated Grid middleware system used in the Virtual Organization (VO) set up by the project partners in the UNICORE Plus project. The UNICORE Plus project VO includes large computing centers, public offices, universities, commercial sites and private users.

The sites within the VO need to work together across a common, prevalent network infrastructure. However, using the Internet as the communications platform introduces significant security risks. As the Internet is a very heterogeneous public network, all job and data transmissions have to be protected against

- data manipulation or deletion and
- data theft (e.g. wire tapping).

In particular for commercial sites data theft is a potential growing risk in a distributed computing environment.

At the same time, the target sites need to verify the identity and access rights of users, who submit jobs to run on the target sites' resources and must also verify that the jobs they receive for execution belong to the appropriate users.

For secure data communication and user authentication the UNICORE Plus project employs a Public Key Infrastructure. Certificates are used to

- authenticate users,
- authenticate UNICORE Gateways,
- authenticate the NJS (for distributing sub-jobs),
- sign jobs, and
- sign software.

Figure 1 gives an overview about the U-PKI. Currently, there is a two level CA hierarchy, the Root CA (DFN-CA), which is not part of UNICORE itself, and the UNICORE CA (U-CA)[1]. The Root CA is used to sign the UNICORE CA and to guarantee the integrity of the UNICORE

[1] In practice, it is a three level PKI hierarchy consisting of the DFN-CA, UNICORE-CA and UNICORE Signer CA. For the purpose of this discussion, the UNICORE-CA and UNICORE Signer CA will be considered as one single U_CA.

CA. The Certificate Revocation List (CRL) of the Root CA may revoke the self-signed certificate of the Root CA as well as the certificate of the UNICORE CA.

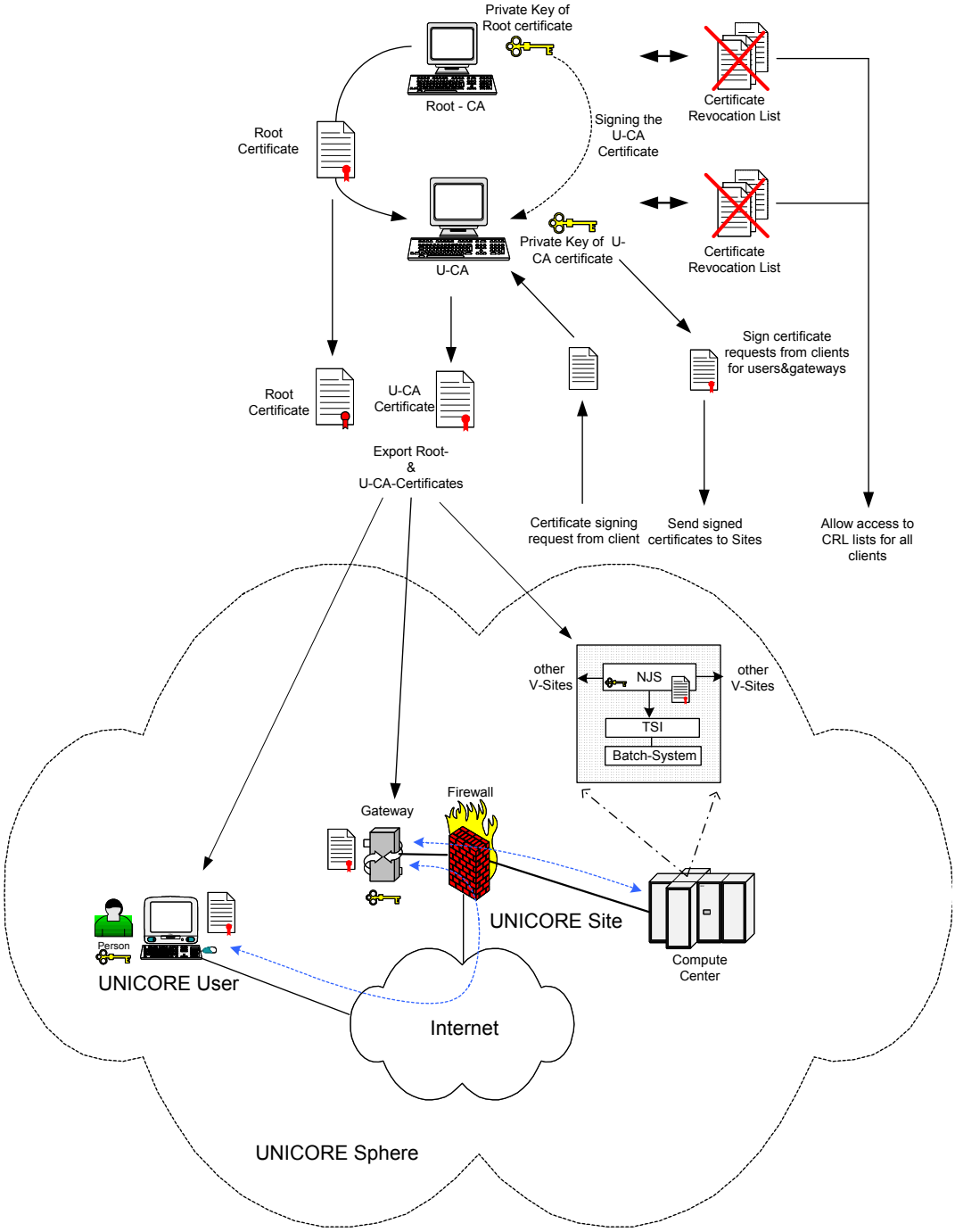

Figure 1

The Root CA should always be a trusted, “well-known” CA. Such a CA is operated according to applicable law[2] with strict security standards.

The UNICORE CA (U-CA) currently is a dedicated CA which is central to the UNICORE Plus project community.

[2] International CA which complies to the different countries’ laws.

The certificate of the U-CA is signed by the Root CA and is given the necessary rights to sign subordinated user- and server certificates. The U-CA utilizes its own CRL to revoke obsolete certificates and certificates with a broken seal, i.e. certificates whose private key has been stolen or lost. Both the root and the U-CA certificates as well as the CRLs are made available to all clients, so that they can verify the certificate chain. A low level certificate such as a user or a server certificate is only valid if all superordinated certificates are valid. Otherwise the whole chain has to be rebuilt starting with the first invalid certificate and then moving down the chain.

The lowest level certificates are the Client, Gateway and NJS certificates. These are signed by the U-CA and may be revoked by the U-CA if they become invalid. A UNICORE user uses his Client certificate to authenticate himself against the UNICORE Gateway of the target site(s) where he wants to run his job(s), to *endorse* the jobs (by signing with the private key) and to *consign* the job to the primary target site. The Gateway may be located within the Service Network of the target sites' firewall, but this is not mandatory. In essence, then, the user certificate serves as an electronic ID card.

The Gateway certificate is used to authenticate the UNICORE Gateway to the user. When communicating to a (remote) UNICORE site (U-Site) the user can thus validate the authenticity of the remote gateway. Only the signer of the gateway certificate is checked; the user does not have the gateway's public key.

There is another type of Client certificate which is used by the NJS to establish secure SSL connections with virtual sites (V-Sites) and to consign sub-jobs there. These certificates are used to authenticate the NJS against the target sites.

The current U-CA architecture is described in more detail in [Boet1].

2.2 The CA and RA Infrastructure

The security within a PKI depends on three contributing factors:

- how well a user is authenticated by the RA before the certificate is issued
- how safe the private key of the certificate is stored within the client system
- how well the certificate chain and the CRLs are verified for each communications relation

User verification is done through RAs proving the identity of a user by checking his ID card, for instance.

Each UNICORE Plus project partner should have his own RA which is registered with the U-CA. Partners who want to have their own RA should use an appropriate RA of one of the other partners, e.g. the RA of a U-Site where the partner's target machines are located.

Figure 2 shows the CA & RA structure as well as the certification process. The following steps need to be performed in order to get a signed Client certificate for a user. Server certificates are treated the same way, except for the fact that the respective U-Site administrator applies for the appropriate certificates.

1. A user generates a certificate signing request (CSR) with a corresponding private/public key pair. The private key is safely stored within the client's local keystore.
2. The CSR is sent to the responsible RA. This could be the local RA or an appropriate remote RA. The RA tells the user how validation should be performed. This could be done through
   a) personal identification (user visits the RA) or
   b) identification through video conferencing.
3. The RA validates the user and user's CSR. In the current security model only method a) is specified. However, method b) is currently being tested.
4. The RA changes the status of the user's CSR online in the database at the U-CA.

5. The U-CA generates a valid, signed certificate out of the CSR and delivers it to the user. The certificate could then be stored in the public LDAP server of the user's local site.

The certificate policy (CP) written to accommodate the needs for the UNICORE Plus project can be found in [Boet2].

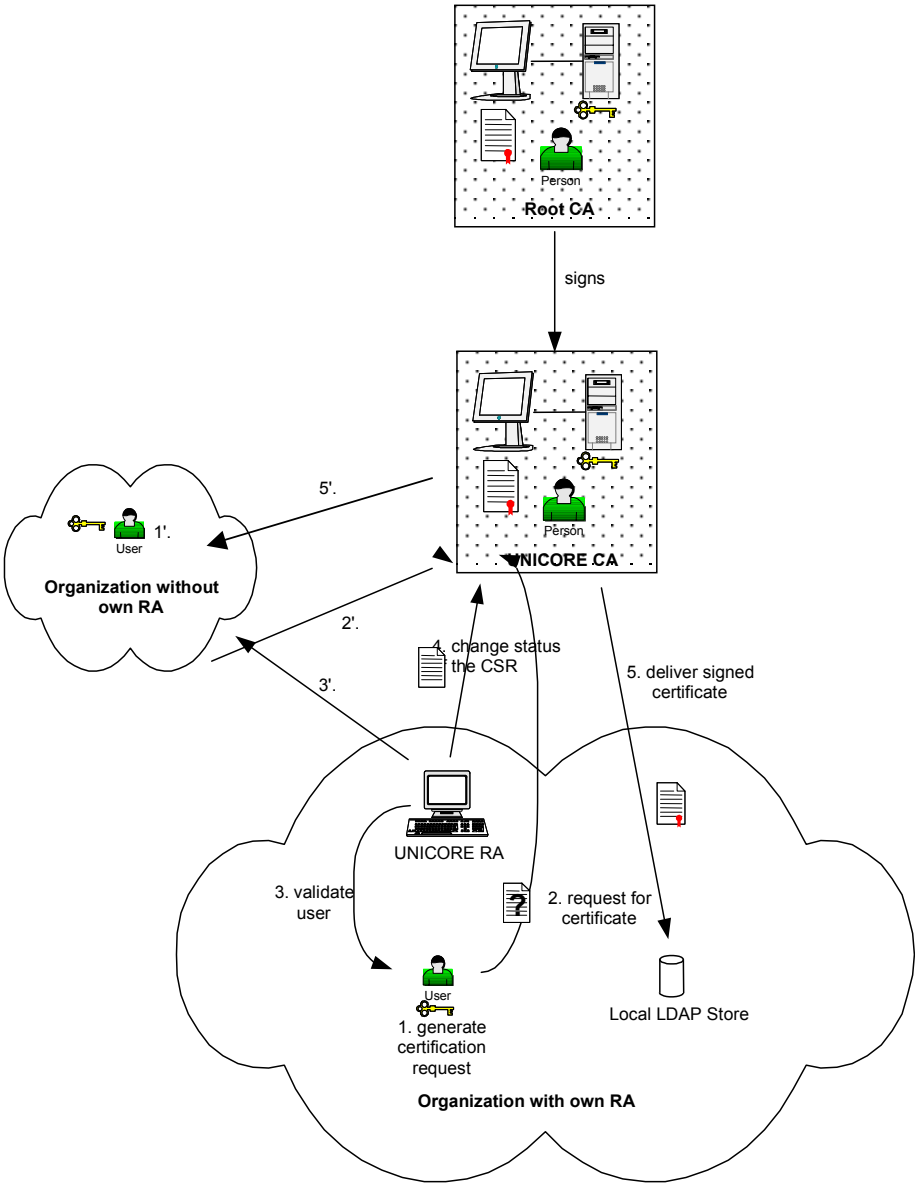

Figure 2

2.3 Process Flow on the Application Layer

Once the certificates have been generated and delivered to the participating users and sites, communication and job distribution can take place.

There are two kinds of actions that rely on certificates for user and server verification and authorization:

- establishment of secure communication channels using SSL
- endorsing of AJOs (UNICORE jobs) by signing them

Before running UNICORE all participating sites have to import both the certificate of the Root CA and the certificate of the U-CA. They are needed to verify the client and server certificates. The PCs and workstations involved must have access to the CRLs in order to check for revoked and therefore invalid certificates. Figure 3 gives an overview about the process flow.

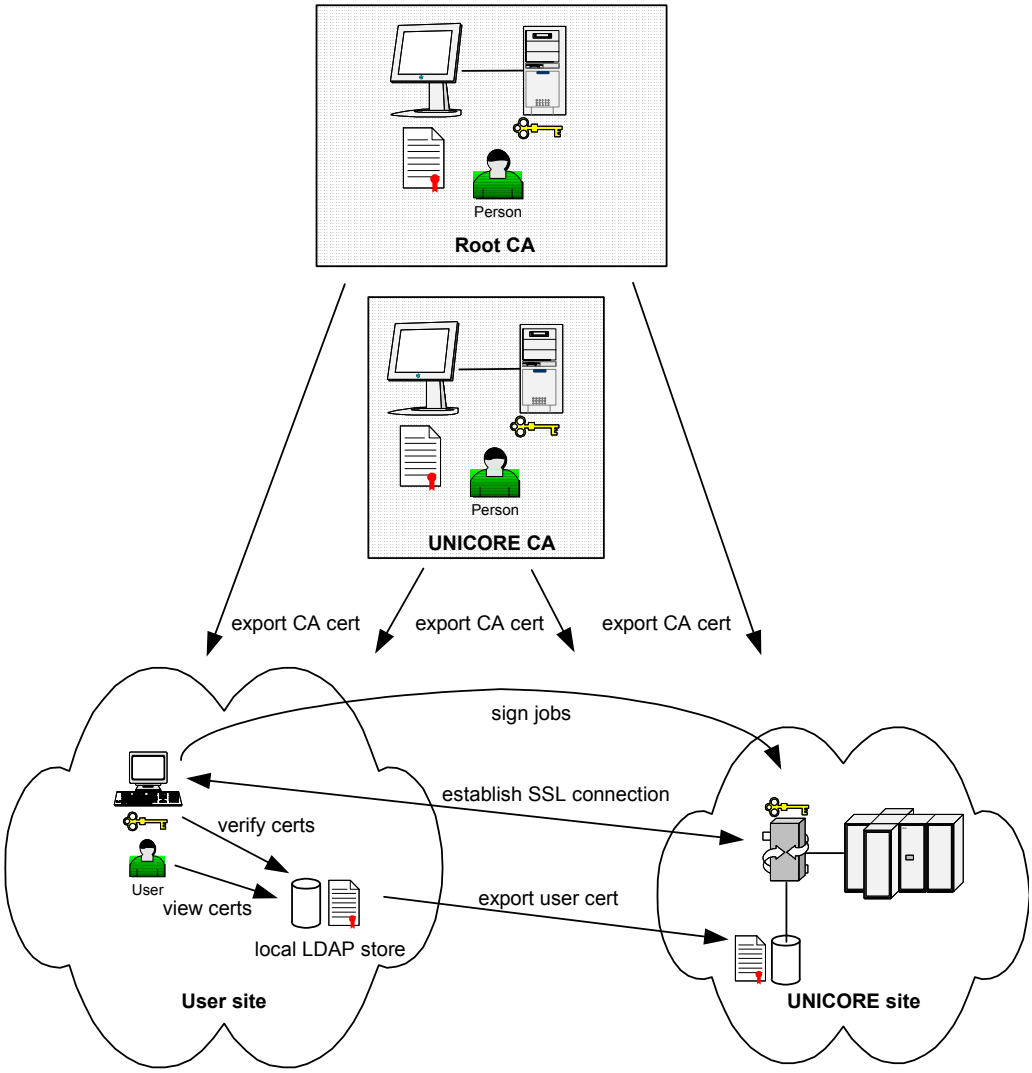

Figure 3

The local LDAP directory servers are optional. It is also possible to store the required certificates within the appropriate server and clients. However, a public directory server offers comfortable administration and access of certificates.

The following scenarios have to be considered:

1. *Establish an SSL connection between a user site and a UNICORE site (target site)*.
   The two involved sites mutually verify each other's certificates by verifying the certificates' signatures and the certificate chain.
   However, neither client nor UNICORE gateway verifies the identity of its respective counterpart. Instead, only the certificates' signatures are validated by the appropriate signer certificates. Client (user) identification and authorization is done by the NJS in a next step. The client has to rely on the authenticity of the gateway certificate's signature which it validates with the appropriate signer certificate (U-CA certificate).

2. *Endorse UNICORE jobs*.
   The user endorses a job by signing it with his private key. The signature can be verified using the user's certificate, which is passed as part of the job.
3. *Consign sub*-jobs.
   The NJS consigns sub-jobs for distributing them via SSL connections to corresponding V-Sites. To identify itself and to establish secure SSL connections to target sites the NJS uses its certificate similar to 1).

2.4 User Authorization

Before a user is allowed to submit AJOs to a target system he must be given access. Authentication and authorization take place at the target site:

1. *User authentication*
   To establish a connection to a target site via SSL, the UNICORE client has to present a valid client certificate to identify a UNICORE user, and the Gateway of the target site has to present its Gateway certificate to identify itself.
   As described above, UNICORE only makes use of signature validation during this step.

2. *User authorization*
   Before the user is allowed to submit a job to the target supercomputer she must be given authorization. So, the NJS of the target site validates the user's signature on the AJO against its local UNICORE user database (UUDB).
   If both the certificate and the certificate chain are valid and the user is registered with the UUDB, user authorization is successful.

A user is currently administered on two levels: first, he is given a user certificate to allow access to the UNICORE Grid, and secondly, his certificate is registered with the U-Sites granting access rights to him.

Recommendations for a future UNICORE PKI can be found in [Schu].

3 Analysis of the UNICORE Security Model

Overall security within UNICORE heavily depends on

- the security within the UNICORE PKI (CA security & RA authentication policy),
- the security of the private keystores within the user clients and servers and
- the diligence with which the individual certificates and certificate chains are validated before trust is granted.

The current PKI model is based upon a single central U-CA which signs the certificates of all UNICORE users.

This model was good for the project phase and is now subject to change as the German HPC centers set up a virtual organization for production.

3.1 Disadvantages of a Single U-CA vs. Multiple CAs

UNICORE is a distributed computing infrastructure adaptable to a manifold of target systems. Also, it is still growing and might have thousands of users across several countries in a few years. A centralized PKI with one central UNICORE CA would be overloaded within a short timeframe.
On the other hand, there are existing PKIs which already provide certificates to UNICORE partners for other purposes, e.g. secure communications via SSL, mail signing and Single-Sign-On. It is desired to re-use those PKIs to generate UNICORE certificates.

Furthermore, different user groups with distinct projects do not need to interwork very closely. Those considerations result into the concept of Virtual Organizations (VO).

3.1.1 Single U-CA

Use of UNICORE within the UNICORE Plus project is currently based upon a single U-CA which is signed by a Root CA. A single U-CA issues certificates for all UNICORE users, Gateways, NJS' and developers. By this means, there are neither interworking problems nor compatibility issues. All users and HPCs may work together, because they share common CAs.

This model is good for a limited number of users and HPCs. As soon as the number of users and/or HPCs increases, the load for the U-CA steps up, too. A higher U-CA load means:

- increasing delays in issuing certificates
- increasing number of RAs which condition a higher administrative load and possible security problems due to more frequent RA status changes (new RAs, diminishing RAs, changing RA representatives, etc.)
- in case the U-CA certificate expires or gets compromised (stolen private key) all subordinated certificates have to be exchanged against new ones. This would cause a total freeze of the whole UNICORE sphere.
- a single U-CA leaves no space for redundancy (no backup certificates from a separate U-CA).

3.1.2 Multiple U-CAs

UNICORE users might already base their other services like secure mail or Single-Sign-On on existing PKI infrastructures. In theory, existing user certificates might be re-used for UNICORE purposes. However, there are a couple of issues that have to be considered:

- PKI clusters have to be set-up. A cluster is a logical entity which consists of one CA, several RAs, a number of HPC centers and a group of users who run most of their applications on HPCs within the cluster. Disjunctive clusters may interwork by importing

each other's Root and U-CA certificates or by cross-signing those certificates on the CA level.

- Server certificates (Gateway certificates, NJS certificates) are still issued by the appropriate U-CA individually. They are not subject to re-usage.
- Re-using existing user certificates could be dangerous if the security requirements of the individual applications are different.
  In the current UNICORE environment all certificates comply with a high security level according to [Butl]. When re-using user certificates it has to be made sure that the existing certificates satisfy the appropriate security level.
- The security of the user certificate also depends on how safe the private key is stored on the client computer. So, it is not always possible to re-use an existing user certificate, especially if the private key has to be copied from a secure to a less secure keystore.
  A solution would be a common keystore for all applications running on the client computer which is accessed through a dedicated interface (see 3.2).

Nevertheless, multiple PKIs require that UNICORE partners who have a communications relation (user/server or server/server) need to import their partner's root and U-CA certificates (see Figure 4).

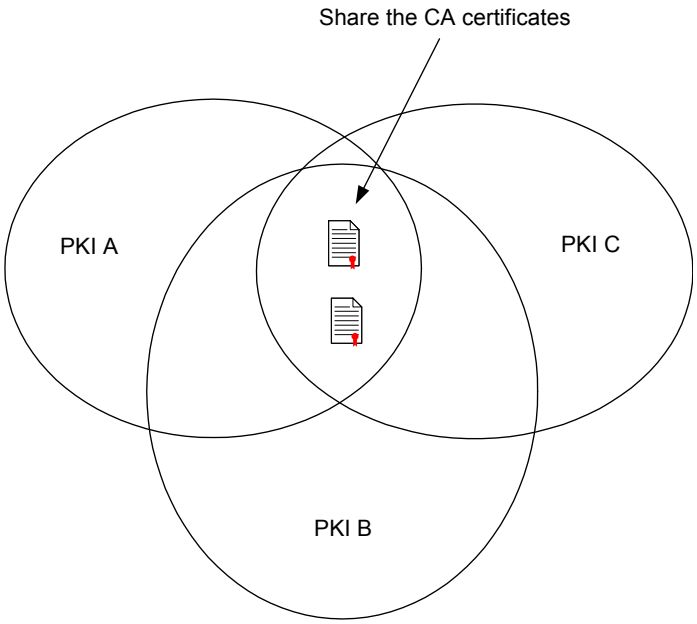

Figure 4

An alternative to sharing CA certificates is to cross-sign the different Root and UNICORE CAs. When using public CAs this may not always be possible, because the CAs are in charge of accepting or not accepting other CAs' certificates. So it is not advisable to rely on cross-signed CAs.

Instead of getting local copies of the different CA certificates it may be possible to access the appropriate LDAP servers of the required Root and UNICORE CAs, if the CAs provide such directory servers. This would also prevent having obsolete copies of expired CA certificates on local computers.

3.1.3 Summary

The current single CA environment has a couple of weaknesses which can be overcome with introducing multiple CAs:

- In a multi CA environment the load per UNICORE CA is reduced, so requests can be processed faster compared to a single CA environment.
- U-CAs have a much smaller set of registered U-RAs in a multi CA environment. The risk of being compromised by a bogus RA is reduced.
- If a central U-CA is compromised all UNICORE certificates become obsolete. In a distributed U-CA environment only a limited set of certificates have to be re-issued.
- If a central CA gets compromised the whole UNICORE Grid needs to be frozen until all certificates are replaced. This is not only a very expensive and long lasting process, but it also means that no-one within the UNICORE community could work. This would be a knock-out criteria for commercial, high availability applications.
  In a distributed environment normally only partial outages occur. For commercial and/or very important applications there could be backup certificates from a different U-CA, so that those jobs could be re-submitted immediately.
- A distributed CA infrastructure best matches the actual communication relations between partner sites within a UNICORE Grid. It is most likely that there will evolve groups whose members work closely together while there are only loose relationships between different groups. This concept is called “Virtual Organizations”. Members within closely related group should use the same U-CA.
- The GRIP project [GRIP] has been established to develop an interface between the two worlds of UNICORE and Globus. So, GRIP users need special certificates which they may access the Globus gateways with and which are valid within Globus.
  Optionally, GRIP may use distinct U-CAs issuing GRIP compliant UNICORE certificates with a
  - distinct policy which is easier to adapt to security requirements of Globus
  - limited set of certificates

  Currently, it is not necessary to use distinct U-CAs for GRIP because the security level of Globus is not higher than that of UNICORE.

Note that disjunctive UNICORE CAs can, but need not, share one single Root CA. A common Root CA does not avoid cross-signing or sharing of U-CA certificates.

3.2 Private Keystore

The most critical factor in running UNICORE with single or multiple CAs is the safety of the local keystore which holds the users private keys.

Assume the user runs three different applications, secure mail, VPN and UNICORE, all based upon certificates. If those applications cannot share a common keystore, it is recommended to use dedicated UNICORE certificates instead of re-using existing user certificates.

Having just one certificate for all applications, but no common keystore would mean to make copies of the certificate’s private key to three different keystores. The chance that the key gets compromised by security holes within the keystores grows with the number of keystores which corresponds to the number of copies of the private key.

A common keystore with a dedicated interface library would solve the problem of running multiple keystores with different security levels. All applications which should be accessible through Single-Sign-On must be provided with an interface to the appropriate library which grants access to the common private keystore.

The user would then be able to use one certificate for all different applications like secure mail, VPNs and UNICORE.

The UNICORE client is able to access a common JAVA- or PKCS#12-based keystore. The disadvantage of a PKCS#12-based keystore is that there is no enforcement to the pass phrase which encrypts the private keys within the store. So, a user may choose a bad pass phrase which may easily be compromised by a malicious hacker through a brute-force attack. Enforcing secure pass phrases with the UNICORE client has the potential to alienate users, since they would need to use the (more cumbersome) pass phrases for all applications using this keystore.

The JAVA-based keystore supports pass phrase enforcement.

UNICORE Gateways and NJS' do store certificates and private keys within PKCS#12-based keystores.

A future improvement to a common, software-based keystore is to store the private key on a smart card.

3.3 Certificate Chain Validation

The current UNICORE architecture implements a seamless check of the certificate chain. While importing the certificates into its keystore, the client checks each certificate whether it is

- not outdated,
- not listed in the appropriate CRL and
- issued by a trusted ("well-known") CA.

Certificate chain validation is terminated as soon as

- a known certificate is found (successful termination) or
- an invalid certificate is found within the chain (unsuccessful termination).

If there is an invalid certificate within the chain all subordinated certificates are considered invalid and have to be exchanged.

4 Review of the UNICORE CA Policy

The existing UNICORE CA policy is designed for a single CA issuing certificates for all UNICORE users, NJS' and gateways. It is adopted to the DFN-PCA [DFNP] and is described in [Boet2].

For a future PKI, especially when UNICORE is interfaced to Globus through the GRIP project, it is essential to adapt the UNICORE CA policy to the GGF certificate policy reference model as described in [Butl].

Contradictory certificate policies would make it difficult to guarantee a common level of security among the different Grid communities.

This chapter outlines important differences between the UNICORE CA policy and the GGF certificate policy reference model. This discussion is all about policy – technically, the UNICORE software can accommodate any X.509–compliant certificates.

4.1 Security Levels and User Identification

The UNICORE CA policy currently defines one level of security, while the GGF CP defines four different levels:

- Rudimentary
- Basic
- Medium
- High

The current UNICORE CA policy offers a high level of security according to the definitions made in [Butl]:

- The subscriber has to personally appear in front of the RA.
- The subscriber has to present a valid photo ID card.

Validation by video conference is currently used for identifying remote users to avoid travelling. It is in a testing phase and needs to be defined in a future UNICORE CA policy. In [Butl] user identification by videoconferencing is also mentioned as a possible alternative for in-person appearance.

It is recommended to adopt further authentication procedures, especially for those users whose organizations do not have own RAs. They should use the RAs of their partners' organizations, i.e. organizations where they want to run their applications.

The authors of this document suggest the following identification procedures for a future U-RA policy:

- well-known users (users who are personally known by the administrator of the RA):
  identification through telephone
- known-users (users who are already registered with the RA):
  identification through video conference, showing a valid photo ID card in front of the camera
- new users (users who are new to the RA):
  in-person identification requiring a valid photo ID card

Other identification methods as described in [Butl] may also be adopted, e.g. using valid signed certificates (certificates signed by a trusted CA) for online identifications of users.

A graded authentication policy reduces travelling and speeds up the process of issuing certificates compared to the current model.

The user's private key is stored in software, but should be stored in hardware according to [Butl].

4.2 Common Certification Policy Reference Model

A distributed Grid architecture consisting of multiple PKI clusters should base on a common CP reference model like the GGF CP reference model described in [Butl] to permit interoperability while keeping a common level of security.

The current UNICORE CA policy is based upon a single U-CA, so the policy applies for all UNICORE entities (users, Gateways, NJS' and developers).

In a multi PKI environment a common CP reference model must be the basis for choosing appropriate Root CAs and U-CAs. While the CP of an own CA can arbitrarily be adapted, the policies of the various commercial CAs normally cannot be modified to match a given reference model. While setting up an own CA means a high administrative load for the organization, contracting a commercial CA is far more effective.

A commercial CA already applies to certain security, legal and availability standards which may especially be important to commercial UNICORE users who demand a high level of security and availability.

Choosing a commercial CA (Root CA, U-CA) the following aspects have to be taken into account:

1. The CA shall be internationally known and accessible.
2. The CA shall be a "well-known", trusted CA which complies to certain security, legal and availability requirements[3] as outlined in [Butl]. As mentioned above, legal requirements may be of high importance especially for commercial users.
3. A UNICORE CA policy, which complies to the requirements of the GGF CP reference model, must be derived from the general policy issued by the CA. Especially, the security levels shall be equal or higher than those defined in [Butl].
4. The CA of choice must accept the subordinated RAs as specified by the contractual partner.
5. An RA policy, which complies to the requirements of the GGF CP reference model, must be derived from the general policy issued by the CA.
6. The CA must offer a signed CRL.

Figure 5 illustrates how a common certificate policy reference model would underlie the different grids ("intra-grid", i.e. within UNICORE, as well as "extra-grid", i.e. UNICORE and Globus).

A PKI within a Grid sphere bases upon a common CA and comprises all users, developers, and HPCs closely working together.

There may be individual users and HPCs who need to co-operate with other PKIs within the same Grid sphere. Those entities need to exchange resp. be able to access the appropriate certificates, namely the

- root certificate (if Root CAs are different and are not cross-signed) and
- U-CA certificate

of the remote PKI to successfully establish communications relations with each other.

It is also required that each PKI gets access to the other PKI's

- Root CA CRL,
- U-CA CRL and
- Directory Server containing the users' certificates.

Those requirements can easily be matched by using publicly accessible CAs. The common CP reference model ensures a homogenous certificate management (e.g. certificate

[3] The CA of choice for the Deutscher Wetterdienst is "TeleTrust", the CA of Deutsche Telekom.

revocation lists, unbroken certificate validation chain) and a common security level (level of trust, keystore safety).

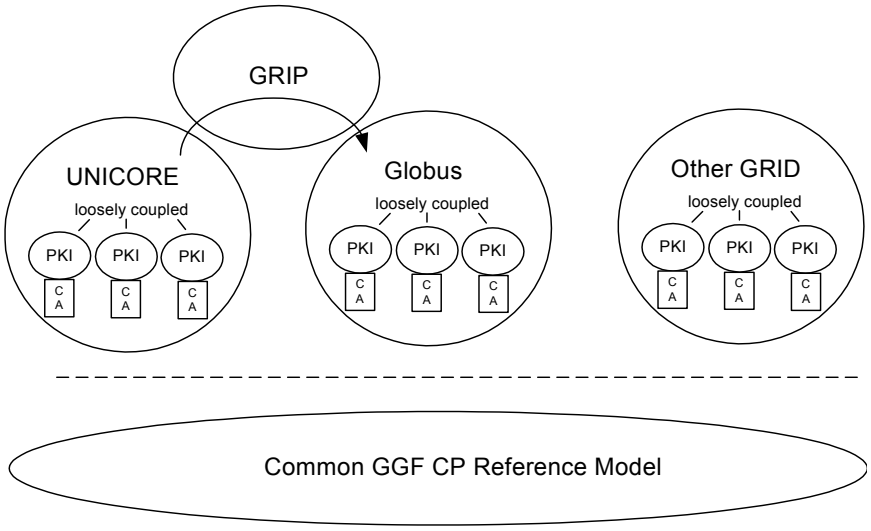

Figure 5

4.2.1 Security in Globus

Like UNICORE the Globus system also relies on a Public Key Infrastructure to authenticate users as well as HPC sites and to secure their communication relations.
However, Globus does not use any signed objects for transmission of the job, though the actual transfer is also performed via a SSL connection.
Most commands executed on Globus target systems use so called “proxy certificates” in order to simplify access and to implement a “single–sign–on” policy.
A Globus proxy certificate is a standard X.509v3 certificate. The Globus “proxy structure” contains the following items:

- An unencrypted private key (key protection is through file system permissions only)
- An X.509 certificate self signed by the user
- The user original certificate signed by the CA containing the users public key

They usually expire after a few hours (currently 12 hours) and are stored in the TMP directory of the users workstation. This has the drawback that if someone gets hold of the certificate containing the users private key he/she can submit jobs on behalf of that user. To avoid a permanent misuse by hackers getting hold of Globus’ sensible proxy certificates, they are only valid for a short period of time.

On the other hand, the use of proxy certificates allows support for highly dynamic jobs in Globus, like for instance the construction and submission of jobs from within a running Globus job. This is not possible with the stricter UNICORE security model described above.

In Globus those temporary “proxy” certificates are used to establish secure SSL connections between sites to distribute jobs. A Globus job is not signed, so a target system is not able to validate the Globus job itself.

The information about the user submitting a Globus job is only known by the Globus gateway creating proxy certificates after identifying the Globus user through his user certificate.

Thereafter a Globus job may be changed by any of the traversed gateways undetectable to the target system where the job shall be executed.

So, for a UNICORE HPC it would not be possible to

- clearly identify a Globus user who submits a job to UNICORE
- validate the integrity of a submitted job

4.2.2 UNICORE and GRIP

For GRIP, the UNICORE client uses a proxy–init plugin to generate a temporary Globus certificate basing upon a valid UNICORE client certificate. This temporary certificate is passed as part of the UNICORE through an unmodified gateway to the NJS server, which unpacks the temporary certificate and passes it to the TSI for interaction with Globus. So, GRIP does not require an own PKI, but relies on

- valid UNICORE certificates which have been checked against the appropriate CRLs
- valid certificate chain (valid U-CA certificate, valid root certificate)

For use in GRIP, the UNICORE security is not compromised for UNICORE sites; only Globus resources that (parts of) UNICORE jobs run on are accessed with the (weaker) Globus–specific mechanisms. In the current phase of GRIP development only the UNICORE to Globus gateway will be implemented. As the PKI of Globus, which makes use of proxy certificates as described above, is weaker than that of UNICORE, it is not feasible to build a bi-directional gateway maintaining UNICORE's high security standards.

5 Bibliography

[Boet1] Ernst Bötsch:
UNICORE Certification Authority (U-CA)
June 28th, 2000 (Version 1.3)
Leibnitz Rechenzentrum Munich

[Boet2] Ernst Bötsch:
UNICORE CA (U-CA) Policy – Zertifizierungs-Richtlinien für UNICORE.
February 1st, 2001 (Version 1.0)
Leibnitz Rechenzentrum Munich

[Schu] W.D. Schubring, H. Richter, C. Wimmer:
Analysis of Alternatives to the UNICORE PKI and Recommendations for its Future
August 5th, 2002
Leibnitz Rechenzentrum Munich

[Uproj] UNICORE Partners:
Joint Project Report for the BMBF Project: UNICORE Plus – Uniform Interface to Computing Resources
June 2002
UNICORE Forum e.V.

[Butl] Randy Butler, Tony Genovese:
Global Grid Forum Certificate Policy Model
September 2001
Global Grid Forum

[DFNP] http://www.pca.dfn.de

[UNICORE] http://www.unicore.org

[Globus] http://www.globus.org

[GRIP] http://www.grid-interoperability.org

[GGF] http://www.gridforum.org

6 Appendix A : Glossary

AJO	Abstract Job Object
CA	Certifying Authority
CP	Certificate Policy
CRL	Certificate Revocation List
CSR	Certificate Signing Request
DFN	Deutsches Forschungs-Netz e.V.
GCP WG	Grid Certificate Policy working group
GGF	Global Grid Forum
GRIP	Grid Interoperability Project
HPC	High Performance Computing Center
NJS	Network Job Supervisor
PKI	Public Key Infrastructure
RA	Registration Authority
SSL	Secure Socket Layer
U-CA	UNICORE CA
U-RA	UNICORE RA
U-Site	UNICORE-Site
UUDB	UNICORE User Database
V-Site	Virtual Sites (within a U-Site)

7 Appendix B : Security Considerations

7.1 Author Information

Torsten Goss-Walter	**Deutscher Wetterdienst Kaiserleistrasse 44 63067 Offenbach Germany**	**Torsten.Goss-Walter@dwd.de**
Reinhard Letz[4]	**Deutscher Wetterdienst Kaiserleistrasse 42 63067 Offenbach Germany**	**Reinhard.Letz@dwd.de**
Dr. Thomas Kentemich	**Pallas GmbH Hermülheimer Str.10 50321 Brühl Germany**	**Thomas.Kentemich@pallas.com**
Hans-Christian Hoppe	**Pallas GmbH Hermülheimer Str.10 50321 Brühl Germany**	**Hans-Christian.Hoppe@pallas.com**
Dr. David Snelling	**Fujitsu Laboratories of Europe Hayes, Middlesex, UB4 8FE United Kingdom**	**d.snelling@fle.fujitsu.com**
Philipp Wieder	**Forschungszentrum Jülich 52425 Jülich Germany**	**ph.wieder@fz-juelich.de**

7.2 Intellectual Property Statement

The GGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the GGF Secretariat.

The GGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the GGF Executive Director (see contacts information at GGF website).

7.3 Full Copyright Notice

[4] Reinhard Letz is no longer with Deutscher Wetterdienst.

8 Change History

Date	Comment	Author
09/16/02	First draft release	T. Goss-Walter/ R. Letz
09/19/02	First review	T. Kentemich, P. Wieder
09/20/02	Focus on UNICORE PKI, short introduction to GRIP	T. Goss-Walter
10/07/02	Paper reviewed by FECIT	D. Snelling
10/09/02	Paper reviewed by Pallas	Dr. T. Kentemich/ H.-C. Hoppe
10/21/02	Added some “formal stuff” (legal statement etc.)	T. Goss-Walter
10/23/02	Added more information on Globus security derived from WP 2.1 deliverable	T. Goss-Walter
03/27/03	Changed description of Proxy Certs according to Markus Lorch’s request	T. Goss-Walter
05/09/03	Changes applied according to M. Romberg’s remarks	T. Goss-Walter
07/09/03	Introductory chapter about UNICORE added	T. Goss-Walter