Project no. IST-033576

XtreemOS

Integrated Project
BUILDING AND PROMOTING A LINUX-BASED OPERATING SYSTEM TO SUPPORT VIRTUAL ORGANIZATIONS FOR NEXT GENERATION GRIDS

Methodology and Design Alternatives for Trust Services in XtreemOS
D3.5.9

Due date of deliverable: February 28^{th}, 2010
Actual submission date: March 24^{th}, 2010

Start date of project: June 1^{st} 2006

Type: Deliverable
WP number: WP3.5
Task number: T3.5.5

Responsible institution: STFC
Editor & and editor's address: Alvaro Arenas
STFC
Didcot, Oxfordshire
OX11 0QX, UK

Version 1.0 / Last edited by STFC / March 22^{nd}, 2010

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history:

Version	Date	Authors	Institution	Section affected, comments
0.0	25/01/10	Alvaro Arenas	STFC	first draft and table of contents
0.1	08/02/10	Benjamin Aziz	STFC	added the main trust model
0.2	09/02/10	Benjamin Aziz	STFC	added alternative trust models
0.3	17/02/10	Benjamin Aziz	STFC	added the section on introduction on trust
0.4	18/02/10	Zhouyi Zhou	ICT	added the isolation section
0.5	24/02/10	Benjamin Aziz	STFC	Revised the sections on the trust protocols
0.6	25/02/10	Ian Johnson	STFC	Section on Certificate Management
0.7	26/02/10	Benjamin Aziz	STFC	Added the Introduction and Conclusion
0.8	01/03/10	Ian Johnson	STFC	Added contents of RCA VO Attribute Certificate, corrected spelling etc
0.9	01/03/10	Alvaro Arenas	STFC	Final revision before sending to internal review
1.0	22/03/10	Alvaro Arenas, Benjamin Aziz and Ian Johnson	STFC	Applied corrections and feedback based on internal reviewers' comments

Reviewers:
Alvaro Martínez Reol and Philip Robinson

Tasks related to this deliverable:

Task No.	Task description	Partners involved°
T3.5.5	Trust management in Grid-based Operating Systems	STFC*, INRIA, XLAB
T3.5.13	Isolation	ICT*, INRIA, XLAB

°This task list may not be equivalent to the list of partners contributing as authors to the deliverable
*Task leader

Contents

Glossary

AMS	Account Mapping Service - maps XtreemOS GUIDs to UIDs/GIDs
CDA	Credential Distribution Authority
CA	Certification Authority
GID	Group Identifier (as used in Linux/UNIX)
GUID	Global User Identifier - unique identifier for an XtreemOS user
GVID	Global VO Identifier - unique identifier for an XtreemOS VO
RootCA	Root Certification Authority
PKC	Public Key Certificate
PKI	Public Key Infrastructure
RCA	Resource Certification Authority
RSS	Resource Selection Service - XtreemOS component used in selecting resources for job execution
UID	User Identifier (as used in Linux/UNIX)
UUID	Universally Unique Identifier [18]
VO	Virtual Organization
VOPS	Virtual Organization Policy Service
XtreemFS	The XtreemFS Filesystem
XVOMS	XtreemOS Virtual Organization Management Service

List of Figures

List of Tables

Executive Summary

This deliverable aims at providing a complete description of the techniques for trust management in XtreemOS. Our presentation of trust revolves around three main elements: the model of trust adopted in XtreemOS, the mechanisms of exchanging such trust and finally, trust-enhancement services.

The XtreemOS trust model gives a high-level overview of how the different organisations, users, resources and services are divided and managed in XtreemOS. The deliverable presents the main elements of the XtreemOS trust model: domains, certification authorities, credentials and trust protocols. It is also discussed possible alternative settings for the trust model, including the pros and cons of each setting.

The main trust-establishment mechanism in XtreemOS is credentials. By using credentials, such as digital certificates, an entity can convey certain attributes about itself to other entities for the purpose of enhancing the trustworthiness of the former. The deliverable describes the type of information carried in XtreemOS certificates and the main mechanisms for credential distribution.

Trust services permit the users of the XtreemOS operating system to have better faith in its dependability when handling their job submissions. Our main example of such services is the isolation mechanism, which allows users to run their jobs in different degrees of isolation from other jobs.

1 Introduction

This deliverable discusses the methodology and design alternatives for trust services in XtreemOS. Our presentation of trust revolves around three main elements: the model of trust adopted in XtreemOS, the mechanisms of exchanging such trust and finally, trust-enhancement services.

The first element is the trust model itself, which gives a high-level overview of how the different organisations, users, resources and core XtreemOS services are divided and managed. The main element of trust is the *domain*, which specifies the organisational boundaries of what can be trusted and where everything outside is not to be trusted by default unless trust is enhanced using mechanisms and services.

Credentials, such as digital certificates, are the main example of trust-establishment mechanisms in XtreemOS through which an entity (user, resource, service) can convey certain attributes about itself (such as its identity, features, context etc.) to other entities for the purpose of enhancing the trustworthiness of the former.

Finally, trust services permit the users of the XtreemOS operating system to have better faith in its dependability when handling their job submissions. Our main example of such services is the isolation mechanism, which allows users to run their jobs in different degrees of isolation from other jobs.

The rest of the deliverable is structured as follows: In Section 2, we introduce the XtreemOS trust model and discuss possible alternative settings for this model including the pros and cons of each setting. In Section 3, we discuss the format of the digital certificates used in XtreemOS. In Section 4, we give the description of the XtreemOS isolation service capabilities, its design and its application programming interface. Finally, in Section 5, we conclude the deliverable and give directions to future work.

2 XtreemOS Trust Model

This section gives a general and brief overview of the XtreemOS trust model, first discussed in [4], and used to integrate the different security and VO management services by setting-up trust between them. It also concerns the setting-up of trust between these services and users and resource providers.

2.1 Introduction

There have been many attempts in most fields of sciences and business domains to define the notion of trust in order to explain certain behaviours or models. For example, in the domain of social sciences, Gambetta [9] defines trust as the subjective probability by which an individual expects that some other individual will perform a given action or set of actions on which the former individual's overall welfare depends. On the other hand, in the economic domain, the European Commission Joint Research Centre defines trust as that property of business relationships, which allows reliance to be placed on the business partners and the business transactions developed among them [13].

One of the earliest attempts to give a computational definition of trust came from Marsh [15], based on Deutch's notes on trust [7], which state that trusting behaviour occurs when individuals perceive ambiguous (good or bad) paths whose end result is dependent on the actions of other persons, where bad results are more harming than good results are beneficial. If individuals choose to go down a path, they can be said to have made a trustful choice. This definition of paths resembles the notion of execution traces in language semantics and suggests that some degree of trust is needed in making trace choices.

Grandison and Sloman [11] provide a more recent definition of trust in the context of computer security. They regard trust as the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context. Trust is classified into several calsses among which are *service provision trust* and *certification trust*. Service provision trust denotes the reliance of a user on the functionality of a service, which is an essential aspect of Grid-based applications, whereas certification trust refers to trust built on a set certified attributes.

Josang et al. [14] make a difference between *reliability trust* as a subjective probability, perceived according to Gambetta's definition above, and *decision trust* as being the extent to which an entity is willing to depend on some other entity in a given situation or context with a feeling of relative security, even though negative consequences are possible.

Falcone and Castelfranchi [8] present a cognitive view of trust applied to the context of task delegation. When delegating a task, an entity might evaluate the trust it places on another entity considering the different beliefs it has about the

latter. These include: 1) competence beliefs, i.e. the delegatee is competent to do the task, 2) disposition beliefs, i.e. the delegatee actually will do the task, 3) dependence beliefs, i.e. the delegator entity believes that at least it is better to rely on the delegatee for the task than not to rely on it, 4) fulfillment beliefs, i.e. the delegator believing that the task can be done, 5) willingness beliefs: the delegatee intends to carry out the task, 6) persistence beliefs, i.e. the delegatee is stable enough with regards to its intentions, 7) self-confidence beliefs, i.e. the delegator should believe that the delegatee knows it can do the task and finally, 8) motivation beliefs, i.e. the delegatee has some motive to run or execute the task.

2.1.1 Trust in XtreemOS

In XtreemOS, the notion of trust that we adopt is based on three main aspects. First, trust is perceived as an administrative separation of resource and service ownership and management. Different organisations, resource owners, users and core XtreemOS service managers are allocated their own domains, which define their own boundaries of trust. This notion is based on the notion of service provision trust defined by Grandison and Sloman [11]. Second, trust is asserted in special tokens created based on cryptographic mechanisms such as digital certificates and other credentials, which are then verified by their consumers. These tokens convey verifiable attributes of entities that allow them to establish trust with other entities existent in other domains. This aspect is similar to certification trust as defined by Grandison and Sloman [11]. Finally, the transmission of trust tokens is achieved via trustworthy communication channels and protocols, which could be either offline possibly involving humans or cryptographic such as authentication protocols based on the Secure Sockets Layer (SSL) protocol [12].

In the following sections, we shall discuss trust domains and the different elements of the XtreemOS trust model.

2.2 Trust Domains

Trust domains refer to the separations in the ownership and management of software and hardware resources as well as human membership. Administrative domains are a type of trust domains. We use the term *trust domain* to indicate the level of assurance that each domain provides the designers, administrators and users of the XtreemOS operating system with.

Assuming that $S \rightarrow S'$ is an assurance ordering relation taken from some lattice of assurance levels (e.g. [6]) to indicate that S has a higher assurance level, and thereby is more trustworthy, than S', then Figure 1 illustrates this relation among the three main trust domains in XtreemOS, *Core Sites*, *Resource Sites* and *User Sites*.

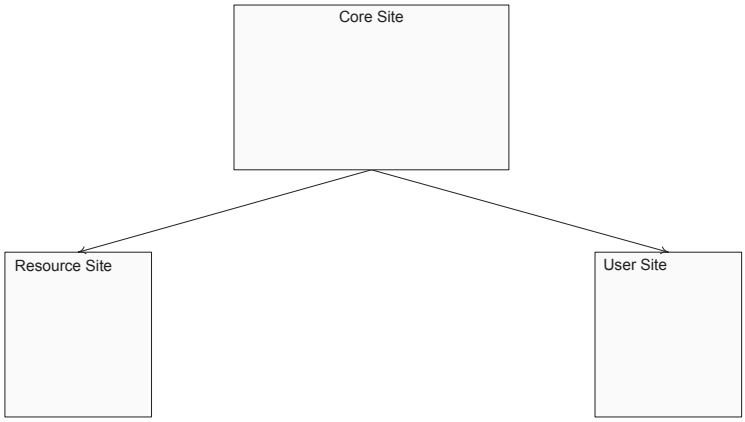

Figure 1: Ordering among the XtreemOS Trust Domains.

A VO is regarded as a logical domain, which comprises a single Core site and any number of Resource and User sites. In the following sections, we discuss each of these domains.

2.2.1 Core Site Domain

The *Core Site* domain is the domain in which all the core security and VO management services, as described in [3], are running. This implies that the site must be the most trusted site, and therefore, must have a high level of assurance. This domain may be split into two domains, as shown in Figure 2.

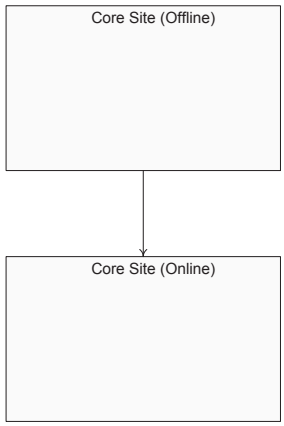

Figure 2: Online and Offline Core Site Domains.

- *Online Core Site Domain*: This domain represents the site in which any security and VO management service running in the online mode is hosted. This means that the domain is networked to other domains, however, it still

maintains high levels of assurance by adopting strong security protection measures.

- *Offline Core Site Domain*: This domain is an offline site in which the most sensitive element in the trust chain is running, i.e. the Root Certification Authority as will be explained later on in the section on trust model elements. This site has no network connections to any other sites, it utilised strict security measures and its services are only accessible via authorised administrators interactions. Therefore, the site is considered to have the highest assurance level among all other sites.

2.2.2 Resource Site Domain

The *Resource Site* domain represent any domain in which resources (machines, services, software) are hosted and are connected to the Grid, therefore making them available to any VOs formed out of the Grid. The level of assurance of a resource site cannot be guaranteed and hence they represent a lower trust level than the core site.

2.2.3 User Site Domain

The *User Site* domain is any domain hosting users of the Grid, which may apply to join VOs and avail of the VO resources. User sites have no guarantees regarding their assurance levels, therefore, the user site domain is considered to be lower in trust than the core site. Note that user and resource sites cannot be compared in their levels of assurance.

2.3 Elements of the Trust Model

We now turn our attention to the main elements constituting the XtreemOS trust model. These are shown in Figure 3. These elements can be classified into five categories: *Certification Authorities*, *Users*, *Resources*, *Credentials* and *Protocols*. We discuss these categories in the next sections.

2.3.1 Certification Authorities

Certification authorities represent points of trust from which users and resources can obtain credentials to certify their identities and/or their attributes. These constitute our definition of a Public Key Infrastructure (PKI). We define three such authorities: the *Root Certification Authority* (Root CA), the *Credential Distribution Authority* (CDA) and *Resource Certification Authority* (RCA). These authorities are organised in a hierarchy as shown in Figure 3, where trust is delegated from

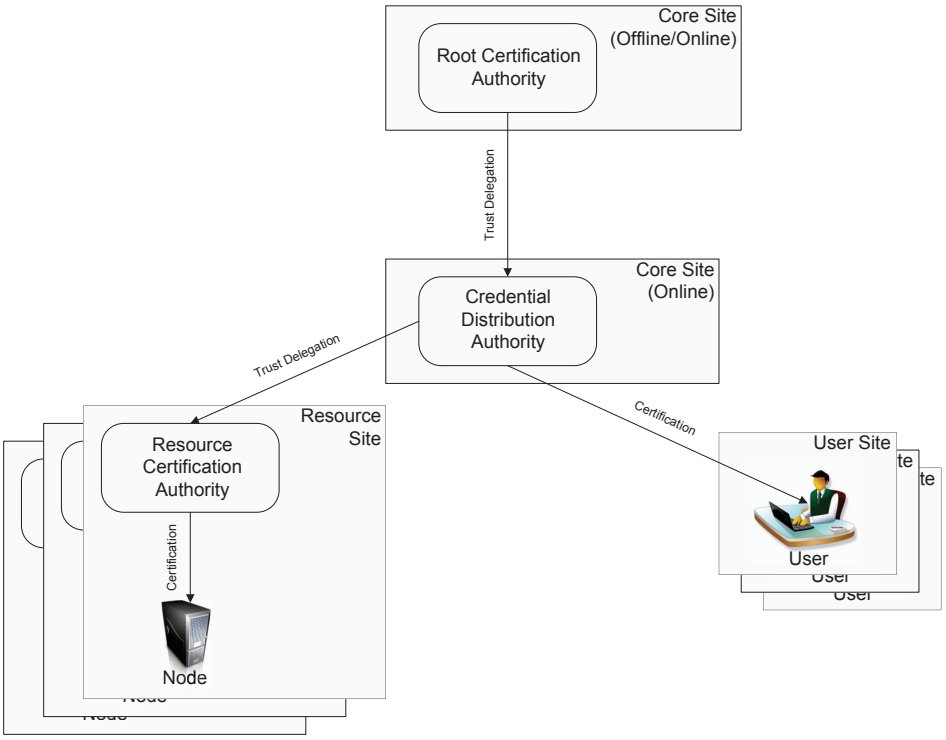

Figure 3: The XtreemOS Trust Model.

the Root CA to the CDA and then to the RCA. This trust delegation implies that the CDA has its public key signed by the Root CA and that the RCA has its public key signed by the CDA. The Root CA itself is a self-signing authority meaning that it will sign its own public key.

An important advantage of adopting separate authorities for users (CDAs) and resources (RCAs) is that one can achieve cleanly a separation of concerns between user credential management and resource credential management through the use of the CDA and RCA services. This design strategy ensures modularity and better representation of trust in the real world.

The Root Certification Authority. The Root CA is the trust anchor for any XtreemOS-based Grid system. The Root CA can be configured to issue identity certificates to the core services, or it may delegate this task to the CDA. The Root CA always delegates trust to the CDA for the purpose of certifying users and resource sites. This means that the Root CA does not itself certify any large-scale Grid system but rather provides a*trust anchor* for that system. Such an anchor of trust is necessary to bootstrap trust, either initially in the Grid or whenever any of the other services are compromised and their certificates need to be reissued. Without an anchor, it is not possible to (re)bootstrap trust in a distributed system

such as a Grid. Therefore, it is important to ensure that the Root CA is highly protected from unauthorised accesses and is running on a highly secured machine.

The Root CA operates in an offline mode. This means that any trust delegation (to the CDA) is carried out via offline means (e.g. emails, telephones, administrators direct access using command line programs). No network connection is provided to any services or programs running on different machines. This is considered to provide a higher assurance level than if the Root CA provided online access, which would increase the risk of the Root CA being compromised (e.g. by the theft of the Root CA private key).

The Certificate Distribution Authority. The CDA acts as a subordinate of the Root CA to which trust is delegated by the Root CA. This delegation of trust means that the CDA can certify the public keys of users and RCAs such that any entity consuming those certificates will be able to trace the chain of trust up to the Root CA. The CDA also can issue any attribute certificates to users to indicate their VO attributes such as their membership of particular VOs. The CDA serves several main purposes:

- It acts as the online frontend to the Root CA, if the Root CA is running in the offline mode. Therefore, one can achieve a separation of concerns between the management of the Root CA's security and its online functionality.
- The CDA permits the separation between user certification and resource site certification as represented by the RCA. It functions as a distribution authority separating between users and RCAs.
- The CDA acts as a trust domain within which VOs can be formed out of any users and RCAs that have been certified by a common CDA.

The Resource Certification Authority. The RCA is a subordinate of the CDA to which trust is delegated by the CDA for purpose of managing resource certification within the RCA's site. The RCA facilitates the following:

- The management of the resource certification within each site belonging to the Grid.
- The separation of concerns from the CDA, as the CDA does not have to manage the certification of each and every resource in the Grid.

In its own site, the RCA will certify the public keys of resources belong to that site. Additionally, it can also issue any attribute certificates required for those resources, such as certificates stating the storage capacity, speed of processor and assurance and QoS levels.

2.3.2 Credentials

Credentials are pieces of data held by the different actors that provide some form of information required by the capability that the actor requires to perform. Credentials have the general format <*Assertion, Proof, Validity*>, where the *Assertion* represents some attributes of the subject of the credential, *Proof* is verifiable information that the assertion is true and *Vailidity* are some conditions that render the credential itself valid.

Credentials may or may not have been created using cryptographic means. Passwords are examples of non-cryptographic credentials whereas digital certificates based on the X-509 standard [17] represent cryptographic credentials. In the following sections, we discuss three types of credentials used in XtreemOS.

Passwords. Passwords are the simplest form of credentials created by users and resource site administrators, e.g. during the initial request for registration phase (part of the Grid Management Capabilities) in a Grid from the Grid administrator. These passwords will be used in a password-based authentication protocol, such as the Secure Remote Password (SRP) [19], to establish secure (i.e. secretive and authenticated) communication channels.

A password has the format <*Assertion=username+password, Proof=(asserted password==stored password), Validity=password renewal*>. The proof is that the asserted password supplied by the entity being authenticated must be the same as the password stored by the system. The validty of this password is then determined by whether it has passed its renewal date/time or not.

Identity Certificates. These are digital certificates that enable their consumer to validate cryptographically the binding between the identity of the certificate's holder and its public key. This binding is important as it allows the consumer in the future to validate the authenticity of any information signed by the certificate holder. In XtreemOS, identity certificates are issued to any entity in the trust model - in fact, these are the most essential trust mechanism without which entities cannot participate in any VOs.

Identity certificates have the format <*Assertion=subject-has-identity, Proof=pubKey-of-issuer validates certificate signature, Validity=expiration-of-certificate*>. This format states that the assertion is that the subject has the identity in the certificates. The proof is that the public key of the issuer validates the signature on the certificate, and finally, the validity is the date/time expiration of the certificate.

Attribute Certificates. Technically, these are similar to the identity certificates except their purpose is different. Instead of binding the entity to its public key, the

attribute certificate has a field enumerating all the attributes of the entity. Attribute certificates are necessary in order to prove that entities has certain attributes. For example, users could be issued attribute certificates by the CDA to certify that they are members of certain VOs. RCAs also issue attribute certificates to resources to bind them to their functional and qualitative attributes.

Attribute certificates have the format, <*Assertion=subject-has-attribute, Proof=pubKey-of-issuer validates certificate signature, Validity=expiration-of-certificate*>. This is similar to the format of the identity certificates except that the assertion is related to some attribute(s) of the subject.

2.3.3 Users

Users are either humans or software that interact with the XtreemOS system and utilise the Grid resources within well-defined VOs.

2.3.4 Resources

These are the individual machines that offer computational and storage power to the Grid.

2.3.5 Protocols

Protocols refer to the sets of well-defined message-passing interactions used to carry certificates and other data messages among the certification authorities, users and resources in the model. In the context of the XtreemOS trust model, the main purpose of protocols, in addition to exchanging information, is to have a procedure for the establishment of trust among the different elements of the trust model. In the following paragraphs, we discuss a number of such protocols.

The XtreemOS User Registration Protocol. This protocol represents the entry point for users who wish to use the resources offered by a XtreemOS-enabled Grid. It is, in some sense, a pre-authentication step to what will follow. XtreemOS users will normally apply for an account in an XtreemOS Grid through a web interface called VOWeb. This allows the applicant to enter their account details (such as username and password), and contact details (such as organisation and e-mail address). The steps involved are:

- The user accesses the VOWeb registration page through their Web browser. To protect the confidentiality of user account details, this should only be accessed over an HTTPS connection secured with SSL. The user submits an application to join the Grid.

- The Grid administrator views the application and may wish to obtain more details about the user in order to verify the authenticity of their request. This communication between Grid administrator and the applicant may be offline in the form of email exchanges, phone calls, or even face-to-face meetings between the two parties.

- If the Grid administrator approves the request, the user is informed via email and can then start using their account in the Grid. Otherwise, the request is rejected and the user is notified.

The process above is similar to signing-up to conventional online services. The Grid administrator has the option, when considering a registration request, of applying a level of scrutiny to the registration applications appropriate to the level of security and assurance required in their Grid. Once the user has had their application approved, they can use the VOWeb application to join existing VOs or to create their own VOs. The user can then request an XOS-Certificate containing their VO attributes.

Obtaining a user XOS-Certificate. The main aim behind this process is to allow the users to be certified by the CDA, which will allow them to start creating VOs and to use VO resources. More concisely, it allows the users to enter the operational mode of the VO.

In this process, the user can obtain their XOS-Certificate from either the VOWeb application or from the command-line CDA client program. In both cases, the underlying protocol used is the same and takes place over an encrypted, authenticated channel obtained using SSL, where we denote by $[A \longrightarrow B]$ SSL-secured communications from A to B:

1. $[U \longrightarrow C]$: *username, password*
2. $[C \longrightarrow U]$: *status*
3. $[U \longrightarrow C]$: CSR_U
4. $[C \longrightarrow U]$: $(< \mathit{cert}_U >_{SK_C}, < \mathit{cert}_C >_{SK_R})$

In step 1, the user, U, sends their username and password to the CDA server, C. The CDA server then returns in step 2 the status corresponding to the authentication of the user. If this authentication status indicates that the user has been authenticated (step 2), then the protocol proceeds by sending a Certificate Signing Request (CSR_U) to the CDA server in step 3. This CSR contains the user's public key and request attributes, all signed with the user's private key. The CDA server authenticates the CSR by checking the signature, then creates an XOS certificate, which is essentially a X.509v3 certificate containing the user's public key and

their VO attributes in extension fields. In step 4, the CDA server sends back to the user a certificate chain ($< \textit{cert}_U >_{SK_C}, < \textit{cert}_C >_{SK_R}$) consisting of the XOS certificate of the user $\textit{cert}_U >_{SK_C}$ signed by the CDA's private key SK_C and the CDA's own certificate $< \textit{cert}_C >_{SK_R}$, signed by the Root CA's private key SK_R.

At the end of this protocol, the user can demonstrate in a verifiable manner its own identity, which will be trusted by any VOs that trust the Root CA.

The Resource Certificate Distribution Protocol. In the second protocol, the RCA, R, aims at obtaining a root certificate and identity certificate from CDA, C:

1. $[R \longrightarrow C]$: $\textit{CSR}_R$
2. $[C \longrightarrow R]$: $(< \textit{cert}_R >_{SK_C}, < \textit{cert}_C >_{SK_C})$

where $\textit{CSR}_R$ is a request from the RCA for the certificate signing by the CDA, $< \textit{cert}_R >_{SK_C}$ is the RCA's identity certificate signed by the private key of C, and $< \textit{cert}_C >_{SK_C}$ is a self-signed root certificate issued and signed by C.

The Protocol between Machines and RCAs. In general, machines need to register with at least one local RCA securely. Because machines are operated within the same administrative (trust) domain as their RCA, the problem of establishing a secure channel between a machine and its RCA is resolved locally within the domain. This will depend on the level of security and assurance adopted in the domain. Therefore, we do not describe here how a secure channel (if needed) is obtained between a machine and its RCA, since this is a local issue.

An Alternative Online Registration Protocol using password-based authentication via SRP. In many online registration systems, users access a web form to enter their account details (username/password) and contact details. To guarantee the confidentiality of their account details, the web form will use HTTPS. This entails the applicant having to trust and accept the SSL certificate presented by their web browser upon initiating the connection, unless the server certificate has been signed by one of the root certificates installed in their browser. The web browser can verify the certificate chain from the server SSL certificate to the trusted root certificate.

The options the user is faced with are summed-up as follows:

- They trust, without question, the validity and authenticity of the SSL server certificate. This is not an uncommon occurrence, but may make the user less confident in the overall registration process;

- They can import the SSL server certificate (or the certificate used to sign the SSL server certificate) into their browser. This raises the issue of how the user can obtain the certificates in the first place.

We can achieve the aim of reducing the certificate set-up overhead needed with conventional PKI-based approaches by giving the user a *certificate-less* method of securely registering their details. The alternative method relies on the user or the RCA administrator (i.e. the 'requester') accessing the registration manager over a confidential channel, which is based on an instance of the Secure Remote Password (SRP) protocol [19]. At this stage, the authenticity of the entities involved does not need to be guaranteed, as there is a conventional approval step following the registration whereby the identity of the requester and its attributes can be established. The registration client and manager can share the username/password for a pre-defined registration account to establish an SRP-based channel. The registration client can hide the *channel* username/password from the requester by automatically establishing a SRP connection when it is invoked, rather than the user needing to provide the username and password for the channel account. Once the SRP channel has been established, the requester can provide their account details (username and password), and their contact details. The registration manager stores this information in the registration database for approval by conventional (manual) means, which we do not describe here.

2.4 Design Alternatives for the Trust Model

In addition to the model presented in the previous section, we introduce in this subsection a set of alternative designs for the trust model.

2.4.1 Single Root CA with Multiple CDAs

A first alternative model is based on having multiple CDAs all running under the same Root CA as shown in Figure 4.

The main advantage of this model is that it permits the division of the Grid into trust domains each with its own CDA. Each domain will also be responsible for the certification of its own users, RCAs and resources, which facilitates further the scalability of trust. The failure of one CDA, for example as a result of compromising its private key, will not result in the failure of the certification of RCAs and users belonging to the other CDAs. Moreover, the model allows users belonging to one CDA domain to utilise resources certified by another domain, hence, catering for resource expansion.

Nonetheless, as shown in Figure 5, as a result of the presence of multiple CDAs, a man-in-the-middle attack can be mounted by a malicious user who's

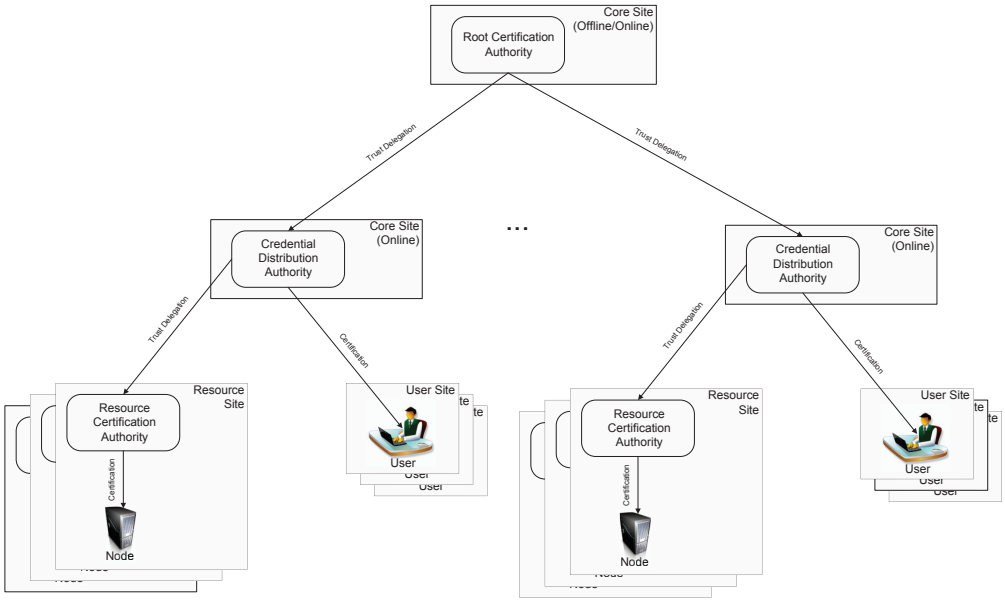

Figure 4: An Alternative XtreemOS Trust Model with multiple CDAs.

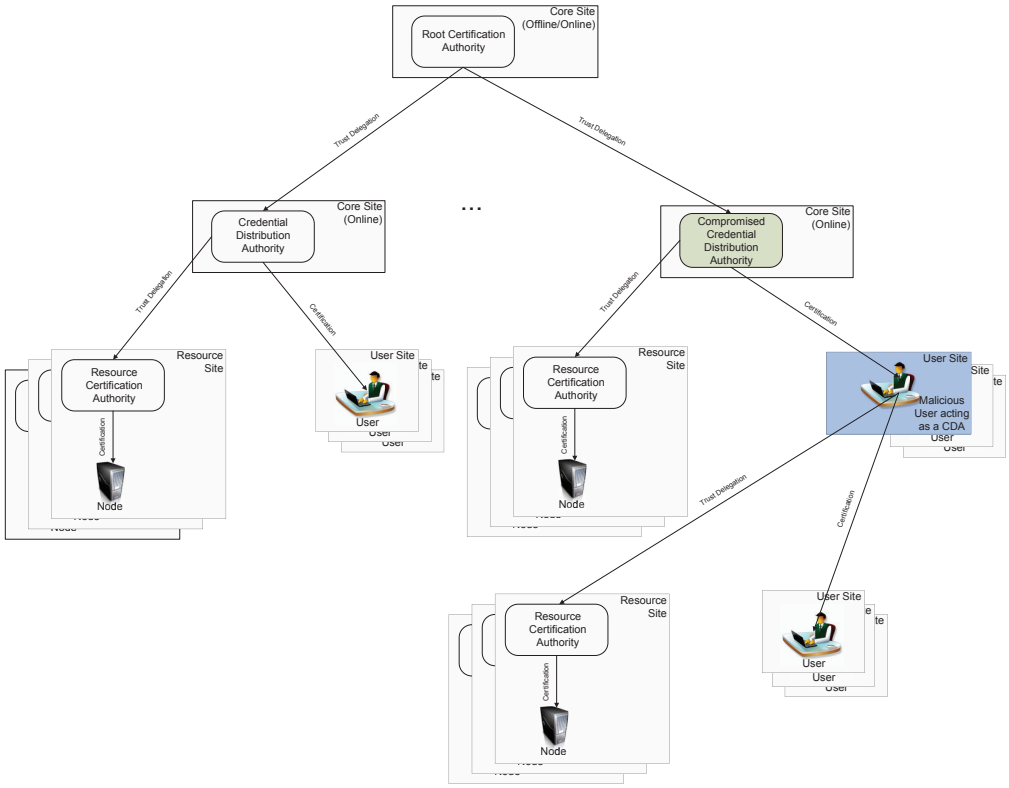

Figure 5: A Man-in-the-Middle Scenario on the multiple CDAs Model.

already compromised a CDA. In this attack, the malicious user will simply set-up and run a fake CDA using the stolen private key, which can then start certifying his own group of users and possibly other fake RCAs. This attack would not have been possible in the single CDA scenario since a second CDA would be disallowed to run.

2.4.2 Multiple Root CAs with Multiple Trust Delegations

The second alternative to the XtreemOS trust model is the one shown in Figure 6.

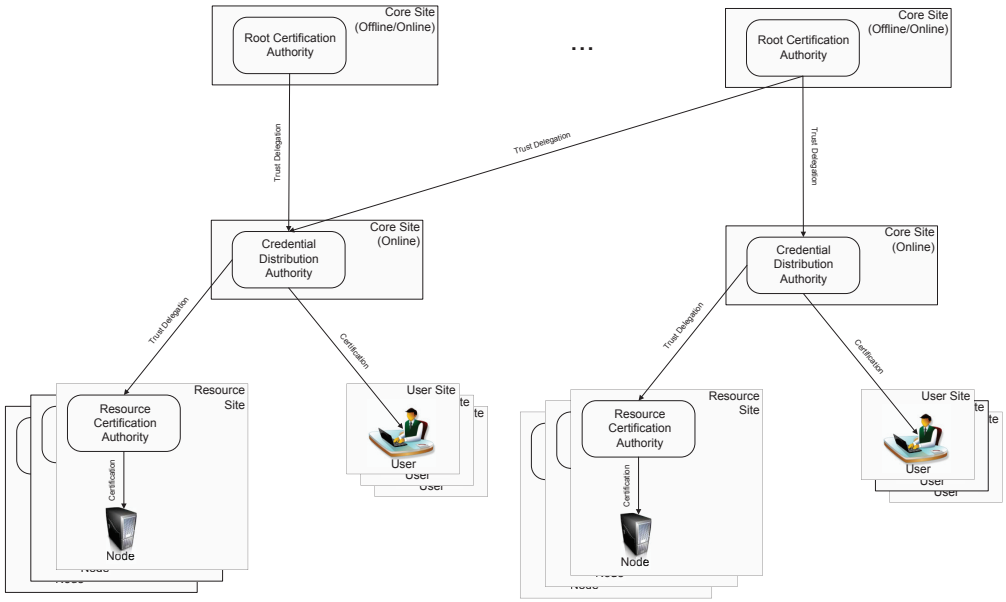

Figure 6: Another Alternative XtreemOS Trust Model with multiple Root CAs and Multiple Trust Delegations.

In this alternative model, the Grid itself is divided into multiple trust domains by running multiple Root CAs. Hence, there can be several roots of trust. This multiplicity at the root requires the definition of the relationship among the different trust domains. In this case, the model provides the possibility of one domain to trust the Root CA of another domain by simply obtaining a certificate for the CDA of one domain signed by the Root CA of the other domain.

This model has the advantage of providing more granularity at the Grid level regarding the management of trust, and at the same time, it is possible to allow users from one Grid domain to utilise resources from another domain. Another advantage also is that failure of one Root CA will not impact trust in the domain of other Root CAs.

Similar to the man-in-the-middle attack in the previous model, in this model, a malicious user who's able to compromise the Root CA of some Grid domain will be able to set-up a fake Root CA and hence, run masquerade the Grid domain belonging to that Root CA, as is shown in Figure 7.

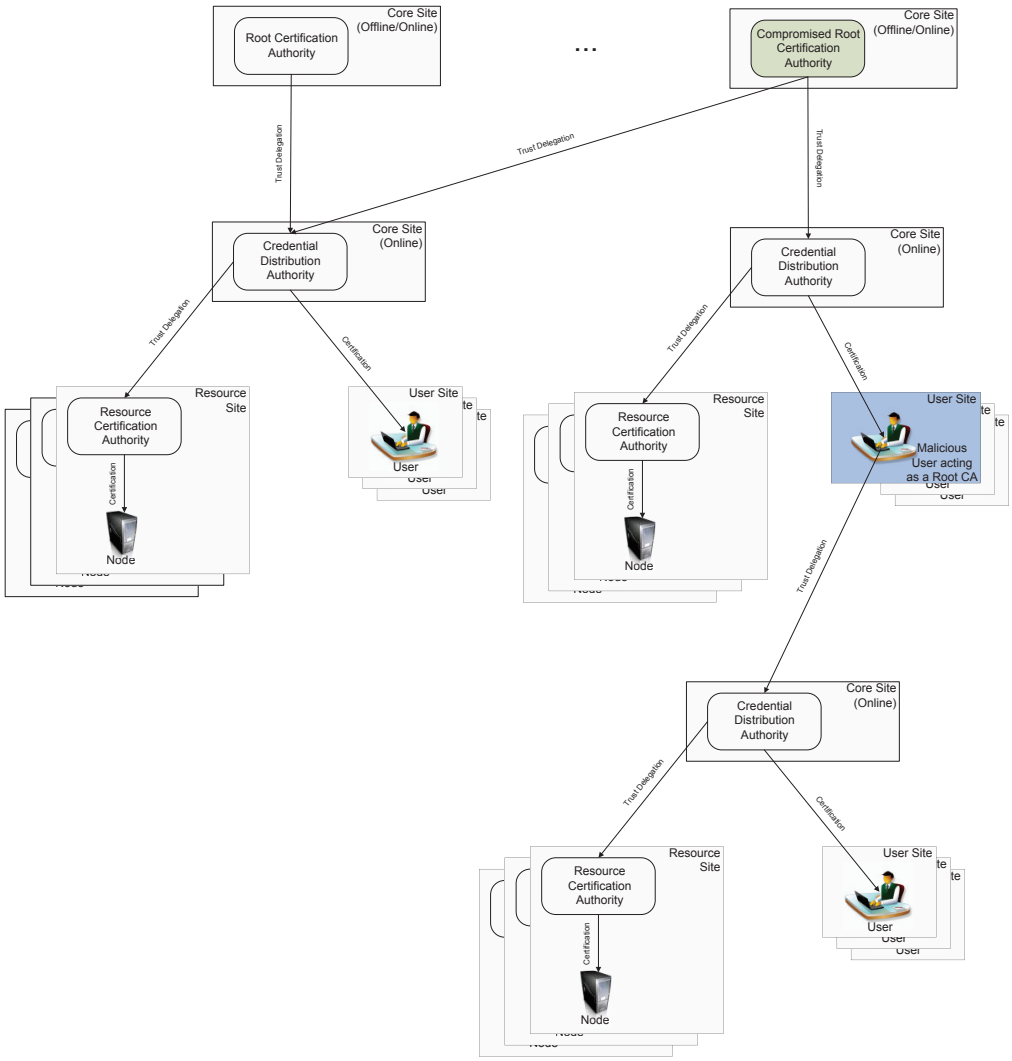

Figure 7: The Man-in-the-Middle Scenario in the Second Alternative XtreemOS Trust Model.

2.4.3 Multiple Root CAs with Cross Certifications

The last alternative model we discuss here is one in which there are multiple Root CAs, which are cross-certified as shown in Figure 8.

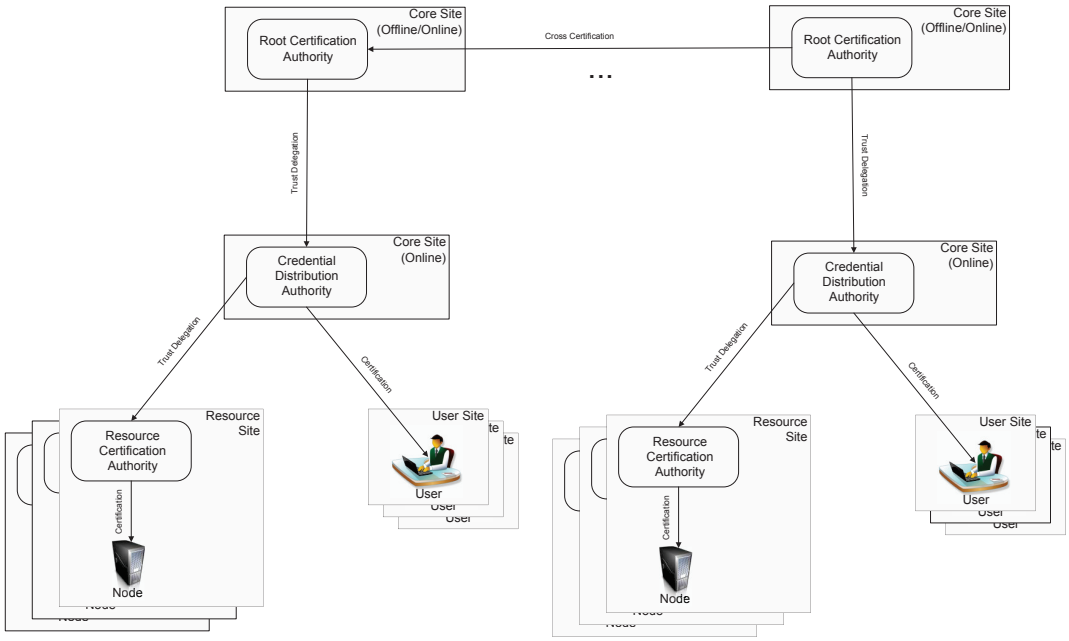

Figure 8: Another Alternative to the XtreemOS Trust Model with Multiple Root CAs with Cross Certification.

Cross-certification here means that each Root CA will certify (by signing the public key of) any other Root CAs it trusts. Therefore, users (resources) from the trust domain of the latter Root CA will be trusted by resources (users) from the domain of the former, simply because it will be possible to create a chain of certificates leading to the latter Root CA from entities of the domain of the former. This has the advantage in that one domain can be expanded to include other domains thus for example, achieving resource scalability.

Similar to the model of the previous section, this model two has the same man-in-the-middle vulnerability. Additionally, as noted in [1], cross-certification can have negative implications in the compatibility of levels of assurance in the cross-certified domains. For example, a “high level of assurance" classification may have different meanings in each domain thus leading to the possibility of reducing the assurance levels of some domains.

2.5 Comparison of Different Trust Models

The above mentioned alternatives for the XtreemOS trust model present varying degrees of complexity, flexibility and decentralisation of trust bootstrapping, establishment and management. In this section, we define a number of common criteria to compare these alternatives:

- Degree of Administration: This factor denotes the number of administrators

requried to bootstrap the trust model to a working Grid. The more the number of administrators required, the more expensive the model is regarding time and resources.

- Domain Multiplicity: This factor refers to the multiplicity of the trust domains established in each model within a single Grid. These include the numbers of user, ressource and core sites as well as VO domains and ultimately, the number of Grids that can be set-up.
- Maintenance Complexity: This factor denotes the maintenance complexity of each model. This includes the complexity of setting-up trust
- Decentralisation of Attack Points: This factor refers to the decentralisation degree of attacks on each model.

According to these criteria, Table 1 shows how each model behaves.

Model	Administration	Domain Multiplicity	Maintenance Complexity	Attack Decentralisation
Single Root CA/ Single CDA	Low	Low	High	Low
Single Root CA/ Multiple CDAs	Medium	Medium	Medium	Medium
Multiple Root CAs with Multiple Delegations	High	High	Low	High
Multiple Root CAs with Cross Certitification	High	High	Low	High

Table 1: Comparison of XtreemOS Trust Models.

3 Certificate Management in XtreemOS

In this section, we describe the types of certificates used in XtreemOS, their purpose, and which components and services create and consume them.

3.1 Introduction

XtreemOS uses the well-established X.509 certificate format, part of the standard for Public Key Infrastructure. The Public Key Certificate (PKC) for an entity carries the public key for that entity and can be used at various points in an XtreemOS system. The private key corresponding to the entity's public key must be kept secured and not disclosed to other entities.

3.2 Compliance with Standards and Recommendations

To enable interoperability between XtreemOS components using different implementations of the TLS/SSL protocol, and to assist interoperability between XtreemOS and Grid middleware systems such as gLite[10], the transport mechanisms and format of security credentials use well-established Internet standards and the most relevant sections of the OGF Grid Certificate Profile[5]. Communications interfaces to the most critical security services use a text-based protocol which is published to allow alternative implementations.

3.3 Certificate Lifecycle

Certificates are issued by a central entity, the Root CDA[1]. Certificates have a validity period set by the interval between the values set in the certificate fields "notBefore" and "notAfter". By contrast, the corresponding private key does not expire. This allows the user to use an existing private key when requesting a new public key certificate (avoiding the needed to generate a new keypair and set a new passphrase).

User certificates are consumed by XOS-SSHD, where the XOS-Certificate is used in mapping the user's GUID to a local Unix UID/GID on the remote host. User certificates are also consumed by the AEM Job Manager. A certificate may be revoked at any point during its validity period, which means that its status is set to not valid. This may be necessary. The change in the certificate's status can be made available to XtreemOS nodes that wish to check the validity of certificates.

[1] We use the term *root CDA* whenever the CDA is running an online root CA functionality.

3.4 Types of information carried in certificates

All certificates in XtreemOS carry at least some form of identity information, in addition, some other XtreemOS certificates carry additional attributes about the entity concerned. Identity information is used, along with a signature verification on the certificate, to check the authenticity of an entity. Attribute information is used to check that the entity is authorised to carry out a certain operation.

XtreemOS services use the certificate header field "Subject" to carry information about the identity of the entity. Additional information may be carried in standard certificate extension fields, such as "KeyUsage" and "subjectAlternativeName" fields. Attributes are carried in XtreemOS-specific extension fields (for example, the user's VO attributes). The XtreemOS-specific certificate extensions are readable by common tools such as OpenSSL.

Identity Identity certificates for services carry details such as the address of the node running the server, and the service type (e.g. CDA, VOPS, RCA). The identity certificate for the user (the XOS-Certificate) carries the user's identity in both the standard "Subject" certificate header field, and in the Global User Identifier (GUID) certificate extension field.

Attributes The user's VO attributes (such as GUID and VO membership) are carried in certificate extension fields as simple, comma-separated string values. Attribute certificates are used by the Resource Certification Authority (RCA) to describe the resources available on a particular resource node.

Certificate Types and Usage XtreemOS uses the X.509 v3 format. The table below lists, for each type of certificate, the issuer of the certificate and its purpose. The XtreemOS components which inspect the certificate, and the key usage associated with the certificate, are also described. The following abbreviations are used:

AuthC Authenticate an entity (prove their identity)

AuthZ Authorise an entity (used to determine which actions the entity can perform)

CDA The Credential Distribution Authority - distributes user certificates and service identity certificates

CRL Certificate Revocation List - a static list of certificates which have been revoked

Cert Sign Use the corresponding private key to sign a certificate (used by a CA entity)

CRL Sign Use the corresponding private key to sign a CRL (used by a CA entity)

Dig Sig Use the corresponding private key to provide a digital signature on data

RCA The Resource Certification Authority - distributes certificates for resource nodes

Root CDA The Root CA if this is enabled to operate offline, or the CDA if this is enabled to handle all certificate requests in online mode

SSL Client Use the certificate to authenticate the client side of a TLS/SSL connection

SSL Server Use the certificate to authenticate the server side of a TLS/SSL connection

VOPS VO Policy Service

vNode set The nodes constituting the Virtual Node replica set

The different certificates used in XtreemOS are shown in Table 2.

Type	Issued By	Purpose	Inspected By	Key Usage
Root Certificate	Root CDA	Root of trust	All Components	Cert Sign, CRL Sign
XOS-Certificate	CDA	AuthC/AuthZ user	XOS-SSH, Job Manager	Dig Sig, SSL Client
Service Identity Certificate	CDA	AuthC service	Clients connecting to services	Dig Sig, SSL Server, (RCA) Cert Sign
RSS Certificate	CDA	AuthC VO member	Nodes in RSS overlay	Dig Sig
Virtual Node Certificate	CDA	AuthC vNode set	vNode set members	Dig Sig
RCA Resource Certificate	RCA	AuthC RCA node	AEM	SSL Client/Server
RCA VO Attribute Certificate	RCA	AuthZ resource is in VO	VOPS	SSL Client/Server

Table 2: XtreemOS Certificates.

3.5 Credential Distribution

Credential distribution consists of the following steps:

- A public/private keypair is generated on the local node;
- The public key is sent in a Certificate Signing Request to an XtreemOS Certificate Authority (either the RootCA, the CDA, or the RCA);
- The XtreemOS CA signs the CSR, producing a public key certificate which is returned to the local node.

These steps occur at various stages during the creation and operation of an XtreemOS Grid.

For setting up core services, XtreemOS offers the Grid Administrator the choice of using an offline Root Certificate Authority (Root CA), which can be on a machine physically isolated from the network (hence not open to compromise); this offers the highest level of trust in the Root CA credentials, but requires out-of-band communication (e.g. email transfer of Certificate Signing Requests to the operator of the Root CA). Alternatively, the Root CA function can be delivered by the CDA server, thus providing an online Root CA. All other certificate requests, apart from those relating to resources, are handled automatically by the CDA server without the need for human intervention.

3.5.1 Credential Distribution for the Grid-wide Core Services

The XtreemOS core services, such as RCA, VOPS, XtreemFS, will need to be set up (at least minimally) before the Grid can start operation. (Additional core services such as extra RCA servers can also be set up during the operation of a Grid). This setup is accomplished by Grid Administrator role. The pre-cursor to setting up core services is to set up the Root CDA, which is then used to distribute credentials for the other core services.

3.5.2 Credential Distribution for the VO-specific Resource Selection Service (RSS)

An instance of the RSS is needed for each VO. This setup is accomplished by an Administrator who is either an Owner or Administrator for the specific VO. This Administrator uses the machine VO attribute certificate, issued by the RCA for this specific VO, to set up this instance of the RSS.

3.5.3 Credential Distribution for the Virtual Node replica set

In a similar fashion, the Administrator configuring up a set of modes to act as Virtual Nodes obtains a Virtual Node certificate, containing the DNS or IP address for each node, along with a Universally Unique Identifier (UUID). This is then supplied to the AEM Job Manager along with a JSDL file specifying the creation of a Virtual Node replica set. The Administrator uses the command "get-vnode-cert" to obtain a vNode certificate from the CDA.

3.5.4 Obtaining User Credentials

The user can obtain an XOS-Certificate after they have registered with the XtreemOS Grid (normally via the VOLife web interface).

The method of obtaining the user XOS-Certificate is to run a command-line tool, the CDA client program "get-xos-cert". This communicates with the CDA server to request a certificate containing the user's public key, their identity (GUID), and their other VO attributes. The user needs to authenticate themselves by providing a valid username/password pair for a valid account in the Grid registration database. By default, the "get-xos-cert' command places the credentials in system-defined default location in the user's filestore where other XtreemOS commands expect to find them.

This creates a private key for the user (or re-uses an existing private key for efficiency), and requests the CDA to sign a certificate request containing the user's public key.

3.5.5 Obtaining Service Credentials

Obtaining service credentials can be accomplished in one of two ways. If the Root CDA is configured to generate service identity certificates, the "get-service-cert" command can be used by a user with Administrator privileges (having the "Admin" role) to obtain a certificate directly from the CDA server, with the user being able to specify the output locations on the command line.

If the Root CDA is configured to operate in the manual, offline mode, the "get-service-cert" command creates a Certificate Signing Request which needs to be sent by e-mail to the operator of the Root CDA, In this case, the operator of the Root CDA will need to authenticate the request by means such as contacting the sender of the request by out-of-band means, such as e-mail or telephone. The operator of the Root CDA will send the signed certificate back to the requestor by email, who will have to manually install the certificate and private key. Locations of the credentials, and the private key pass-phrase, need to be specified in

the properties file for the service concerned.

RSS Certificates contain a field "RSS-VO:" specifying the GVID of the VO for which the RSS is being set up. An RSS certificate can be requested by the owner or administrator of this VO.

Virtual Node certificates contain a list of node addresses where members of the Virtual Node replica set will run.

Resource certificates are obtained by a series of interactions between the resource administrator operating on a resource node, and the RCA administrator operating the RCA server. The steps involved in getting a resource certificate for a resource node, then adding the resource node to a VO and obtaining a VO attribute certificate, are described in [3], pp84.

3.6 Example Certificates

In this section, we show the typical contents of a service identity certificate, a user XOS-Certificate, a RCA VO Attribute certificate, and the fields specific to an RSS certificate and a vNode certificate. The fields used in these certificates are shown in Table 3.

Field Name	Header/Std/XtreemOS extension	Field type
Signature algorithm	Header	String specifying cipher and length
Issuer	Header	String specifying DN of this certificate's issuer
Validity	Header	Dates of certificate validity
Subject	Header	String specifying DN of this certificate's subject
Basic Constraints	Standard Extension	Boolean: true implies this certificate can sign other certificates
Key Usage	Standard Extension	Set of usages for which the corresponding private key can be used
Extended Key Usage	Standard Extension	Additional usage for the corresponding private key
Subject Key Identifier	Standard Extension	A hash of the corresponding private key
Authority Information Access	Standard Extension	(Optional) Methods of finding certificate revocation info
Certificate Usage	XtreemOS Extension	Purpose of this certificate
GlobalPrimaryVOName	XtreemOS Extension	Global Unique Identifier for the user's primary VO
GlobalSecondaryVONames	XtreemOS Extension	Global Unique Identifiers for the user's secondary VOs
GlobalUserID	XtreemOS Extension	User's Unique Identifier
GlobalPrimaryGroupName	XtreemOS Extension	Unique Identifier for the user's primary group
GlobalSecondaryGroupNames	XtreemOS Extension	Unique Identifier's for the user's secondary groups
Role	XtreemOS Extension	User's Role
Group	XtreemOS Extension	User's Group
Subgoup	XtreemOS Extension	User's Subgroup
Owner-Of	XtreemOS Extension	List of VOs owned by the user
Admin-Of	XtreemOS Extension	List of VOs the user can administrate

Table 3: XOS certificate Fields.

Service Identity Certificate Contents. In the certificate shown below, the address of the node runnning the service is encoded into the Subject field, and the subjectAlternativName DNSname field.

```
     Issuer: CN=XtreemOS Testbed CA, O=XtreemOS Project,
     OU=Permananent Testbed Root CA
     Validity
         Not Before: Jul  1 08:22:05 2009 GMT
         Not After : Jul  1 08:22:05 2010 GMT
     Subject: O=Permanent Testbed CDA, OU=cda,
     CN=xtreemos-a.esc.rl.ac.uk/cda
     Subject Public Key Info:
         Public Key Algorithm: rsaEncryption
         RSA Public Key: (1024 bit)
         Modulus (1024 bit):
...
         Exponent: 65537 (0x10001)
     X509v3 extensions:
         Netscape Comment:
             XtreemOS Certificate
         X509v3 Subject Key Identifier:
             E5:DE:70:FA:59:56:B6:A7:64:7A:61:FA:
             B4:BE:D9:F9:B4:51:D1:3D
         X509v3 Authority Key Identifier:
             DirName:/CN=XtreemOS Testbed CA/
             O=XtreemOS Project/OU=Permananent Testbed Root CA
             serial:E4:12:B5:BC:33:29:6B:3B

         X509v3 Basic Constraints:
             CA:TRUE
         X509v3 Subject Alternative Name:
             DNS:xtreemos-a.esc.rl.ac.uk
         X509v3 Key Usage:
             Digital Signature, Key Encipherment,
             Certificate Sign
 Signature Algorithm: sha1WithRSAEncryption

   ...
```

Figure 9: Service Identity Certificate

XOS-Certificate contents. The most important elements of a user XOS-Certificate can be displayed with the "view-xos-cert" command. The example below is for a user who owns the VO with Global VO Identifier "2c0e8cb2-4453-46fe-85b7-74874e76e7c2". The user is an Administrator for the VOs "4c0e8cb2-4453-46fe-

85b7-74874e76e7d0” and
“5c0e8cb2-4453-46fe-85b7-74874e76e709”.

```
Issuer: O=Permanent Testbed CDA,
OU=cda, CN=xtreemos-a.esc.rl.ac.uk/cda
Validity
   Not Before: Feb  1 20:53:26 2010 GMT
   Not After : Feb 1 21:03:26 2011 GMT
 Subject: CN=ea9a7366-e34f-4a99-9e31-277430366475
 X509v3 extensions:
        X509v3 Basic Constraints: critical
            CA:FALSE
        X509v3 Key Usage: critical
            Digital Signature, Key Encipherment
        X509v3 Extended Key Usage: critical
            TLS Web Client Authentication
        X509v3 Subject Key Identifier:
            2C:07:20:F1:FD:7F:FB:DA:61:F6:B2:46:
            1B:13:F9:12:FA:45:59:A4
XtreemOS VO Attributes:
        GlobalPrimaryVOName:
            2c0e8cb2-4453-46fe-85b7-74874e76e7c2
        GlobalSecondaryVONames:
            2c0e8cb2-4453-46fe-85b7-74874e76e7c2
        GlobalUserID:
            ea9a7366-e34f-4a99-9e31-277430366475
        GlobalPrimaryGroupName:
            ae88816f-9f5c-48f9-ad7d-f71a64977904
        GlobalSecondaryGroupNames:
            null
        Role:
            null
        Group:
            group1
        Subgroup:
            null
        Owner-Of:
            2c0e8cb2-4453-46fe-85b7-74874e76e7c2
        Admin-Of:
            4c0e8cb2-4453-46fe-85b7-74874e76e7d0,
            5c0e8cb2-4453-46fe-85b7-74874e76e709
```

Figure 10: User XOS-Certificate

RCA VO Attribute Certificate Contents. The VO Attribute Certificate shown below is for a resource added to the VO with GVID "2c0e8cb2-4453-46fe-85b7-74874e76e7c2".

```
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O=XLAB, OU=rca, CN=core.xlab.si/rca
        Validity
            Not Before: Jan 29 15:05:49 2010 GMT
            Not After : Feb 28 15:15:49 2010 GMT
        Subject: C=SL, L=Ljubljana, OU=Research, O=XLAB,
        CN=Address = [://172.16.117.14:60000(172.16.117.14)]
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment,
                Data Encipherment
            X509v3 Extended Key Usage: critical
                2.5.29.37.0
            X509v3 Authority Key Identifier:
                keyid:E8:2D:8A:5E:73:1F:94:ED:1
                8:E3:F8:68:47:8E:A3:05:80:97:41:9F
                DirName:/CN=xtreemos-a.esc.rl.ac.uk/cda,
                O=Permanent Testbed CDA, OU=cda,
            CPUSpeed:
              3.605004288E9
            CPUCount:
              1
            MemorySize:
              3.06184192E8
            Services:
eu.xtreemos.system.communication.proxy.ServiceCallProxy,
eu.xtreemos.system.communication.redirector.ServiceCallRedirector,
eu.xtreemos.xosd.daemon.Daemon,
eu.xtreemos.xosd.security.rca.server.RCAServer,
eu.xtreemos.xosd.jobmng.JobMng,
eu.xtreemos.xosd.resourcemonitor.ResourceMonitor,
eu.xtreemos.xosd.resallocator.ResAllocator,
eu.xtreemos.xosd.security.vops.VOPS,
eu.xtreemos.xosd.resmng.ResMng,
eu.xtreemos.xosd.daemon.Daemon,
            VO:
              2c0e8cb2-4453-46fe-85b7-74874e76e7c2
```

Figure 11: Machine VO Attribute Certificate

4 Isolation

The XtreemOS isolation mechanism allows fine-grained control of user applications on resource nodes as well as the enforcement of resource quotas (memory, disk, cpu, network, etc.). Having such mechanism in place will increase the trust on a XtreemOS system. On one side, resource providers are ensured that users' application will be restricted to policies defined at the resource side. On the other side, VO users are ensured that jobs running on containers at the resource side are isolated from other users.

As presented in [2], isolation is usually defined along three axes: security isolation, resource isolation, and fault isolation.

- **Security isolation**. Containers or sandboxes enable isolated execution environments for VO users. Running a VO user's job in a container assures that no other users on the same node can see what they are doing, or compromise information.

- **Resource isolation**. Each container is assigned a specified allocation of CPU, memory, and disk, according to VO policies set on a VO user.

- **Fault isolation**. A fault or a process in one container does not adversely affect processes running in other containers. The container is also independent from the physical hardware, making it possible to move (i.e. checkpoint/restart) a container from one physical node to another with no downtime.

Design and implementation of XtreemOS isolation is based on virtualisation technology by exploiting OS level resource containers to create isolated execution environments for VO users. High-level design of isolation in XtreemOS was presented in deliverable D2.1.5 [2]. We introduce here main isolation capabilities and describe the application programme interface.

XtreemOS provides support for resource isolation based on Linux Control Group (`cgroup`) framework. As mentioned in Linux kernel document, to tracking resource usage of the system, one must require the basic notion of a grouping/partitioning of processes, with newly forked processes ending in the same group (`cgroup`) as their parent process. Within a `cgroup`, one can provide resource limit to that group. The total resource usage of all processes contained in the group. Fig.12 shows the high view on resource isolation in XtreemOS.

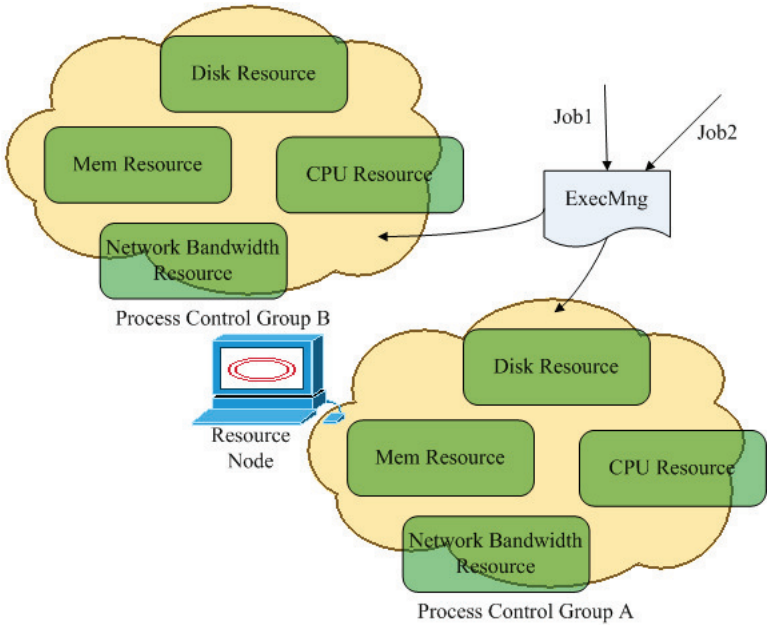

Figure 12: Resource Isolation: High View

4.1 Isolation Capabilities

OS level quotation set up

Goal

Linux kernel's cgroup framework provides fundamental resource isolation support. The types of resource are: disk, memory , CPU sets and network bandwidth. AMS sub system directly operate on OS's cgroup filesystem to setup the quotation for processes.

Actors

AMS sub-system.

Interface Used

Standard system interface to mount a cgroup system, write to a file in cgroup filesystem.

Success Scenario

The resource quotation is set for the process executing a job.

Pre-conditions

The quotation is not set for target process.

Post-conditions
The quotation is set for target process, and corresponding cgroup directory is established

OS level quotation remove
Goal
AMS sub system directly operate on OS's cgroup filesystem to remove the quotation settings for a process.

Actors
AMS sub-system.

Interface Used
Standard system interface to write a file in cgroup filesystem.

Success Scenario
The resource quotation is removed for the process executing a job.

Pre-conditions
The quotation is set for target process.

Post-conditions
The quotation is not set for target process, and corresponding cgroup directory is removed.

OS level quotation adjustment
Goal
AMS sub system directly operate on OS's cgroup filesystem to adjust the quotation settings for a process.

Actors
AMS sub-system.

Interface Used
Standard system interface to write a file in cgroup filesystem.

Success Scenario
The resource quotation is adjusted for the process executing a job.

Pre-conditions

The quotation is set for target process, and the corresponding cgroup directory is established.

Post-conditions

The quotation is set for target process, and the corresponding cgroup directory is established.

Job's quota set by ExecMng

Goal

JobMng (or a service on behalf of JobMng) extracts the quota-related information from the JSDL and XACML (policy from VOPS) and provides it to ExecMng, so that it can establish the isolated session with proper parameters. ExecMng establish (through AMS) the isolated session with corresponding quota set.

Actors

ExecMng

Interface Used

ExecMng establish the isolated session by calling AMS's interface function xpamexecvp in function executeJob. The parameter for xpamexecvp has data type "struct jobInfo". Quota information will be contained in this parameter.

Success Scenario

The resource quotation is set for the process executing a job.

Pre-conditions

Post-conditions

The quotation is set for target process, and the corresponding cgroup directory is established.

Job's quota remove by ExecMng

Goal

When the job has quit, job's quota settings is removed.

Actors

ExecMng

Interface Used

ExecMng call pam_close_session in function xpamexecvp when job exits.

Success Scenario
The resource quotation is removed for the process executing a job.

Pre-conditions
The quotation is set for target process, and the corresponding cgroup directory is established.

Post-conditions
The quotation is not set for target process, and corresponding cgroup directory is removed.

Job's quota adjustment by ExecMng
Goal
Dynamically adjust job's quota settings.

Actors
ExecMng

Interface Used
ExecMng call function setJobQuota with proper parameters.

Success Scenario
The resource quotation is adjusted for the process.

Pre-conditions
The quotation is set for target process, and the corresponding cgroup directory is established.

Post-conditions The quotation is set for target process, and the corresponding cgroup directory is established.

4.2 Design of Isolation Components in XtreemOS

Isolation in OS level is implemented by means of Linux cgroup framework.

To interact with cgroup framework, one must mount a file system of type "cgroup". When mounting such file system, users may specify a comma-separated list of subsystems to mount as the filesystem mount options.

For example, the following commands mount a cgroup hierarchy with resource type cpuset and memory in directory /cgroups/job1:

```
#mkdir -p /cgroups
```

```
#mount -t cgroup job1 /cgroups -o memory,cpuset
```

The user may add a process to the above cgroup hierachy by appending thus process's pid to job1's tasks file:

```
#echo $pid > /cgroups/job1/tasks
```

Then the user can can alter the memory limit of above cgroup hierachy by:

```
# echo 4M > /cgroups/job1/memory.limit_in_bytes
```

As mentioned in previous section, AMS subsystem directly interact with OS for upper layers of XtreemOS. ExecMng directly interact with AMS subsystem on behave of AEM. In current development cycle, the JobMng (or a service on behalf of JobMng) extracts the quota-related information from the JSDL and XACML (policy from VOPS) and provides it to ExecMng, ExecMng invoke AMS subsystem with job parameter which include quotation information, AMS set the quotation for the process that run the job.

4.3 Application Programming Interface for Isolation

From the ExecMng point of view, all the other existing interfaces needn't be changed except that in order to set isolation information, except following program interface should be called when isolation setting for a VO changed.

Function name setJobQuota(unsigned char *jobcert, struct jobQuota *jobquo)
Parameters

- jobcert: pointer to a buffer contains a job's x509 certificate.
- jobquo: jobquo has following structure (defined in "XPamAPIs.h"),

```
/** type of quota */
enum XPLY_QT {
  XPLY_QT_ABS,    /**< ABS - absolute value */
  XPLY_QT_REL     /**< REL - relative value */
};

/** type of resource for enforcement */
enum XPLY_REST {
  XPLY_REST_CPU,    /**< CPU - CPU resource */
  XPLY_REST_MEM,    /**< MEM - Memory resource */
  XPLY_REST_DISK,   /**< DISK - Disk resource */
  XPLY_REST_NETWORK, /**< NETWORK - network bandwidth */
  XPLY_REST_NPROC, /**< NPROC - maximun number of open process */
  XPLY_REST_NOFILE /**< NOFILE - maximum number of open files */
```

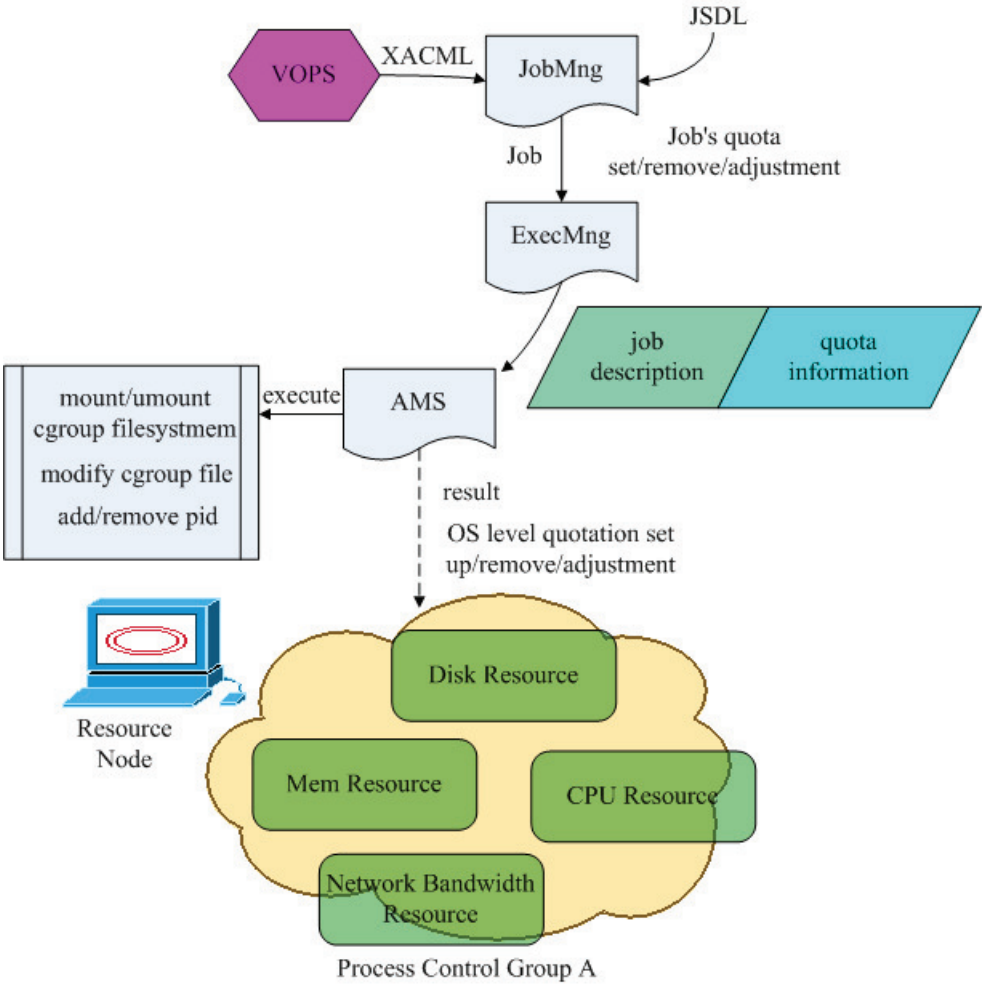

Figure 13: Isolation Components

```
};
struct jobQuota {
  enum XPLY_REST restype;
  enum XPLY_QT quo_type;
  int quo_val;
};
```

use case
Limit the VO's disk usage to 10M:

```
struct jobQuota jobquota2 = {
   XPLY_REST_DISK,
   XPLY_QT_ABS,
   101
 };

 if(setJobQuota(cert_buffer, &jobquota)) {
   fprintf (stderr,"Can not set job quota");
   return (void*)-1;
 }
```

5 Conclusion and Future Work

In this deliverable, we provided an overview of the XtreemOS trust model and the different alternative settings of this model, their pros and cons. We also discussed the formats of digital certificates used to exchange entity attributes, such as identities and features, among the users, resources and services in XtreemOS, and gave examples of such certificates. Finally, we defined the capabilities, design and application programming interface of the XtreemOS isolation service, which is an example of a trust service that enforces non-interference among user jobs to varying degrees.

One of the future uses of the XtreemOS operating system that is envisaged is in the area of Cloud computing [16]. This new playground for XtreemOS will pose new challenges to the trust model, mechanisms and services, which will provide directions for future work. Among these will be the challenge of managing credential in federated Clouds and ensuring clean isolation among users and resources belonging to different Clouds.

References

[1] Canadian Institute for Health Information. Cross-Certification Guidelines: National PKI Framework for Health, 2001.

[2] XtreemOS Consortium. Design and implementation of advanced node-level vo support mechanisms. In *XtreemOS public deliverables - D2.1.5*. Work Package 2.1, Dec 2008.

[3] XtreemOS Consortium. Fourth specification, design and architecture of the security and vo management services. In *XtreemOS public deliverables - D3.5.13*. Work Package 3.5, December 2009.

[4] XtreemOS Consortium. Third specification of security services. In *XtreemOS public deliverables - D3.5.11*. Work Package 3.5, January 2009.

[5] J. Jensen D. Groep, M. Helm. Gfd-c.125- grid certificate profile. OGF, March 2008.

[6] Dorothy Denning. A lattice model of secure information flow. *ACM Transactions on Programming Languages and Systems*, 19(5):236–243, May 1976.

[7] M. Deutch. Cooperation and trust: Some theoretical notes. In *Nebraska Symposium on Motivation*, pages 275–319. Nebraska University Press, 1962.

[8] R. Falcone and C. Castelfranci. Social trust: A cognitive approach. In Cristiano Castelfranchi and Yao-Hua Tan, editors, *Trust and Deception in Virtual Societibes*, pages 55–90. Kluwer Academic Publishers, 2001.

[9] Diego Gambetta. *Trust: Making and Breaking Cooperative Relations*, chapter Can We Trust Trust?, pages 213–237. Department of Sociology, University of Oxford, 1988. http://www.sociology.ox.ac.uk/papers/gambetta213-237.pdf.

[10] gLite Project Team. Wikipedia entry on gLite project, 2000. http://en.wikipedia.org/wiki/GLite.

[11] T. Grandison and M. Sloman. A Survey of Trust in Internet Applications. *IEEE Communications Surveys and Tutorials*, 3(4), September 2000.

[12] Transport Layer Security Working Group. The ssl protocol version 3.0, November 1996.

[13] S. Jones and P. Morris. Trust-ec: requirements for trust and confidence in e-commerce. Technical Report EUR 18749 EN, European Commission Joint Research Centre, 1999.

[14] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, March 2007.

[15] S. Marsh. *Formalizing Trust as a Computational Concept*. PhD thesis, University of Stirling, 1994.

[16] Christine Morin, Yvon Jégou, Jérôme Gallard, and Pierre Riteau. Clouds, a new playground for the xtreemos grid operating system. *Parallel Processing Letters (PPL)*, 19(3):435–449, 2009.

[17] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Rfc 3820 - internet x.509 public key infrastructure (pki) proxy certificate profile, June 2004.

[18] wikipedia. Wikipedia entry on Universally Unique Identifier, 2000. http://en.wikipedia.org/wiki/Universally_Unique_Identifier.

[19] Thomas D. Wu. The secure remote password protocol. In *Proceedings of the Network and Distributed System Security Symposium*, San Diego, California, USA, 1998. The Internet Society.