Project no. IST-033576

XtreemOS

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

Evaluation of Security
D3.5.10

Due date of deliverable: November 30^{th}, 2008
Actual submission date: January 14^{th}, 2009

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

Type: Deliverable
WP number: WP3.5
Task number: T3.5.10

Responsible institution: SAP
Editor & and editor's address: Philip Robinson
SAP Research
TEIC Building, University of Ulster
BT37 0QB Belfast
Northern Ireland

Version 1.0 / Last edited by Philip Robinson / January 14^{th}, 2009

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
PU	Public	$\surd$
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	21/11/08	Philip Robinson	SAP	first draft of structure
0.1	12/01/09	Philip Robinson	SAP	release of unfinished draft for reviewer feedback
0.2	13/01/09	Philip Robinson	SAP	incorporation of feedback from WP3.5 members
0.3	14/01/09	Philip Robinson	SAP	incorporation of official review feedback
1.0	14/01/09	Philip Robinson	SAP	final

Reviewers:
Michael Schoettner (DUS) and Haiyan Yu (ICT)

Tasks related to this deliverable:

Task No.	Task description	Partners involved°
T3.5.10	Evaluation of XtreemOS security	SAP

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

Executive Summary

What is the value of XtreemOS security? Where is the innovation? These are questions that have been frequently asked internally and externally. This document seeks to answer these questions by presenting the results of an evaluation. The evaluation primarily checks the fulfillment of cross-application requirements from WP4.2, as these have been selected across various application scenarios. The requirements are consolidated and presented as security requirements models. The evaluation checks if there exist the relevant concepts and mechanisms in the XtreemOS security specification and release in order to implement each security requirements model.

Secondly, management is an important capability for large scale systems. Included in the goals of proper management is security. XtreemOS is proposed as an alternative to traditional Grid middleware and hence security in Grid middleware. One of the advantages the XtreemOS is expected to show over Grid middleware is in the area of management as the amount of software layers becomes reduced and only well-known tools are used for administration. In addition XtreemOS security borrows and depends on some preexisting Grid security mechanisms, but focuses on the integration with the OS and the automation of management as the key values. For each of the security requirements models, the implementation in XtreemOS is assessed from the perspective of management in the evaluation summary when applicable.

By combining a basic security assessment with a management-oriented assessment, it is intended that clearer answers to the innovation of XtreemOS and XtreemOS security become more evident for users of XtreemOS. This should also be useful for prioritisation of development effort and future exploitation of XtreemOS.

Contents

Glossary

The glossary has been inherited from D3.5.11, such that there is no need to constantly cross reference.

AEM	Application Execution Management
CDA	Credential Distribution Authority
CA	Certification Authority
GGID	Global Group IDentifier
GUID	Global User IDentifier
GVID	Global VO IDentifier
NLP	Node Level Policy
PDP	Policy Decision Point
PKI	Public Key Infrastructure
TCB	Trust Computing Base
VOM	Virtual Organization Management
VOPS	Virtual Organization Policy Service
XtreemFS	XtreemOS File System
XOSD	XtreemOS Daemon

List of Figures

List of Tables

1 Introduction

Security is an important factor in the acceptance of a technology to be used for integrating computational resources across domains. The standard requirements are authentication, access control, integrity of communications and confidentiality of communications of users, data and machines distributed across the various domains involved. However, the security protocols and architectures for cross-domain interactions are today well established, such that the critical factor is to validate that these solutions are correctly implemented and that the technical choices for implementation add value to the environment for which they are intended. One factor is the negative impact on performance of systems that security might introduce. Another issue comes with the management of security, as well as the additional effort required to manage secure systems. This deliverable considers these factors in the same evaluation framework.

XtreemOS is an operating system especially designed for *Grid systems*. A Grid system is considered to be a network of distributed computers, with different local users and administrators, that allow various distributed software components and instances to share their resources and be accessed by different remote users and administrators. Installation of large-scale applications in a Grid systems distributed across different administrative domains is a challenge for collaborating organisations in the area of E-Science, as well as for datacenters that host applications for several customers. Data-centers today already manage thousands of IT resources, including servers, storage and networks, and constantly seek ways of reducing management costs and increasing resource utilization. These become more critical as business applications are more extensively deployed in these execution environments. For this reason, it seems likely that there can be significant transfers of knowledge and technology from the computational grids domain into better management (deployment and migration) of large-scale business applications. Computational Grids enable the sharing, selection, and aggregation of distributed computational resources (such as supercomputers, compute clusters, storage systems, data sources, instruments, people) and presents them as a 1 single, unified resource for solving large-scale compute and data intensive computing applications[8, 10].

1.1 General Challenges for XtreemOS Security

To better understand the challenges for XtreemOS security, with respect to value, the analysis is done within the domain of business applications. Picture a network of 1000's of servers and workstations, each having 1000's of different users and 100's of different administrators, as opposed to a sparse network of a few supercomputers and specialist users. For a typical business application the execu-

tion environment consists of these large number of dedicated application servers, each consisting of multiple application modules. Each of these applications may then require multiple deployment environments for development, quality assurance/testing, and production, each having different levels of access, load balancing and fault tolerance. Furthermore, the applications consists of many tiers, where each tier may itself have a dedicated server. The large amount of servers results in increased complexity, decentralization and cost of management, including time and costs to keep the various systems running, up-to-date and consistent. Because individual servers are typically configured to handle peak workloads, they are not actually utilized to their fullest capacities. Security should serve to support ready accessibility to servers as opposed to making it harder to achieve resource sharing.

Firstly, data owned by a single organisation may be distributed across server nodes belonging to various other organisations with their own internal policies and administration. **Access controls** mechanisms are required that locally enforce global access policies (such as those determined within a Virtual Organisation (VO)). In addition to access control, it should be possible to **validate the integrity of stored data** and detect incidences of tampering.

Secondly, business applications consists of a set of data objects and processes, where users manipulate processes to perform actions that read, write, delete or modify data objects. Processes may then be composed of several threads that execute in parallel on the operating system. Processes may be distributed across organisational domains and need to communicate for the purpose of the application logic or for coordination. Communications needs to be **confidential** and maintain **integrity** without the introduction of unacceptable latencies.

Thirdly, **identity and membership management** are necessary for globally-distributed systems from a usability perspective for users, but also to protect the resources of providers. Given the scales mentioned above, users should not have to constantly key passphrases or manage 1000s of certificates for each node they have access to in the Grid system.

Fourthly, **resource management** including accounting becomes an increasingly critical component for business applications, or in cases where the deployment of a Grid system is for the purpose of generating business. Users must be assured that the resources they use are "genuine" and that they can have the expected guarantees. Meanwhile resource providers need to be protected against malicious users bent on abusing their resource access.

Fifthly, as resources are shared among several users, applications and VOs, it is necessary for **isolation mechanisms** to be in place. Isolation mechanisms need to provide assurance of exclusive access data and processes that have been assigned to particular users, groups and VOs.

Finally, there are often cases where a security architecture needs to be tuned and re-engineered as the security mechanisms and protocols interfere with other

properties of the overall system including performance, scalability, efficiency, usability, accessibility and manageability. This is therefore a challenge for **security management** and its relation to more general systems management. Management should should also be considered as one of the directions for innovations in XtreemOS security.

1.2 The Innovation of XtreemOS Security

The key innovation of XtreemOS and of XtreemOS security is moving the Grid security logic (GSEC) into the Operating System Logic (OSL). It is intended that better management automation and consistency with OS security are achieved in doing so. Figure 1 depicts this fundamental design choice in XtreemOS. It shows that Traditional Grid Middleware has the Grid Logic (GRD) and hence the GSEC separated from the operating system logic (OSL) as well as the application system logic (APP) and application security (ASEC). This means that there are more bundles of software to be managed and a higher likelihood of security integration problems for developers and administrators. There is also higher likelihood of inconsistencies between the various layers. Grid-enabled Applications build the GRD and GSEC into the application. This entails that the GRD and GSEC are custom-built and deployed per application. This results in redundancy of functionality when multiple applications are installed in a system, as well as the likelihood for resource contention and conflicting access policies. One application might allow a user illegitimate access to data and resources that another blocks. The idealistic goal for a Grid-enables OS is that the applications treat the GRD and GSEC transparently; there is no need to change the API of the OS nor change the behavior and management of the application. XtreemOS does not achieve this ideal but uses this as a motivation.

Given that XtreemOS adheres to well-known, fundamental security design principles, protocols and mechanisms, we believe that more emphasis needs to be placed on identifying the core values that the design and implementation choices can bring to an organisation. In the context of Grid-based systems, the value of design decisions associated with software solutions, including security, are important from the perspective of software engineers, application service providers, infrastructure providers and administrators. These are the potential parties to have most impact if selecting XtreemOS or some Grid solution with architectural similarities. By achieving better management automation and integration with OS security, it is intended that the following higher-level goals for XtreemOS are supported by security, given that the security implementation is correct:

1. *Better Resource Utilization*: as opposed to having compute nodes performing at maximum constantly, have a means of adjusting their resource con-

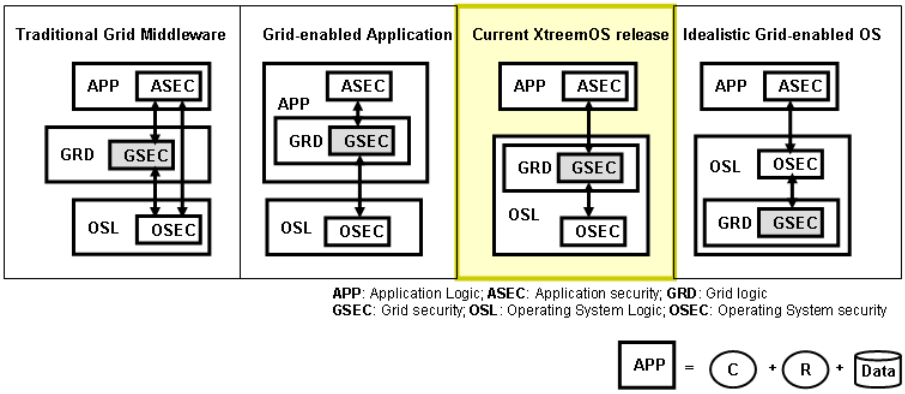

Figure 1: Strategic positioning of XtreemOS security with respect to other Grid-security and Operating System architectures

sumption according to their load, as well as detecting when compute needs can and need to increase their resource consumption. This involves monitoring components that can accesses instrument data across the execution environment. The security design and mechanisms should make it easy to configure this type of monitoring without destroying other security guarantees.

2. *Flexible Resource Sharing*: different or replicated processes and objects of the same application can be executed in parallel on different physical nodes yet managed in the same application context. Nodes can be introduced or excluded to and from the application context according to their current utilization. The security design and mechanisms should support adding and removing of resources to the execution environment of applications rapidly and securely.

3. *Distributed Data Management*: more scalable queries and cost-effective storage and archival of up to petabytes of data by using a parallelized, distributed data storage and management infrastructure. The security protocols and mechanisms need to be sufficiently scalable and fast to support such large scales of data.

The analysis performed on XtreemOS security is motivated from a business perspective, but requires technical insights into the underlying concepts and mechanisms for comprehensive assessment.

1.3 Features of XtreemOS

XtreemOS aims to support VOs within the entire operating system. The concept of a VO is fundamental in many Grid security specifications. The XtreemOS approach consists of two types of entities: global entities and local entities, which are conventional operating system level entities. Local entities are users and resources such as files and processes, which are consistent with the concept in Linux. Global entities are identifiable using a global identifier that agrees with a specific format and alphabet in a given global namespace. These global entities include users, machines, services and VOs.

In order to support global and local entities securely, XtreemOS needs to provide a set of system-wide (global) services to serve any machine running XtreemOS within the context of a single deployment. These system-wide services provide the following functionalities:

- manage global identities such that they remain unique and can be used for authentication
- manage attributes that are assigned to globally-unique identifiers
- manage policies that govern membership and resource usage globally

In addition, there is a need for new node-level system services to be installed on the native OS in order to coordinate with the global services. These local services have the following properties and functionalities:

- provide bi-directional mappings between a global user and a local user
- mapping of global and local resources
- allow OS-level services to recognize and make use of global attributes in its handling of local resources and providing varied quality of services guarantees
- corporately work with global services to enforce policies to govern the usage of resources locally

Along with the support for maintaining associations between local and global identities, there are four types of identifiers in a deployment of a XtreemOS system:

1. Global User Identifier (GUID)
2. Global Virtual organization Identifier (GVID)

3. Global Group Identifier (GGID)

4. Global Node Identifier (GNID)

This implies that both human users and computer nodes need to have credentials. These are technically encoded using X.509 public key certificates, such that for each user and each node there is a separate public/private key pair associated in order to support authentication of users and nodes. The GUID, GVID and GGID and GNID are encoded as attributes in the certificates. This full specification is known as a *XtreemOS Identity Certificate XOS-Cert*.

1.4 Document Structure

The document proceeds with the background to XtreemOS security, including an analysis of wider security issues for large-scale and Grid systems, as well as a description of the evaluation methodology. The remainder of the document dedicates a section to each the five classes of security requirements:

1. XtreemOS Data Storage Security

2. XtreemOS Communications Security

3. XtreemOS Identity and Membership Management

4. XtreemOS Resource Management Security

5. XtreemOS Isolation

For each class of security requirement, the ability of the XtreemOS security specification to satisfy each of the specific security objectives is assessed. An evaluation summary includes a higher-level evaluation of the management aspects surrounding each class of security requirement. A brief conclusion is included to summarise and state the next steps for evaluation and assessment of XtreemOS security.

2 Background and Evaluation Methodology

Security is not effectively solved by applying cryptographic protocols, firewalls and public key infrastructures, as they may tend to break the strategic goals of the organisations employing them. The way in which security architectures and mechanisms are implemented and configured must consider the nature of the business decision with respect to establishing, continuing or terminating business relationships, and hence the secure, on-demand configuration, modification and termination of the security mechanisms involved. In a Grid-based execution environment, as targeted by XtreemOS, the following standard system management and administration activities become more challenging, moreover when security is integrated as an additional concern:

1. *Select compute node(s)*- Any application has various technical resource requirements for its individual components and their interactions. These requirements may be generic for any instance of the application or may be dependent on customer requirements. A selected node should have the correct platform, locale, capacities, connectivity and so on. This implies that data about the nodes capabilities and its need to be registered and made available to administrators that need to install applications. If the application already running, then there should be possibilities for moving it to another node, known as *migration* (see activity 6).

2. *Install software*- Once a node is selected, the application's binaries and data need to be securely transferred or even streamed in preparation for set up. This is normally done by delivering an installer or zipped archive of these binaries [1]. Different pieces of registry data about the node are required by the installer in order to complete the installation. The access to these pieces of data must be provided.

3. *Configure application and node*- Configuration is the setting of software or hardware variables that change the operational behavior and constraints of a system i.e.application or node. The above-mentioned task of providing the data about the node is an initial configuration step, as the installers need to run as a privileged user of the node. However, the assumption that the installer is acting on behalf of the local root administrator is false from a liability and contract perspective.

4. *Deploy application*- After the application and node are configured correctly, for the purposes of their intended operation, the application is deployed.

[1] There are other means of delivering software such as Microsoft's SoftGrid, but still the standard practice for large applications is an installer.

Deployment is making the application available, indicating to its intended users that it is available, providing them with a path to the EPRs (end point references) of the resources they require.

5. *Monitor application and node-* As hosting is based on contracts and performing third-party support, monitoring is even more critical as badly performing applications or nodes can lead to court-cases and payment refusals.

6. *Migrate application-* One of the decisions following monitoring is that one node no longer has the capacity to support its expected load. For this purpose the application and data could be moved to another node or made to be striped across multiple nodes. Another reason to migrate is during testing, as there are typically different nodes and networks for hosting applications while in production/live or test mode.

7. *Terminate application-* Once the application is no longer required or its lease has expired, the application and all residual access to the execution environment needs to be securely removed. The client may also request some guarantee from the provider that all clean-up actions have been completed.

The next section now proceeds to discuss the challenges that arise for the application management lifecycle in a hosting environment, recalling the overall objectives of better resource utilization, flexible resource sharing and distributed data management.

2.1 Challenges for XtreemOS Security Services

In the general case, gaining access to Grid computing execution environments running business applications and storing business data would be more attractive to attackers than gaining access to raw scientific data. However, a security concept that does not consider the peculiarities of the application domain may circumvent the ability to execute, use and manage. There are hence 4 broad objectives to be addressed for security and security services in XtreemOS:

1. The security services and mechanisms in XtreemOS allow applications and jobs to be properly installed, executed and monitored without forcing significant change to the application software.

2. The security properties (confidentiality, integrity and availability) of applications and jobs be maintained in XtreemOS, or do compromises have to be met for the sake of operation

3. The correct segregation of duty policies and other control objectives in applications and jobs can still be strongly guaranteed when running in an XtreemOS environment.

4. The manageability of security in applications and jobs is not made unacceptably complicated by introducing XtreemOS

In order to comprehensively investigate the security problems, the Grid environment is a collection of computational nodes connected by an insecure network, where each node is a computer running a XtreemOS distribution and is trusted for local security. A compute node that is *trusted for local security* suggests the following assumptions that can be locally attested:

- the node is capable of performing authentication, privilege assignment, authorization, encryption/decryption of data and maintaining logs of transactions

- any user or administrator of the node believes that there is no unauthorized process running locally that has administrative rights on the node, or that can perform the above (authentication etc), other than those designated.

- the node has a reliable mechanism that ensures that only privileged users and administrators can perform other privileged actions such as changing settings, installing applications, adding/removing users and adding/removing privileges

- users of the node believe that processes and objects executed and used on the node are the ones that they intend to use.

- the node has a reliable mechanism that ensures that users can only access processes and data to which they have locally specified rights, including ownership. The node therefore has a means of securely retrieving the set of valid users.

- administrators believe that other administrators will not try to violate the node by circumventing the above, but expect that users may maliciously or accidentally attempt to do so

The above are not assumed when nodes are accessed over a non-trusted network, as there can be the chance for man-in-the-middle attacks and there is lesser quality in the attestation of the node. Even if the nodes are running in a single corporate domain, for example a data-center, the corporate network is treated as non-trusted, as the executables running on each node (and even the same node)

could be from different organizations. Furthermore, future Data-Centers will want to offer additional services that expose some data for monitoring, billing and accounting of the resources allocated to them, which could mistakenly provide them with access to the resource performance data of other users. In other words, the platform needs to provide the assurance that a customer can only view the collective resource usage of the set of resources (processes and objects) currently and potentially available to host their deployments, as well as sufficiently tune them for their purposes. It is not possible to predict the extent of digital espionage and malice in the future, but not taking the time to carefully understand and consider the implications for badly designed security in such execution environments could lead to significant problems in the future. In XtreemOS we have the rare opportunity to look at these potential problems from the lowest level of the software and services stack: the Operating System. The OS is today the notion of a machine for most system developers and administrators, underlying how installation of applications is performed.

2.2 VO and Application Management Challenges

As a VO goes through its lifetime of creation, operation and dissolution, several application installers and job managers need to carry out various actions to configure the underlying systems in response to changes in the VO and resource requirements of various members. Application installers and job managers require privileged root access to target hosts in order to customize and tune the machine for interoperability and efficient support of the software and data required for the job at hand. They need to query, set, disable and enable certain variables in the registry, according to the setting stored for the operating system in the system library (syslib). These include the following:

- `accounts`: creation of users and groups, and manipulating access control lists
- `filesystem`: creating installation files and directories, copying and deleting them as well as working with mount points and shares.
- `network`: retrieving the IP address of the network card, registering services (e.g.in the /etc/services file) and altering the IP tables in the /etc/hosts file.
- `process`: manipulating context of current process of external programs from which information is required
- `system`: retrieval of hardware information including RAM, storage, num processors, processor speed, storage type, BIOS

- `timer`: use of time functions for logging, timing and scheduling installation events

Furthermore, in order to perform these actions, the installer needs to assume a standard interface to the OS for performing these actions. In the case of a UNIX-like OS, this interface would need to be POSIX compliant. POSIX-compliance also places some constraints on the way in which users, groups and access controls are organized. Granting privileged root access to installers and job managers on various machines must be done by introducing security holes in the systems. The existence of these holes becomes more challenging to predict and discover when a VO model of collaboration and resource sharing is employed. Granting of temporary privileges, including root privileges, has to be done dynamically, for many different users and usage scenarios, relatively fast and over the wire, such that configuring authorizations and performing installation with too much overhead and manual administrator intervention is undesirable. Furthermore, installation is not a one-off batch job but includes ongoing monitoring and updates of the installation. Solutions for application management need to be scalable, usable and persistent without exposing the target host and its local network to significant risks. Even if the installer or job manager were to be malicious, while having privileged access to the host, the impact and propagation of its attacks need to be contained.

Most business applications are not simple software bundles, but they are typically composed of various components and stand-alone utilities for user interface rendering (i.e. presentation/web servers), execution of business processes and solution logic (i.e. application servers) and maintaining queries and persistence of the critical data that continues to grow with every transaction. This separation into various layers, as shown in Figure 2, means that there are different requirements for performance, scalability and access.

Each server/node would then have a different configuration based on the types of data, processes and connections they need to handle. Configurations of the operating system(s) may be complex, such that doing them incorrectly may lead to operational conflicts, inconsistencies and undesirable application downtime. Another consideration is if there is a need to move applications from one node to another as a result of failure, predicted load increases, test-to-live transition, new hardware or changing partnerships. Being able to migrate operating system instances across distinct physical nodes is a useful tool for administrators of data centers and clusters: It allows a clean separation between hardware and software, and facilitates fault management, load balancing, and low-level system maintenance *[see: Live Migration of Virtual Machines, Clark et al]*.

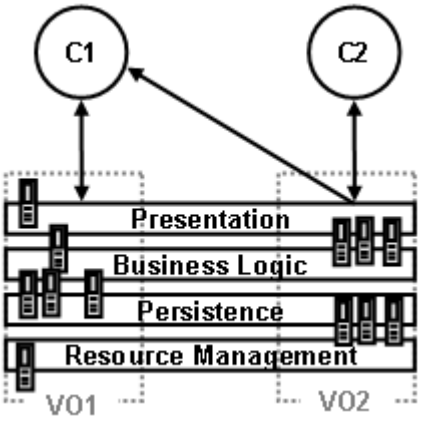

Figure 2: Complex systems are divided into multiple layers that address different concerns of an application or job being processed within a VO context

2.3 User and Certificate Management Challenges

The management of a large set of users is a challenge. A reason for this is scalability and availability of an identity management infrastructure and also the management of user identities and rights in systems among different user management concepts. E.g. UNIX uses the notion of users and groups to describe right and thus access control to files. This was appropriate for the context it was made for. However it cannot cope easily with today's needs. More recent derivatives provide the concept of roles and flat access control lists.

However most legacy applications rely on the user/group notion to achieve security by the concept of delegation of duties. Attackers have multiple barriers to break in order to reach their goal. This also helps to prevent the propagation of errors. There are in addition cases where particular actions still require root access to machines in order to be effective. This is particularly true during software installation and tuning of the infrastructure. Standard mechanisms such as SSL (Secure Socket Layer) exist and are incorporated with Linux, but need to be properly integrated into the higher-level VO, application and job management protocols in the Grid system. Improper usage of these for managing authentication and authorisation keys and data results in breakage of preexisting security properties of a node.

2.4 Evaluation Methodology

The evaluation methodology starts by separating XtreemOS security into 5 classes of security requirements in Grid systems. The first part of the evaluation checks if

the requirements are satisfied by the current XtreemOS security specification and implementation. These requirements were identified in the initial Applications and Evaluation WP deliverables D4.2.1 [3] on requirements capture and use-case scenarios for XtreemOS, where security requirements are discussed in section 4.9. They are however now grouped to consider 5 different classes of security requirements: (1) data storage, (2) communications, (3) identities, (4) resource management and (5) isolation. The ability of XtreemOS security to satisfy these requirements is rated as follows:

1. Fully capable
2. Highly dependent on underlying Linux platform[2]
3. Highly dependent on 3rd party software not necessarily bundled with Linux
4. Partially implemented
5. Not implemented

These requirements are consistent with the Grid security requirements models in the literature[8, 7, 16].

The second perspective evaluates XtreemOS security from a management perspective. As already noted in the introduction and in the preceding technical discussion, there is a need to assess the value of design decisions for security in addition to the ability of a solution to protect against malice or mischance. XtreemOS security does not deviate significantly from existing Grid security protocols and mechanisms, such that the important points refer to the innovations of better management automation and OS integration. In order to do the assessment, either qualitatively or quantitatively, it is necessary to have concrete questions that can be asked about the system. In doing so, we sought to identify the issues that challenge the management of large-scale systems, in particular those based on Linux. In his 2005 SYS-CON article, Akmal Khan identifies some challenges for Enterprise Linux System Management [1]. These challenges are summarised below:

- *Abundance of Servers*: the more servers in use, the more differences and interdependencies.
- *Lack of Sophisticated Tools*: traditional scripting and procedural administration are not scalable for managing hundreds or even thousands of nearly identical servers

[2]XtreemOS is a Linux-based OS; This however means that the XtreemOS technology does not add anything specific but makes an assumption that the pre-existing mechanisms are correct

- *Version Control and Conflicts*: changes to the core of Linux result in multiple customized flavors of Linux that increases the heterogeneity of the environment, increasing the challenges associated with version control.
- *Monitoring*: need for an integrated, centralized monitoring and management system.
- *Disaster Recovery Issues*: the need to avoid having to manually manage individual servers on a regular basis due to failure or compromise.
- *Patch Management and Deployment*: maintenance of uniformity of all installations is tedious.
- *Custom Application Deployment and Package Manager Integration*: complex and customised software is more difficult to package and correctly repackage. Flawed packaging will lead to more frequent roll backs.

Consider that the above challenges are now made more extreme when considering the cross-domain and dynamic Grid execution environments targeted by XtreemOS. In order to further understand the nature of these challenges, a trace of systems management activities starting from the installation of software was performed. Figure 3 shows the result of a trace of system management activities throughout the lifetime of a system, showing their dependencies. The dependencies (in the form of control flow) are indicators of the relative criticality of each step, as well as of the corresponding challenge discussed below.

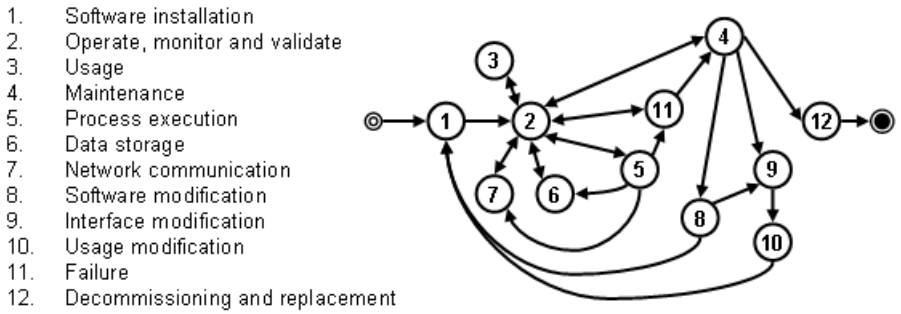

Figure 3: A trace of system management activities to show where challenges occur

The consolidated evaluation questions were derived from this trace, considering the problems for Enterprise Linux systems management and the value criteria discussed previously. These questions and their relevance are now discussed below, corresponding to 1-12 in Figure 3:

(1) Installation complexity of software increases the cost and time required to install, as well as increases the risk of flawed installation. Minimisation of installation complexity for security is hence critical. Secondly, the inclusion of security services and mechanisms in a system should be well integrated with the standard installation of the system.

(2) Human analysis during operation, monitoring and validation of a system's runtime does not scale. The information required for performing analysis that a system is correctly (and securely) operating needs to be kept to a minimum. Automation for operation, monitoring and validation is encouraged. The reduction of human analysis is hence the most critical value-factor considered in the evaluation.

(3) Human interaction and specification is unavoidable in interactive systems, as well as during administration and maintenance. The complexity of interfaces, forms and data structures should hence be kept to a minimum, such that only necessary interactions are included. This is a very application-dependent factor when considering value.

(4) Software tools for maintenance should be as simple and familiar as possible. Given that XtreemOS is based on Linux, the tools required to manage XtreemOS security should be all based on Linux or other standard, well-established tools for security management. It is considered that users of XtreemOS are not necessarily familiar with traditional Grid infrastructures and Grid security.

(5) Processing complexity is dependent on the type of application, networking and hardware architecture in which XtreemOS is deployed. This factor for evaluation relates to the process execution stage in a system's lifetime. Security and security management should add minimal overhead to a system's standard processing. Security protocols and mechanisms are often introduction of bottlenecks in systems. Processing complexity can have dependencies on almost every other aspect.

(6) Storage requirements are of concern for Grid systems. It is not expected that security policies, certificates and key information disrupt the storage capabilities of a system significantly.

(7) Communications requirements should not be adversely affected by the cryptographic operations and key management of XtreemOS security. Given that

XtreemOS does not prescribe the cryptographic mechanisms to be used, there is some flexibility in selecting and tuning cryptographic parameters to control this factor.

(8) Changes to application methods should be minimal, as XtreemOS security should work with existing/legacy application software.

(9) Changes to user interface should also be minimal, again for the purpose of supporting legacy applications. This could have consequences for training and usability of software applications.

(10) Changes to user practices should be minimal. Users typically particular expectations from the configuration and presentation of software. If security changes this significantly, users are prone to make mistakes that can introduce security holes in the system.

(11) Central points of failure should typically be avoided in large systems, as the compromise of this point will lead to the compromise of a large set of nodes and system elements. From

(12) Technology and vendor lock-in represents when a customer is technically forced to use a certain technology for a given task, even if better, cheaper alternatives come on the market. XtreemOS security should continue to support openness and "pluggability" required in heterogeneous computation environments.

In the evaluation summaries of each chapter, XtreemOS security is rated against these 12 criteria in question form, although each question is not relevant for every feature. In some cases direct comparisons are made in relation to traditional Grid middleware and a standard Linux environment where applicable. This is done to emphasise where the key innovations are in XtreemOS security.

3 Evaluation of XtreemOS Identity and Membership Management

Authentication is a primary security requirement. Without authentication it is hardly possible to realise any other security requirement. It is found that most of the existing Grid security solutions, such as the Grid Security Infrastructure (GSI)[16], the foundation of many other solutions, focus initially on how entities are identified and how membership is maintained. The Community Authorization Service (CAS)[2] for example builds on the public key authentication and delegation mechanisms of GSI. XtreemOS security uses similar architectural decisions and mechanisms as GSI - in some cases the GSI mechanisms can be used (although this is not yet fully implemented and supported at the time of evaluation). This section now looks at how the XtreemOS security specification is designed to implement protection mechanisms against threats to identity and membership in a Grid system.

3.1 Single Sign On

It should be possible for a user to use a single method of authentication (i.e. single sign-on) to gain authorized access to resources in a VO. Identification and authentication are again fundamental requirements for security, as integrity and confidentiality are difficult without the capability to identify and authenticate principals. Identification ensures that different principals (e.g. a source or receiver of a message) are repeatedly distinguishable from each other, while authentication associates attributes used to identify a principal with a unique root attribute such as a legal name or public key. It must be possible for all resources in a VO to identify and authenticate users requesting access to data. Users of resources should not have to be bothered with changing the way they interact due to changes in the hosting of the resource. One example is cross-domain single-sign-on (SSO), which requires an agreement of how tickets and attributes are encoded and verified, which assert that users have been authenticated and possess the appropriate authorizations to perform actions in the VO.

Figure 4 shows a client C requiring access to a receivers R1 and R2 on Resource-1 and Resource-2 respectively, where both are in the same VO. If for every change in resource C required a new means of authentication, this would not scale. Consider a system where C had files striped across just 10 disks. This already suggests that C would need to have 10 different key-pairs and passphrases (unless using the same passphrase for all). As long as the CŠs passphrase remains secure then there is no chance of X tampering with or stealing CŠs key.

However, the implementation must be secure against an attacker X that tries

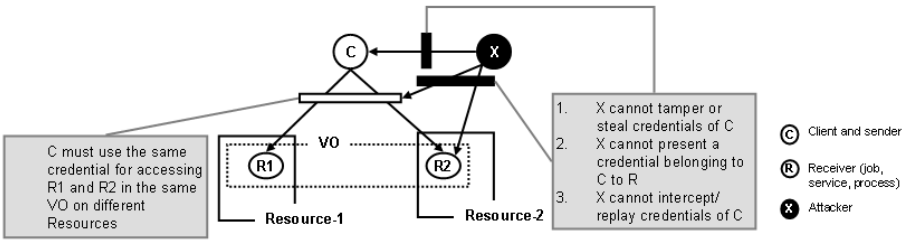

Figure 4: Single sign on authentication VO security requirements model

to steal the credential of C and gain access to one of the receivers. In order to support SSO, XtreemOS security would be integrated with GSI's proxy certificate approach. GSI focuses primarily on authentication and message protection, defining single sign-on algorithms and protocols, cross-domain authentication protocols. There exists such infrastructures that are readily available. XtreemOS already supports single-sign-on with proxy certificates using SSL, HTTPS and configuration files, as described in Section 2.4.6 of D3.5.11. SSO in XtreemOS is therefore **highly dependent on third-party mechanisms** such as GSI. GSI is however the established solution, such that interoperability with other forms of Grid systems like EGEE and TeraGrid would be supported.

3.2 Delegation

In a Grid system the assumption that all entities will always be available and capable of performing actions does not reflect reality. Clients will need to delegate authority to other clients to act on their behalf, should they need to be offline or require parallelisation. It must be possible to transfer and validate authorizations to virtualized resources when the host is changed. This is known as dynamic delegation of authority or privileges. In Figure 5, two forms of delegation are shown. Firstly, it is shown that a client C1 goes offline but needs to complete work with a receiver R. C1 then delegates to another client C2, such that C2 can interact with R. Secondly, R is initially on Resource-1. If Resource-1 needs to be allocated for other purposes, R can be migrated to Resource-2, given that Resource-2 has the authority to host R. Therefore Resource-1 also needs to delegate this authority, such that C2 can validate that Resource-2 is a valid host before accessing R running on Resource-2.

Figure 5 also shows an attacker X attempting to tamper with the delegation process, as it presents an opportunity for masquerading. This can be protected against using Proxy Certificates, already introduced when discussing SSO and also discussed in detail in D3.5.11[4]. Proxy Certificates allow an entity hold-

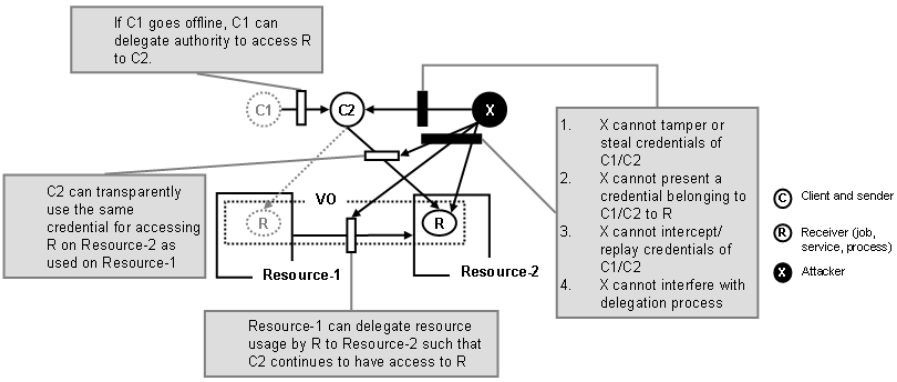

Figure 5: Delegation VO security requirements model

ing a standard X.509 public key certificate to delegate some or all of its privileges to another entity which may not hold X.509 credentials at the time of delegation[17]. As the XtreemOS-Certs are based on X.509 certificates, it is feasible to use Proxy Certificates for dynamic delegation. Delegation in XtreemOS is also **highly dependent on third-party mechanisms**. Again, GSI is the defacto standard for Grid security infrastructure such that attempting to fully recreate and re-implement for this requirement is not of value.

3.3 Evaluation Summary

SSO and delegation are especially important requirements in large scale system deployments. Data and application processes need to be distributed across multiple nodes for the purposes of scalability and redundancy for ensuring fault tolerance and high availability of systems. XtreemOS security fully supports the integration of third party mechanisms for these purposes, such as the GSI proxy certificates[7] used in most other Grid systems.

Installation complexity in this regard is similar to standard Linux and Grid middleware. The creation and distribution of proxy certificates for delegation purposes would be built on some of the mechanisms in place for the automated certificate distribution. There are hence some potential advantages for reducing the complexity of installation and system initialisation.

Enabling SSO and delegation reduces the additional analysis, specification and interaction for security management by users and administrators. Users can use the same certificates and public keys to access a multitude of nodes in the entire Grid system.

4 Evaluation of XtreemOS Data Storage Security

Given that XtreemOS targets a wide range of applications, it must be assumed that attackers will attempt to gain access to data for illegitimate reading, writing or deletion. There is hence a need for strong access control and integrity mechanisms in XtreemOS for data security.

4.1 Access Control

Data stored on resources must only be accessible by users and administrators that are members of a VO with the appropriate read access rights. Confidentiality is a fundamental requirement of systems that store, process and exchange sensitive data and information. In a Grid-enabled system the requirement for confidentiality of stored data is to ensure that data can only be accessed and read by services, users and administrators (together known as Principals) that have a need to read the data. A principal C has a need if the following conditions are true: C is owner of the data OR C is registered as a member of a VO with rights to the data AND assigned to a task that requires access to the data. A principal with such properties is referred to as a "valid principal" otherwise we refer to the principal as an "invalid principal". Figure 6 shows that a Client C is a valid principal with respect to a targeted Data store if an only if C is still a member of the VO within which the Data belongs, it conforms to a predefined resource usage policy, C is the owner of the Data or C is the administrator of the Resource hosting the Data.

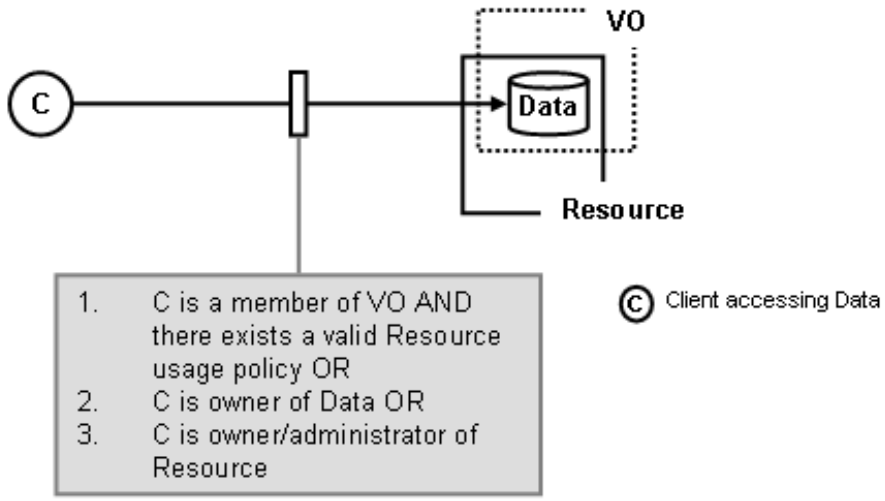

Figure 6: Access control VO security requirements model

The security requirements model is implemented locally on each resource as there is no central, system-wide access controller in the deployments assumed by XtreemOS. They are encoded as two types of policies: VO-level and node-level. The policies are however only enforceable using the Pluggable Authentication Module (PAM) described in the Node-level VO support mechanisms[5]. The Client must first be issued with a XtreemOS-Cert, issued by a trusted Credential Distribution Authority (CDA), which asserts a Global Unique Identifier for authentication, Global Group Identifier (GGID) that is compatible with a group that has access to the Data and a Global VO Identifier (GVID) that represents the VO within which the Data is hosted.

Stronger requirements might exist where the latter constraint (i.e. C is the administrator of the Resource hence can access all) is not permitted, but this would require that encrypted data be stored without the secret key. The data would have to be decrypted locally at the Client. Enforcing this is therefore an administrative decision and not a core requirement for XtreemOS security. This security requirements model is therefore **fully supported** by XtreemOS security.

4.2 Data Storage Integrity

Loss of integrity of stored data must be preventable and detectable using hash mechanisms (i.e. checksums), the standard approach to integrity enforcement[15, 14]. Storage integrity is typically stated as the ability to prevent illegal changes to data. As data in a VO may be sourced from different participants, who may not be "owners" of the data, it must be possible to validate that the data has not been altered by illegitimate principals. Data should be hashed and digitally signed by a trusted key stored on the operating system. Again a legitimate party must be a member of a VO and have the appropriate rights to make changes to data.

Figure 7 shows an attacker X potentially gaining write access to Data, although the Access Control model above in subsection 4.1 should serve to prevent this from occurring. In order for X to write to Data without corrupting it, X must be the owner of the *last valid owner's* private key used to sign a hash of the data. If X writes to the Data without having been issued the respective key, then C can detect illegitimate changes made to the data.

Given that the access control requirements are correctly implemented, it can be guaranteed that X has no access to data if it has not been issued with a credential from a trusted CDA. However, for strong integrity to be ensured, the Standard Linux hash digest mechanisms for checksum functions are applied. In addition to using hash digests such as SHA and MD5, Linux provides commands for creation and verification of hash digests. The `md5sum` program is installed by default in most UNIX, Linux, and UNIX■like operating systems[14]. The data integrity function is therefore **highly dependent on the local Linux OS** in order to be

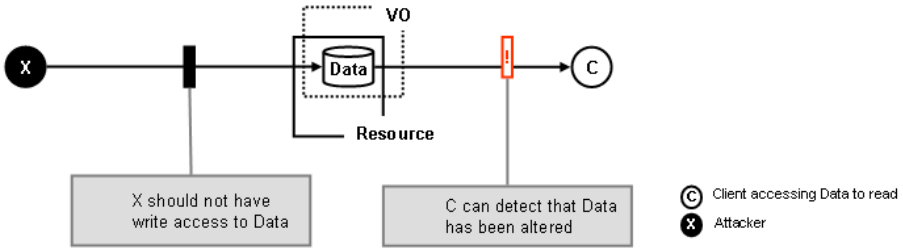

Figure 7: Integrity VO security requirements model

implemented.

4.3 Evaluation Summary

The value of strong storage security is a clear requirement for business applications, given legislation and potential economic impact of data loss. Installation complexity for storage security is in relation to the installation of cryptographic keys and policies required for authentication, access control decision making and integrity checks. The additional complexity introduced by XtreemOS is the need to maintain mappings between global and local accounts such that a global and local identity are mappable to a single key-pair stored on a node. Given that the standard X.509 certificates are being used, there are no additional complexities for installation foreseen in comparison with a standard Linux platform. Similar installation procedures are also seen with Grid middleware in order to maintain data security.

5 Evaluation of XtreemOS Communications Security

Distributed systems require more communication than centralised systems, given that they are implementing the same application. This is a result of the need to remain consistency and for nodes to interact in order to achieve the functional requirements of the system. The communications is in addition more complex. The volume and complexity of communications makes it an attractive target for attackers.

5.1 Confidential Communication

Confidential data communicated between resources in the same VO must be encrypted using crypto protocols and keys agreed to within the VO. It is assumed that data is transmitted over insecure channels such as the Internet, such that there is a need for mechanisms that protect messages and responses in transit between computational nodes. More specifically, confidentiality of data is concerned with the protection of message inputs and the corresponding outputs of responses from invalid observers. An invalid observer has similar properties as that of an invalid principal of stored data as discussed in subsection 4.1. However, the security policies and mechanisms are now concerned with the properties of the channels over which messages and responses are transmitted.

Figure 8 shows a client C sending encrypted data to a receiver R. It should not be possible for an eavesdropper X to read the data intended for R.

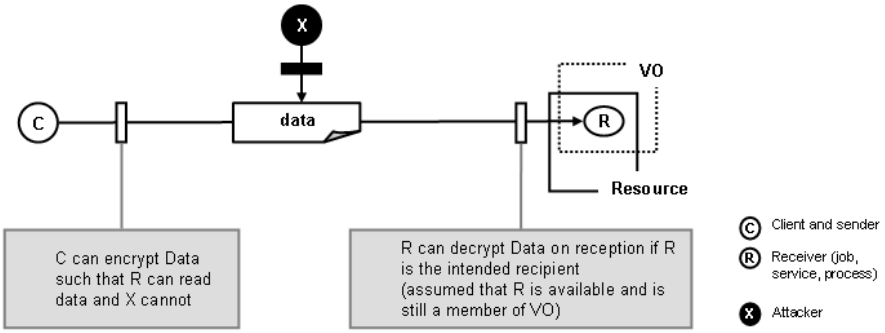

Figure 8: Communications confidentiality VO security requirements model

The underlying assumption when deploying a XtreemOS system is the availability of a Public Key Infrastructure (PKI)[11]. This is the underlying infrastructure of the XVOMS implementation. The mechanisms for maintaining confi-

dential communications are therefore standard with respect to a distributed Linux implementation. Following the PKI trust model[11], it is assumed that all communicating parties have key pairs. The public keys of the key-pairs are used as the basis for the XtreemOS-Certs issued to entities.

Secure Socket Layer (SSL) is used for all generation of session keys and encryption in the current release of XtreemOS security. There is no need to change the functionality of SSL as XtreemOS does not produce any new requirements for the session key generation implemented by SSL. XtreemOS security is therefore **highly dependent on the underlying OS**, i.e. the version of SSL integrated, in order to achieve secure communications. However, according to Mitchell et al.[13] SSL 3.0 is a robust solution for communications security, given improvements on SSL 2.0, as previously identified by Wagner and Schneier[18]. The implementation of XtreemOS is based on SSL 3.0, as instances that use SSL 2.0 cannot be guaranteed to be secure against version rollback attacks.

5.2 Integrity of Communication

The integrity of data transferred between resources or received from users must be validated before being committed. There is then a need for a mechanisms to capture and validate all incoming and outgoing network traffic. The integrity of communicated data is concerned with ensuring that illegal change is not possible to data in transit. If data is altered by an attacker X, it must be detectable by a receiver R, as shown in Figure 9. This differs from integrity data in storage as the properties of communication channels tend to be more dynamic, based on the location, operating system and medium used by end point nodes. The operating system must therefore be capable of signing and verifying signatures of data in an end-to-end manner. The term "committed" suggests that a transaction framework is necessary, considering the distributed nature of the resources.

The implementation of communications integrity in XtreemOS follows similar assumptions of confidential communications in subsection 5.1. There is a need for a PKI in place. XtreemOS security is therefore **highly dependent on the underlying OS** in order to achieve secure communications.

5.3 Evaluation Summary

XtreemOS is dependent on the native cryptographic mechanisms in order to support secure communications. Standard protocols for key negotiation and session key generation are applied.

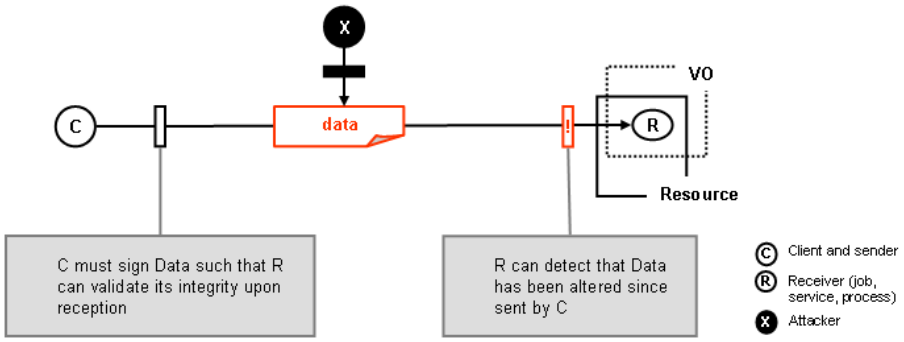

Figure 9: Communications integrity VO security requirements model

6 Evaluation of XtreemOS Resource Management Security

Resource management in XtreemOS can be described from a global and local, node-level perspective. From a global perspective it involves the assurance that only valid members of a VO gain access to valid resources. From a node level perspective local administrators must be able to account (and eventually charge) for the resources used by other entities. Security services and mechanisms are therefore part of the fundamental capabilities required for effective resource management.

6.1 VO Membership Verification

It must be possible to validate membership in VOs and ensure access to resources given proven membership and rights. Authorization and guaranteed access are two different requirements although enforced by interdependent security mechanisms/ services. That is, a principal may have been provided with a token, ticket or credential but the appropriate access control policy or service interface is not available at the time of request. The locally evaluated rules that determine if a party is authorized or not (beyond the possession of a token, ticket or credential), must also be agreed to across the set of resource providers. It is not possible to implement entirely in the OS, but there need to be calls to higher level services that can perform such evaluations.

Figure 10 shows a client C accessing a receiver R on a Resource, where some portion of the Resource within which R is running has been made available within a VO. C must present a credential signed by a Membership Manager M that proves that C is part of the VO and can hence have access to the Resource. Secondly, from

the perspective of the Resource, the receiver R must be capable of validating that the Resource it is currently being executed on is authentic and has been selected as a resource by the membership manager M of the VO. The credentials encoding the membership of C and R must be unforgeable by an attacker X, such that there is some assurance of linking the correct client with the correct resource and correctly tracking resource usage.

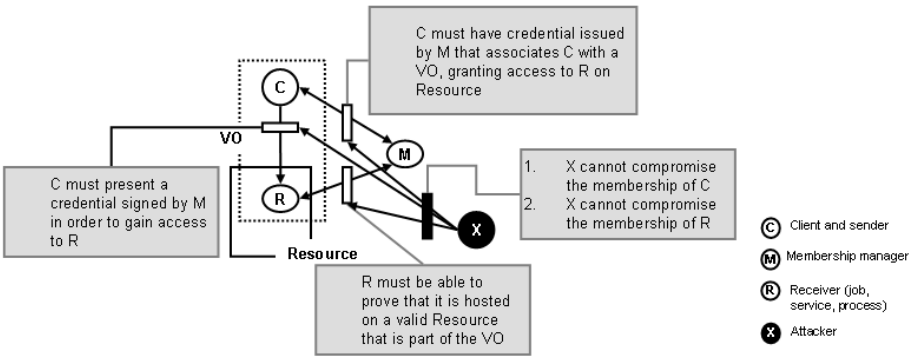

Figure 10: Membership verification VO security requirements model

In XtreemOs the functionality of the Membership Manager M is implemented as a combination of the CDA and XVOMS. Page 52 of deliverable D3.5.11[4] shows the interaction of these components in a single trust domain. XtreemOS also introduces the notion of a Resource Certification Authority (RCA) that issues resource certificates to resources to mark them as valid resources. This therefore covers point number 2. Membership verification is **fully implemented** by the XtreemOS security services.

6.2 Accounting

It must be possible for administrators to record usage (by whom and when) of resources without users being able to deny (repudiate) usage. Accountability is the ability to enforce and prove that a principal has performed an action on a given resource at a given time. The requirements for accountability are typically a secure audit service with the ability to timestamp messages. This is for the purposes of non-repudiation, should there be a case where it must be proven that a principal has indeed performed an action, as well as billing. It should also be possible to record within which VO the resource was used.

The security requirements model for accounting is shown in Figure 11. The first requirement here states that a client C must not be capable of using more

resources (e.g. CPU, Storage, Networking, Licensed time) than granted by a local resource administrator A to the user while acting as a member of the VO. Secondly, the revocation of C's privileges by the local administrator A must always be possible. A must hence be capable of monitoring C's resource usage and detecting attempts at violation of resource usage policies.

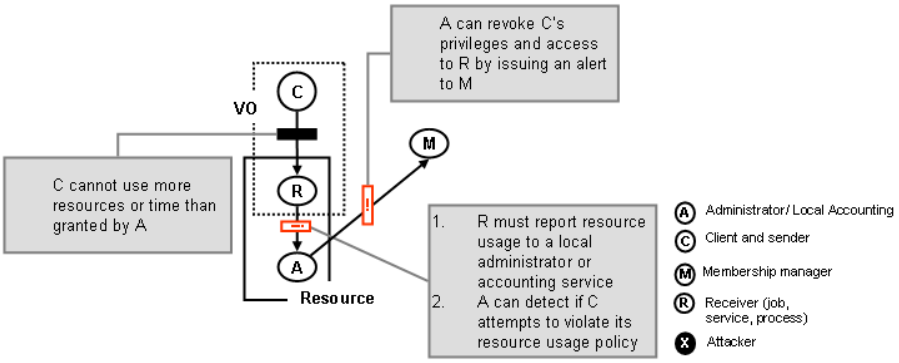

Figure 11: Accounting and resource usage VO security requirements model

The accounting requirements primarily depend on the node-level VO support mechanisms in WP2.1. However, accounting is planned for the next release, as stated in section 3.2.8 of deliverable D3.5.11/citeD3511. Accounting is hence **not implemented** in the current release. The VOPS is already available as a means of specifying global resource usage policies but the underlying mechanisms to effectively enforce and monitor based on these policies locally are not yet in place.

6.3 Evaluation Summary

The novelty for Grid resource management XtreemOS introduces here is that of the Resource Certification Authority (RCA) and the issuing of special XtreemOS certificates for resources. Hence there is some mechanism for providing clients with assurance that the resources they require will be available. Both clients and resources can be managed as members of VOs. Membership verification is hence the only means of protecting resource management at the moment, with some dependence on the node-level VO support in WP2.1.

Installation complexity is reduced by the automated distributed certificate management protocols, given the assumption of a PKI. It is however not yet possible to evaluate or estimate the additional complexity that will be introduced for resource administrators when fine-grained accounting is to be done for a large set of clients. A comprehensive evaluation of secure resource management in XtreemOS must hence be evaluated first in the next release.

7 Evaluation of XtreemOS Isolation

Isolation is a protection goal for data and processes. It combines the guarantees of confidentiality and non-interference. Given that confidentiality is a fundamental security requirement and most organizations with sensitive data will time and money in acquiring, developing and integrating mechanisms to enforce confidentiality policies. However, changing operational models to support resource sharing can serve to compromise the confidentiality property if isolation mechanisms are not included.

7.1 Data Isolation

Data isolation is traditionally the assurance that an executing transaction has exclusive access to its data, such that the data does not change between operations. Otherwise the transaction results are not predictable. Data isolation in XtreemOS is the assurance that data elements on the same resource can be operated on exclusively when under the ownership of different users, groups and VOs. This suggests that users, group administrators and VO administrators have some assurance of control over data that they currently have the ownership of even if sharing physical resources with other users and administrators. Data belonging to a VO should be logically isolated only for that VO, such that changes made to data belonging to one VO, although on the same resource, should affect the data belonging to another VO.

Figure 12 shows that a client C can only have access to Data1 that is under the ownership of a particular VO, the one in which C is a member. The following should be unknown to C concerning Data2, which should be isolated from Data1: (1) C should not know that Data2 exists on the resource; (2) C should not be able to determine any meta-data that describes the contents, location, size, owner or access control policies of Data2; (3) C should not be able to inadvertently delete, alter or corrupt Data2 by performing operations on Data1.

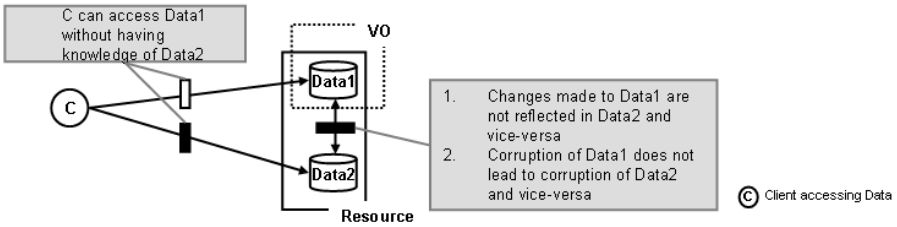

Figure 12: Data isolation security requirements model

The XtreemOS software does not itself implement data isolation mechanisms, but has the mechanisms in place for integrating with native isolation mechanisms of Linux. For example, in recent Linux kernels, a filesystem namespace provides a private filesystem tree to the process running in the namespace. The application can mount its own filesystems on its own file tree without making this local tree visible to other namespaces[6]. With this capability in place, the requirements in Figure 12 hold. Data isolation in XtreemOS is however **highly dependent on third-party mechanisms**.

7.2 User and Service Isolation

The security access to *virtualized resources* and services must be customized for each VO. It must be possible to maintain users for different VOs separately. As users may be involved in multiple VOs, it is then necessary to separate their user data and have a means of determining for which VOs they are currently working when accessing data. XtreemOS. In addition, services must also be isolated per VO. That is, different instances of the same VO will have different security requirements and must not provide means of illegal information flow. It must not be possible for parties in different VOs to recognize that they are sharing resources nor to gain knowledge of what other parties are doing with those resources. If one of two virtualized services to the same physical resource fails, this should not interfere with the other. Figure 13 shows two clients C1 and C2 in two different VOs VO1 and VO2 accessing receivers R1 and R2 respectively on the same Resource. Data isolation is again a concern here as well, as R1's access to Data1 must not interfere with R2's access to Data2 and vice versa.

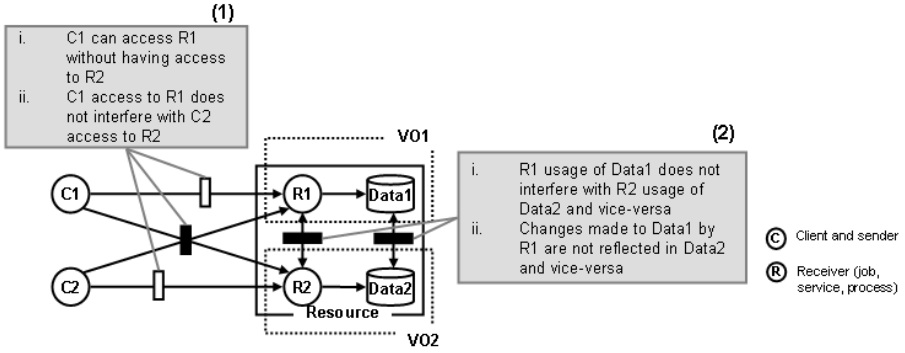

Figure 13: Service and user isolation security requirements model

Service isolation is not discussed within deliverable D3.5.11 explicitly. However, given that access control decisions are decided with respect to VO mem-

bership, this is a first level of isolation for services. Software processes running services may also be contained within virtual machines or containers. Service isolation in XtreemOS is hence **highly dependent on third-party mechanisms**. Building specific, low-level Linux isolation mechanisms is beyond the scope and effort of the project. As discussed in WP2.1[6], *the implementation of these requirements in XtreemOS necessitates some support from the Linux kernel in order to control and limit resource usage, protect applications against each other and provide stable execution environments to VO users*.

7.3 Evaluation Summary

The requirements for XtreemOS Isolation have been sufficiently met, but are highly dependent on third-party mechanisms. The relevant security requirements have been met, given the assumption that the passphrases and session keys used by entities for key generation and secure communication are not derivable or disclosed.

The evaluation summary includes an assessment of XtreemOS isolation from the perspective of system management for the security mechanisms that implement isolation in XtreemOS. There are not many areas of value gain for isolation with XtreemOS, as the focus in the project has not been on developing such mechanisms. A discussion of variations for approaching isolation is given by Franke and Robinson[9]. A more technical evaluation of third-party Linux mechanisms for isolation has been presented in deliverable D2.1.6[6].

Installation complexity is reduced in comparison to Grid middleware, as there is no need for additional software and linking to higher-level software in order to implement isolation. For example, the Globus Toolkit has an additional service called the *Workspace Service*[12], which is used to implement a virtual execution environment in a Grid system. The implementation os isolation is based on standard container concepts implemented by SELinux. Containers can be linked with a GUID, GGID or GVID in order to implement isolation.

The need for human involvement in administration is similar to both Grid middleware and standard Linux. The instantiation and maintenance of e.g. container mechanisms is typically in the background and does not cause significant disruption to users.

Changes to application methods and user interfaces are not required, given that the isolation mechanisms are at a lower level (in some cases hardware). However, changes to user practices will typically arise, as the inclusion of isolation mechanisms typically results in different systems management practices being in place.

There are no real concerns for technology and vendor lock-in as a result of isolation mechanisms. XtreemOS can be integrated with most native isolation

mechanisms. XtreemOS does not bring significant enhancement to isolation of data, users or services in Grid systems. Focusing on isolation in particular is beyond the scope of XtreemOS security. However, by keeping this requirement in mind, it has influenced various design decisions concerning how the mapping of global accounts to local accounts and namespaces is done. For further discussion on this topic, see Delieverable D2.1.6[6] on "Evaluation of Linux Native Isolation Mechanisms for XtreemOS Flavours".

8 Conclusions

This document serves to create an evaluation framework for XtreemOS security, as this was the first evaluation. The evaluation framework was based on 5 classes of security requirements that are consistent with those established in the Grid security community. These requirements have been previously derived within WP4.2 and spawn multiple application domains. In addition, an emphasis on the management aspects has also been included when summarising the evaluation. XtreemOS security to date meets the requirements, although, in some cases, it is necessary to have high dependencies on the native OS or third party mechanisms. We have not discovered any significant deviations from standard security protocols and mechanisms established in the Grid Security Infrastructure specification. In fact, compliance with these standards has been achieved for certificate representation, as well as security services.

8.1 Soundness of XtreemOS Security

XtreemOS security uses established protocols and certificate formats as its basis. Identity and membership make use of standard X.509 certificates by adding additional attributes. The certificate distribution protocol is based on a protocol for mutual authentication using passphrases. This is limited by the strength of the passphrase and responsibility of the user not to leak the passphrase. However, the increased usability is the tradeoff, as users do not need to maintain and interface with certificate generation tools.

Data and communications security are also based on standard cryptographic mechanisms that are readily available in a base Linux system with which XtreemOS is integrated. Availability of the SSL software library is assumed, as well as checksum mechanisms including SHA and MD5.

Comprehensive resource management, accounting and isolation have not yet been fully implemented within the project but the groundwork for supporting these is already in place. For example, the ability to link global and local identities and accounts is fundamental for distributed accounting, as well as flexible, dynamic isolation. Node-level, native mechanisms for monitoring resource usage such as CPU, memory and storage are assumed. Native mechanisms are also assumed for creating and maintaining namespaces, containers and virtual machines that implement different forms of isolation. The innovations for XtreemOS security are therefore at a higher level.

8.2 Innovations and Relevance of XtreemOS Security

The first area of innovation in XtreemOS security is inherent from the overall project objective of integrating Grid mechanisms with OS mechanisms. This requires mechanisms for maintaining bindings between global and local entities. The XtreemOS-Cert is the first approach to doing this, where global identifier (GUID) and global group identifiers (GGID) are included as attributes. The PAM module integrated with native mechanisms maintains a mapping of these attributes to local Linux users and groups respectively.

The second innovation addresses the reduction of registration complexity and manual overhead that is involved in today's Grid systems. The standard practice is that for every user and machine that becomes a member of a Grid system, there is a need to manually install its identity certificate issued by a particular authority. XtreemOS security services (XVOMS and CDA) support online registration using passphrase-based mutual authentication. The combination of XVOMS and the CDA acts a CA, given that they are deployed within a single trusted domain. However, multiple CA's and hence XVOMS + CDA instances can be used for bootstrapping a XtreemOS system, given that a cross-certified hierarchical trust model is established between these CAs.

The third area of innovation is the inclusion of support for resource certificate authority and resource certification. In some system instances, machines must be issued with a machine attribute XtreemOS-Cert before being registered as part of the system. This provides some levels of assurance for users of resources concerning the validity of claims made by that resource. It also provides some basic means of contractual liability and non-repudiated traceability should a user's data or applications be compromised.

8.3 Future Evaluation

Future evaluations will use this same framework introduced in this deliverable. However, there is need to consider some aspects in a more quantitative manner, in collaboration with WP4.2 on applications and evaluation.

9 References

References

[1] Akmal Khan. Enterprise Linux Systems Management Headaches. LINUX For You Magazine, 2005. http://linux.sys-con.com; issue August 12;5.

[2] S. Cannon, S. Chan, D. Olson, C. Tull, V. Welch, and L. Pearlman. Using cas to manage role-based vo sub-groups. In *Proceedings of the 2003 Conference for Computing in High-Energy and Nuclear Physics (CHEP03), March 24-28, 2003, La*, 2003.

[3] XtreemOS Consortium.

[4] XtreemOS Consortium. In *XtreemOS public deliverables - D3.5.11*.

[5] XtreemOS Consortium. Design and implementation of node-level vo support. In *XtreemOS public deliverables - D2.1.2*. Work Package 2.1, November 2007.

[6] XtreemOS Consortium. Evaluation of linux native isolation mechanisms for xtreemos flavours. In *XtreemOS public deliverables - D2.1.6*. Work Package 2.1, November 2007.

[7] I. Foster, C. Kesselman, Gene Tsudik, and Steven Tuecke. A Security Architecture for Computation Grids. In *Conference on Computer and Communication Security*, 2001.

[8] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid Enabling Scalable Virtual Organizations. In *International J. Supercomputer Applications*, 2001.

[9] Carsten Franke and Philip Robinson. Autonomic provisioning of hosted applications with level of isolation terms. In *EASE '08: Proceedings of the Fifth IEEE Workshop on Engineering of Autonomic and Autonomous Systems (ease 2008)*, pages 131–142, Washington, DC, USA, 2008. IEEE Computer Society.

[10] Gridbus Project Team. Grid Computing Info Centre (GRID Infoware), 2008. http://www.gridcomputing.com/.

[11] John Linn. Trust Models and Management in Public-Key Infrastructures. http://citeseer.ist.psu.edu/386363.html.

[12] K. Keahey, I. Foster, T. Freeman, and X. Zhang. Virtual workspaces: Achieving quality of service and quality of life in the grid. *Sci. Program.*, 13(4):265–275, 2005.

[13] John C. Mitchell, Vitaly Shmatikov, and Ulrich Stern. Finite-state analysis of ssl 3.0. In *SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium*, pages 16–16, Berkeley, CA, USA, 1998. USENIX Association.

[14] S. Parthasarathy. Checksums : Your Best Friends For Security. LINUX For You Magazine, 2008. http://www.lfymag.com; issue August 01; Page 92-95.

[15] Gopalan Sivathanu, Charles P. Wright, and Erez Zadok. Ensuring data integrity in storage: techniques and applications. In *StorageSS '05: Proceedings of the 2005 ACM workshop on Storage security and survivability*, pages 26–36, New York, NY, USA, 2005. ACM.

[16] The Globus Team. Overview of the Grid Security Infrastructure, 2008. http://www.globus.org/security/overview.html.

[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] D. Wagner and B. Schneier. Analysis of the ssl 3.0 protocol. In *IProceedings of the 2nd USENIX Workshop on Electronic Commerce (EC-96)*, pages 29–40, Berkeley, CA, USA, 1996. USENIX Association.