Project no. IST-033576

XtreemOS

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

State-of-the-Art in trust & security for OS and Grids

D3.5.1

- WP 3.5 -

Due date of report: 30 Nov 2006
Actual submission date: 4 Dec 2006

Start date of project: June 1st 2006

Report Type:Deliverable
WP number:3.5

Ian Johnson
CCLRC

Version 3.0 / Last edited by Michael Wilson / 30/11/06

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	Sections Affected / Comments
1.0	16/10/06	M. Wilson, Alvaro Arenas	CCLRC	First draft
2.0	17/11/06		ICT	2.2, 3.4
3.0	30/11/06	M. Wilson	CCLRC	Updated for review comments

Contents

Summary

This document is a survey on trust and security in Grid systems.

The survey presents an overview of the different concepts and technologies relevant to trust and security in Grid systems. It analyses the relation between trust and security, describes trust and security challenges in the Grid, and introduces the existing mechanisms for managing trust and security. It relates these existing mechanisms to the requirements for the XtreemOS operating system to meet the needs for trust and security in Virtual Organisations.

1. Introduction

This survey presents an overview of the different concepts and technologies relevant to trust and security in Grid systems that relate the XtreemOS project.

The basic principle of protecting something, whether it is your mobile phone, a house, a bank, a museum, or a military installation, is to ensure that more resources are required to breach the security than are available to the attackers, or that in gathering or using those resources they leave enough information about themselves to be caught and prosecuted. No installation or system is totally secure. If enough resources are brought to bear on the problem any security system can be breached. The second part of the principle that information needs to be made available about attackers is essential to any sustainable security system in order to support forensic analysis.

A second principle is that there is no point in spending resources defending one part of a system disproportionately with another. For example, there is no point in making the front of a house secure at great expense, while leaving the back door open for the children to get in – since that is the point of weakness. In computing terms, the physical security of the resources should be considered as well as the information security, although, having made this point, these will not be considered further.

Following the first, basic principle, security systems are designed around three approaches:

- *Increase the resources required for access to an asset* – for example, limit the number of people with access credentials, partition the resource to minimise the scope of any credentials, separate duties between individuals to minimise the abuse an individual can make of credentials, minimise vulnerabilities open to misuse without appropriate credentials.

- *Increase the resources required to escape from a secure location* - frequently monitor access to reduce the time for unintended activity, close down the system when unintended activity is identified to catch the intruders, ensure restoration of assets accessed without intended authorisation.

- *Ensure that an information trail is left to trace intruders after accessing an asset* – ensure issue, transfer, transmission, storage and use of credentials/privileges is monitored and auditable.

Any individual security architecture is designed as a balance between the three approaches in order to permit the achievement of the functional requirements of the process involved. For example, a bank will allow the first approach to dominate since there is no functional requirement for end users to access the main resources, while a museum will allow the second approach to dominate since end users need access to the artifacts on display, whereas the third approach dominates in the situation where there is a need to catch not only the intruder, but those conspiring with him.

Computer security objectives are often described in terms of three overall objectives:

- *Confidentiality* (also known as secrecy), meaning that the computing system's assets can be read only by authorized parties.

- *Integrity*, meaning that the assets can only be modified or deleted by authorized parties in authorized ways.

- *Availability*, meaning that the assets are accessible to the authorized parties in a timely manner (as determined by the systems requirements). The failure to meet this goal is called a denial of service.

Some authorities separately identify non-repudiation as an objective; this is the ability to ``prove" that a sender sent or receiver received a message (or both), even if the sender or receiver wishes to deny it later. Privacy is sometimes addressed separately from confidentiality; some define this as protecting the confidentiality of a *user* (e.g., their identity) instead of the data. Most security objectives require identification and authentication, which is sometimes listed as a separate objective. Often auditing (also called accountability) is identified as a desirable security objective. Sometimes ``access control'' and ``authenticity'' are listed separately as well. For example, The U.S. Department of Defense (DoD), in DoD directive 3600.1 defines ``information assurance'' as ``information operations (IO) that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and nonrepudiation. This includes providing for restoration of information systems by incorporating protection, detection, and reaction capabilities.''

XtreemOS is developing an operating system where the Linux kernel and OS are extended to include the middleware of conventional Grid systems. The objective in extending the Linux kernel is that it increases the efficiency of what was previously middleware, and makes the programmer interface easier to use. There is no point in introducing security mechanisms that act directly counter to this objective – since it would negate the reason for developing the XtreemOS system. There is a conflict here between the way in which security requirements are met in the current Linux system, and the changes made to Linux by XtreemOS. Therefore, XtreemOS will have to modify other aspects of the security architecture to compensate.

For example, a common security principle mentioned above is that of partitioning – that there are different credentials/privileges required for each partition. Consequently, if an intruder enters one partition with a set of credentials then they are not able to enter other partitions with the same credentials. Also, partitions where unauthorised access puts the overall system at greater risk require greater effort resources to access them (greater privileges, more tests of identity etc...). This conflict between security partitioning and ease of use is particularly relevant to the motivation for XtreemOS. A trend in contemporary distributed systems is to ease their usability by reducing partitions through single sign on mechanisms which allow a user to present a single set of credentials and then access all the resources that they require from then on. In current Linux systems kernel operations require more credentials than shell or application operations – so there are security partitions which reduce the ease of use. XtreemOS is intended to move middleware operations into the kernel, thereby opening up the kernel to access through a single set of remotely applied credentials. This change removes the security benefits of the existing partitioning. The ease of use requirement overrides the security requirement and results in an inherently less secure system. To compensate for this weakness, it will be necessary to increase the monitoring, response to unauthorised activity and auditability of the trace of activity in order to maintain security. The security architecture will have to be extended to seek out, identify, and locate intruders; which will require modifications to some of the security tools (especially those proxy service daemons and event-driven auditors) to trace intruders back to their source, and otherwise maintain logs of data on intrusion attempts. This information can prove vital in taking an offensive stance against security break-in's and can help prosecute offenders.

The main body of the survey is organised as follows. The next section analyses the security of the existing Linux architecture and code, then the class of use of the XtreemOS system is described with respect to security requirements, then the relation between trust and security, and prepares the content of the rest of the survey. The core of the survey is Section 3, which analyses trust and security challenges in the Grid, and describes the existing mechanisms for managing trust and security. Section 4 describes the impact of trust and security across the architecture. Finally, Section 5 summarises some Grid projects tackling trust and security.

2. Security in the Linux OS

To introduce security mechanisms appropriate for the Grid into XtreemOS, they must be introduced into the existing Linux architecture. This chapter reviews the security mechanisms in the existing Unix and Linux architectures as a basis for this. The chapter is divided into three sections, the first outlines the generic UNIX security architecture, the second summarises the existing Linux security architecture, and the third describes secure Linux.

2.1 UNIX Network Security Architecture

For each of the layers in the UNIX Network Security Architecture (UNIX/NSA) model below, there is a subsection that follows that gives a brief description of that layer and some of the most widely used tools and methods for implementing security controls. The ISO/OSI style of model is used since most people in the UNIX community are familiar with it. This architecture is specifically based on UNIX Internet connectivity, but it is probably general enough to apply to overall security of any network methodology. One could argue that this model applies to network connectivity in general, with or without the specific focus of UNIX network security.

Layer	Name	Functional Description
LAYER 7	POLICY	POLICY DEFINITION AND DIRECTIVES
LAYER 6	PERSONNEL	PEOPLE WHO USE EQUIPMENT AND DATA
LAYER 5	LAN	COMPUTER EQUIPMENT AND DATA ASSETS
LAYER 4	INTERNAL-DEMARK	CONCENTRATOR - INTERNAL CONNECT
LAYER 3	GATEWAY	FUNCTIONS FOR OSI 7, 6, 5, 4
LAYER 2	PACKET-FILTER	FUNCTIONS FOR OSI 3, 2, 1
LAYER 1	EXTERNAL-DEMARK	PUBLIC ACCESS - EXTERNAL CONNECT

The specific aim of this model is to illustrate the relationship between the various high and low level functions that collectively comprise a complete security program for wide-area network connectivity. They are layered in this way to depict (a) the FIREWALL method of implementing access controls, and (b) the overall transitive effect of the various layers upon the adjacent layers, lower layers, and the collective model. The following is a general description of the layers and the nature of the relationship between them. After this brief discussion of what each layer is, the next section of this chapter will discuss examples of common methods and tools used to implement some of your options at each level, or at least try to tell you where to find out how to get started.

[7 - POLICY] is the umbrella that the entirety of a security program is defined in. It is this function that defines the policies of the organization, including the high level definition of acceptable risk down to the low level directive of what and how to implement equipment and procedures at the lower layers. Without a complete, effective, and implemented policy, a security program cannot be complete.

Everything discussed in layers one to five above involve specific things you can do, tools and techniques to implement, to address a particular area or "hole" in security. Your SECURITY POLICY is what ties all of that together into a cohesive and effective SECURITY PROGRAM. There are many issues to consider when formulating a policy, which alone is one of the biggest reasons why you must have one.

The questions to be addressed are manifold, but they include considering:

- What are the functional requirements?
- What assets need to be protected?

- What is the acceptable use policy?
- What are the legislative constraints and requirements?
- What incident reporting and forensic system should be in place?
- What is the policy for prosecution?

By answering these questions you determine what methods in layers one to five (or their equivalent) that you want to implement, and in what ways you want to modify or configure them. "A security policy is a formal specification of the rules by which people are given access to a computer and its resources." (and to extend that to say...a network and its resources). Whatever tools you install to help you maintain the security of your network and monitor it, they must be configured to implement YOUR POLICY, or else they are not doing the whole job that needs to be done. Therefore, you must first have a POLICY.

[6 - PERSONNEL] defines yet another veil within the bigger umbrella covered by layer 7. The people that install, operate, maintain, use, and can have or do otherwise have access to a network (one way or another) are all part of this layer. This can include people that are not in a single organization, or that a single individual may not have any administrative control over. The organisation's policy regarding personnel should reflect what the expectations are from the overall security program. Once everything is defined, it is imperative that personnel are trained and are otherwise informed of the policy, including what is and is not considered acceptable use of the system.

The local-area network layer [5 - LAN] defines the equipment and data assets that the security program is there to protect. It also includes some of the monitor and control procedures used to implement part of the security policy. This is the layer at which the security program starts to become automated electronically, within the LAN assets themselves.

The internal demarkation layer [4 - INTERNAL DEMARK] defines the equipment and the point at which you physically connect the LAN to the FIREWALL that provides the buffer zone between your local- area network (LAN) and your wide-area network (WAN) connectivity. This can take many forms such as a network concentrator that homes both a network interface for the FIREWALL and a network interface for the LAN segment. In this case, the concentrator is the internal demarcation point. The minimum requirement for this layer is that you have a single point of disconnect if the need should arise for you to spontaneously separate your LAN from your WAN for any reason.

The embedded UNIX gateway layer [3 - GATEWAY] defines the entire platform that homes the network interface coming from your internal demark at layer 4 and the network interface going to your packet filtering router (or other connection equipment) at layer 3. The point of the embedded UNIX gateway is to provide FIREWALL services (as transparent to the user or application as possible) for all WAN services. What this really is must be defined in your policy (refer to layer 1) and illustrates how the upper layers overshadow or are transitive to the layers below. It is intended that the UNIX gateway (or server) at this layer will be dedicated to this role and not otherwise used to provide general network resources (other than the FIREWALL services such as proxy FTP, etc.). It is also used to implement monitor and control functions that provide FIREWALL support for the functions that are defined by the four upper ISO/OSI layers (1-Application, 2-Presentation, 3- Session, 4-Transport). Depending on how this and the device in layer 2 is implemented, some of this might be merely pass-thru to the next level. The configuration of layers 3 and 2 should collectively provide sufficient coverage of all 7 of the functions defined by the ISO/OSI model. This does not mean that your FIREWALL has to be capable of supporting everything possible that fits the OSI model. What this does mean is that your FIREWALL should be capable of supporting all of the functions of the OSI model that you have implemented on your LAN/WAN connectivity.

Figure 1: Illustration of the UNIX/NSA Model

```
------------------------------------------------------------------
|                             POLICY                             |
------------------------------------------------------------------
                                |
                                |
---------------------------------------------------
|                   PERSONNEL                     |
---------------------------------------------------
                        |
                        |
---------------------------------
|              LAN              |
---------------------------------
           Enet |
           Enet |
        -----------------
        |  INTERNAL-D   |
        -----------------
           Enet |
           Enet |
-----------------   UNIX server with two Ethernet interfaces and
| GATEWAY-SERVER|   custom software and configuration to implement
-----------------   security policy (proxy services, auditing).
     Enet |
     Enet |
-----------------
| PACKET-FILTER |   cisco IGS router with access lists
-----------------
          X.25 |
               |
        -----------------
        |   EXTERNAL-D  |     leased DID line to WAN service
        -----------------
               |
               |
        + Public Access +
```

The packet filtering layer [2 - FILTER] defines the platform that houses the network interface coming from your gateway in layer 3 and the network interface or other device such as synchronous or asynchronous serial communication between your FIREWALL and the WAN connectivity at layer 1. This layer should provide both your physical connectivity to layer 1 and the capability to filter inbound and outbound network datagrams (packets) based upon some sort of criteria (what this criteria needs to be is defined in your policy). This is typically done today by a commercial off-the- shelf intelligent router that has these capabilities, but there are other ways to implement this. Obviously there is OSI link-level activity going on at several layers in this model, not exclusively this layer. But, the point is that functionally, your security policy is implemented at this level to protect the overall link-level access to your LAN (or stated more generally; to separate your LAN from your WAN connectivity).

The external demarkation layer [LAYER 1] defines the point at which you connect to a device, telephone circuit, or other media that you do not have direct control over within your organization. Your policy should address this for many reasons such as the nature and quality of the line or service itself and vulnerability to unauthorized access. At this point (or as part of layer 2) you may even deploy yet another device to perform point to point data link encryption. This is not likely to improve the quality of the line, but certainly can reduce your vulnerability to unauthorized access. You also need to be concerned about the dissemination of things at this level that are often considered miscellaneous, such as phone numbers or circuit IDs.

2.2 Security challenges to traditional Linux OS

The emergence of VO and Grid brings security challenges to traditional Linux operation system. There is an essential mismatch between the property of Grid system for distributed collaboration and the fundamental of Linux which is a local system on a single node in nature. This mismatch in basic design idea makes it difficult to support VO in traditional Linux. The challenges include almost all aspects involved in system design. Among all, the security challenges top others because their corresponding solutions are the foundations on which other solutions are built.

Security challenges for supporting distributed VO collaboration in Linux include two basic points. First, the problem of policy decision, i.e. “what kind of policies should be employed to support authentication, authorization and audit in VO lifecycle”, plays the core role of VO support. Second, the problem of policy enforcement, i.e. “what kind of infrastructure should be provided to support flexible policy enforcement”, impose the real challenge considering the system design of Linux. Strictly speaking, the solution to the first problem is not task of OS from the perspective of OS architecture. The solution to the second problem is more important to support different secure decisions.

We list the state-of-art of security related concepts in Linux. First, for the authentication, no kernel support exists. Only several integer values presenting user account, such as uid/gid, exist inside each process. This is an original and crude design, but it is proved useful in the past decades. Second, for the authorization, only simple file permission check exists. The widely-used DAC model dominates the current Linux authorization. In brief, each file object in itself is responsible for storing and providing the information used to check whether a user can be authorized to access this file. Unfortunately, this check is also crude with only coarse-grained user classification (owner/group/others) and operation classification (read/write/execute). An improvement is the ACL mechanism.

The state-of-art is that the policy definition and the policy enforcement are entangled with each other in Linux. The real challenge is to solve the problem of how to support flexible and versatile policy enforcement with guaranteed performance, easy implementation and simplified administrative complexity on such Linux kernel base. The good news is that the LSM framework and SELinux is proposed, but the bad news is that performance and overhead is still unclear.

2.3 Security Enhanced Linux

NSA's Information Assurance Directorate’s mission includes embedding information assurance measures directly into the emerging Global Information Grid. Recognizing the critical role of operating system security mechanisms in supporting security at higher levels, researchers from NSA's Information Assurance Research Group have been investigating an architecture that can provide the necessary security functionality in a manner that can meet the security needs of a wide range of computing environments.

The results of several previous research projects in this area have been incorporated in a security-enhanced Linux system by NSA. This version of Linux has a strong, flexible mandatory access control architecture incorporated into the major subsystems of the kernel. The system provides a mechanism to enforce the separation of information based on confidentiality and integrity requirements. This allows threats of tampering and bypassing of application security mechanisms to be addressed and enables the confinement of damage that can be caused by malicious or flawed applications.

The security mechanisms implemented in the system provide flexible support for a wide range of security policies. They make it possible to configure the system to meet a wide range of security requirements. The release includes a general-purpose security policy configuration designed to meet a number of security objectives as an example of how this may be done. The flexibility of the system allows the policy to be modified and extended to customize the security policy as required for any given installation.

2.3.1 Example Policy Objectives

Included with NSA's release is a general-purpose security policy configuration. It is not a complete security configuration. Its purpose is to provide a concrete example of how the security mechanisms in the system can be used. It provides a good starting point and should be customized to meet the specific needs of any site. Some of its objectives are outlined here. The example configuration controls access to various forms of raw data and protects the integrity of the kernel. It defines distinct types for the boot files, module object files, module utilities, module configuration files and sysctl parameters, and it defines separate domains for processes that require write access to these files. It defines separate domains for the module utilities, and it restricts the use of the module capability to these domains. It only allows a small set of privileged domains to transition to the module utility domains.

The example configuration protects the integrity of system software, system configuration information and system logs. It defines distinct types for system libraries and binaries to control access to these files. It only allows administrators to modify system software. It defines separate types for system configuration files and system logs and defines separate domains for programs that require write access.

The example configuration seeks to confine the potential damage that can be caused through the exploitation of a flaw in a process that requires privileges, whether a system process or privilege-enhancing (setuid or setgid) program. The policy configuration places these privileged system processes and programs into separate domains, with each domain limited to only those permissions it requires. Separate types for objects are defined in the policy configuration as needed to support least privilege for these domains. The configuration also attempts to protect privileged processes from executing malicious code. The policy configuration defines an executable type for the program executed by each privileged process and only allows transitions to the privileged domain by executing that type. When possible, it limits privileged process domains to executing the initial program for the domain, the system dynamic linker, and the system shared libraries. The administrator domain is allowed to execute programs created by administrators as well as system software, but not programs created by ordinary users or system processes.

Other objectives of the example configuration include protecting the administrator role and domain from being entered without user authentication, and preventing ordinary user processes from interfering with system processes or administrator processes by controlling the use of procfs, ptrace and signaling.

3. Virtual Organisations

The XtreemOS Grid Operating system will offer native support for virtual organization in Linux. The XtreemOS operating system is internally composed of two parts: XtreemOS foundation called XtreemOS-F and XtreemOS high level operating system services called XtreemOSG. XtreemOS-F is the modified Linux system embedding VO support mechanisms and providing an appropriate interface to implement XtreemOS-G services. XtreemOS-G is implemented on top of XtreemOS-F at user level, or more accurately, VO level, as users would then be supported at an even higher level. XtreemOS-G consists of services for security, data and application management all based on a common infrastructure for highly available and scalable services.

The issue of what a Virtual Organisation is, has been addressed elsewhere in XtreemOS. Within XtreemOS a VO has been defined as:

> A Virtual Organisation is a coalition of entities that pool resources to achieve common objectives. The coalition can be temporary or permanent. The entities can be individuals, groups, organisational units or entire organisations and are normally geographically dispersed. There usually will be legal or contractual arrangements between the entities. The resources can be physical equipment such as computing or other facilities, or other capabilities such as knowledge, information or data

Here we need to address the consequences of that choice for the security system. The main point is to which entities can authentication be granted, and the consequence of that drives the security policies relating to what actions can be taken after a security failure. There are legislative constraints on what can be done in the extreme case, but there are other actions that can be taken depending on who is permitted to join a VO.

If a VO member breaks the security policies then the following actions can be taken:

1) Issue a warning
2) Escalate warning to a more authoritative person in the VO members organization
3) Increase monitoring of behaviour and recording in an auditable store
4) Issue a fine
5) Remove authorization for actions in VO & remove from VO
6) Prosecute under legislation

To perform actions 4 or 6 the entity must have a legal identity & legal personality to enable enforcement. Therefore the entity must be one of:

- Individual
- Partnership
- Public body
- Incorporated body

An incorporated body is usually a company, but could be created by other means permitted under the laws of the state in which the VO is constituted and under whose jurisdiction legal enforcement would take place. Public bodies have a wide range of legal identities ranging from the nation state itself to independent incorporated bodies such as those created under Royal Charter in the UK.

A consequence of this is that a VO cannot contain another VO as a member unless they are incorporated, if legal enforcement of the actions to manage the VO is required.

A second consequence of this need to enforce management actions, is that VO membership must be created through a legal contract that can be enforced, and which defines the security policies and how they will be enforced.

Actions beyond warnings require the evidence to justify them if legal action is taken, which must be recorded through step 3). Because of this, the auditable recording of actions becomes a requirement for XtreemOS.

The definition of a VO allows for a choice of business relationships to pertain between members. These will now be considered.

3.1 Business Relationship Risk Management

Any organization entering into a business relationship is exposed to risks. Organisations wish to manage or minimize those risks. Contracts are one management technique, but they are only useful if the relationship is monitored, and the monitoring is recorded as evidence to enforce the contract.

In existing monolithic organizations the risks are managed within the organization through employment contracts with staff, and through internal management of resources.

When organizations collaborate, contracts/agreements are created between the organizations which define what is to be provided, when and at what price, but they also contain terms and conditions about confidentiality and inspectability, with others that define what actions can be taken when delivery time or quality are not met. However, the risk is still managed by the contractee who must decide when to call upon these penalty clauses.

Figure 2 from Lutz Heuser of SAP shows the migration path expected from the single corporate system to a configuration as a VO showing the distribution of data resources between the member organizations, and the interactions required to manage the risks in the business relationships through mutual inspection of each other’s processes.

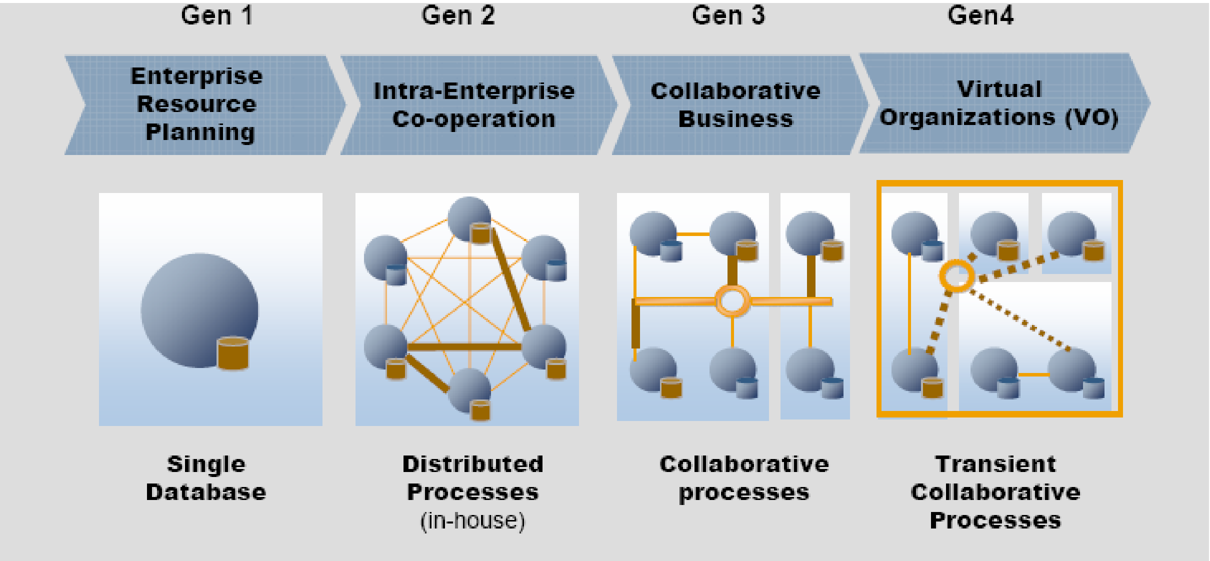

Gen 1
Gen 2
Gen 3
Gen4
Enterprise Resource Planning
Intra-Enterprise Co-operation
Collaborative Business
Virtual Organizations (VO)
Single Database
Distributed Processes (in-house)
Collaborative processes
Transient Collaborative Processes

Figure 2: The migration from single ERP system to distributed VO management system.

From the security perspective this need for monitoring establishes a trade-off between the privacy/confidentiality of the contractor and the need for inspection by the contractee. Security mechanisms are required to be put in place to ensure that both the agreed monitoring can take place, and that the agreed confidentiality is enforced – X.509 proxy certificates are currently used in several systems to provide an authorisation mechanisms to meet these access constraints.

3.2 Business Relationship Topology

The topology of the VO is chosen as a result of resolving the trade-off between the confidentiality of each VO member, and the need to monitor each other's performance. From the security perspective, the VO topology defines the access that is required between VO members to information about each other, and therefore the authorisations that are required.

There have been several proposals to characterise VO topologies [Bur99, Kat00]. For instance, Burn et al [Bur99] defines six types of VOs, ranging from organisations providing services in the web (such as web shops or newspapers on the web) which does not control any user of the service to dynamic networks of entities collaborating to meet market opportunities. We present here a simple topology of VOS introduced initially by Katzy et al [KatXX], based mainly on the network topology (see Figure 3):

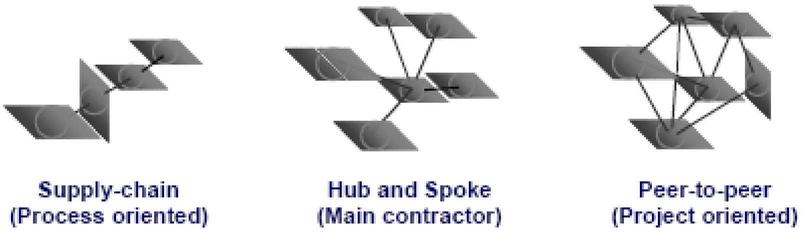


Figure 3. Types of VOs

- **Supply-Chain VOs**
  In a supply-chain topology, the partners' interaction pattern mainly follows a chain, relating mainly to its upper and lower neighbours. Historically, a supply chain was made up of a series of pairwise contracts where each supplier met the needs of their customer, and disclosed information only between themselves. Current supply chains are moving more to hub and spoke relationships where the organisation at the supply chain head wishes to look into the information of not only their tier 1 suppliers, but also into tier 2..n suppliers to ensure that quality is maintained.

- **Hub and Spoke VOs (star or main contractor VOs)**
  In a star topology, partners interact with one central hub or strategic centre. This type of VOs corresponds to a coordinated network of interconnected members, where each member provide key functionalities, and distinguished member plays the role of a leading actor (star), coordinating the whole operation of the VO.

- **Peer-to-Peer VOs**
  Partners in peer-to-peer topology have multiple relationships between all nodes without hierarchy. This management approach is common in academic projects, but once IPR needs to be protected, then pairwise contracting of the supply chain variety, or where the prime contractor dominates the organisations then a hub and spoke topology arises.

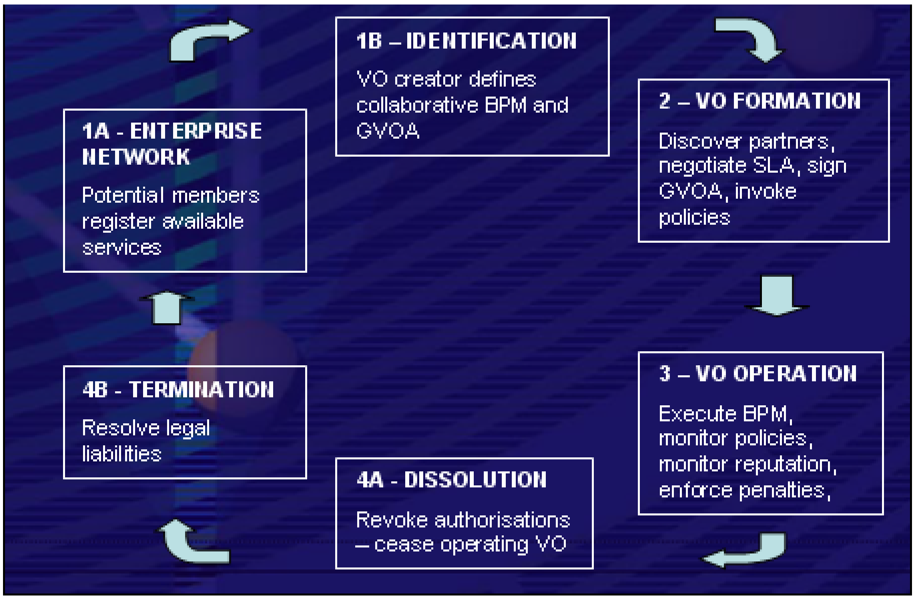
From an organizational perspective these VO types describe the main coordination structure that governs information and material flows as well as the power relationships and decision making within the network and projects.

The relationship topology of a VO has also an impact on its management, business processes and IT structure at the same time. VOs, which adopt a supply chain topology, usually use Supply Chain Management and Efficient Consumer Response (ECR) to improve inter-organisational co-ordination and control. Integration of information flow (e.g. EDI) and material flow creates transparency in the entire value chain and reduces waste and doubles effort in the virtual enterprise. In contrast, VOs with peer-to-peer topology are built on self-organization.

Whichever topology of VO is used, it will have to go through the standard lifecycle from creation to dissolution, and security requirements vary during this lifecycle.

3.3 The Virtual Organisation Lifecycle

The VO Roadmap project [Cam03] developed a VO lifecycle including phases such as identification, formation, operation/evolution and dissolution. The identification phase is dealing with setting up the VO; this includes selection of potential business partners by using search engines or looking up registries. VO formation deals with partnership formation, including the VO configuration distributing information such as policies, agreements, etc, and the binding of the selected candidate partners into the actual VO. After the formation phase, the VO can be considered to be ready to enter the operation phase where the identified and properly configured VO members perform accordingly to their role. Membership and structure of VOs may evolve over time in response to changes of objectives or to adapt to new opportunities in the business environment. Finally, the dissolution phase is initiated when the objectives of the VO has been fulfilled.


1B – IDENTIFICATION

VO creator defines collaborative BPM and GVOA

2 – VO FORMATION

Discover partners, negotiate SLA, sign GVOA, invoke policies

3 – VO OPERATION

Execute BPM, monitor policies, monitor reputation, enforce penalties,

4A - DISSOLUTION

Revoke authorisations – cease operating VO

4B - TERMINATION

Resolve legal liabilities

1A - ENTERPRISE NETWORK

Potential members register available services

Figure 4: The VO Lifecycle

The TrustCoM project has derived security and trust requirement by analysing the lifecycle of a VO [Are05, Wes05]. Here we summarise such requirements. The first and last stages are divided into two in Figure 4 to enforce the need for an infrastructure in which organisations make an offer their role in potential VOs, and for the long term resolution of the legal liabilities of the VO which may persist long after it has dissolved – for example the liabilities of builders resulting from including asbestos in their developments persisted decades after the contract was complete.

VO Identification

The identification phase addresses setting up the VO - this includes selection of potential business partners from the network of enterprises by using search engines or looking up registries. Generally, relevant identification information contains service descriptions, security grades, trust & reputation ratings, etc. Depending on the resource types, the search process may consist in a simple matching (e.g., in the case of computational resources, processor type, available memory and respective data may be considered search parameters with clear cut matches) or in a more complex process, which involves adaptive, context-sensitive parameters. For an example, the availability of a simulation program may be restricted to specific user groups or only for certain data types, like less confidential data, etc. The process may also involve metadata such as security policies or Service Level Agreement (SLA) templates with ranges of possible values and/or dependencies between them, such as bandwidth depending on the applied encryption algorithm. The identification phase ends with a list of candidates that potentially could perform the roles needed for the current VO.

After this initial step from the potentially large list of candidates, the most suitable ones are selected and turned into VO members, depending on additional aspects that may further reduce the set of candidates. Such additional aspects cover negotiation of actual Quality of Service (QoS) parameters, availability of the service, "willingness" of the candidate to participate, etc. It should be noted that though an exhaustive list of candidates may have been gathered during the identification phase, this does not necessarily mean that a VO can be realised - consider the case where a service provider may not be able to keep the promised SLA at a specific date due to other obligations.

In principle, the intended formation may fail due to at least two reasons: (a) no provider (or not enough providers) is able to fulfil all given requirements comes to SLA, security, etc. or (b) providers are not (fully) available at the specified time. In order to circumvent these problems, either the requirements may be reduced ("choose the best available") or the actual formation may be delayed to be re-launched at a more suitable time. Obviously there may be the case, where a general restructuring of the requirements led to a repetition of the identification phase.

VO Formation

At the end of the (successful) identification phase the initial set of candidates will have been reduced to a set of VO members. In order to allow these members to perform accordingly their anticipated role in the VO they need to be configured appropriately. During the formation phase a central component such as a VO Manager distributes the VO level configuration information, such as policies, SLAs, etc. to all identified members. These VO level policies need to be mapped on local policies. This might include changes in the security settings (e.g. open access through a firewall for certain IP addresses, create users on machines on the fly, etc.) to allow secure communication or simply translation of XML documents expressing SLAs or Obligations to a product specific format used internally.

VO Operation

The operational phase could be considered the main life-cycle phase of a VO. During this phase the identified services and resources contribute to the actual execution of the VOs task(s) by executing pre-defined business processes (e.g. a workflow of simulation processes and pre- and post processing steps). A lot of additional issues related to management and supervision are involved in this phase in order to ensure smooth operation of the actual task(s). Such issues cover carrying out financial arrangements (accounting, metering), recording of and reacting to participants' performance, updating and changing roles and therefore access rights of participants according to the current status of the executed workflow, etc. In certain environments persistent information of all operations performed may be required to allow for later examination e.g. to identify fault-sources.

Throughout the operation of the VO, service performance will be monitored. This will be used as evidence when constructing the reputation of the service providers. Any violation – e.g. an unauthorised access detected by the access control systems- and security threats –e.g. an event detected by an intrusion detection system- need to be notified to other members in order to take appropriate actions. Unusual behaviours may lead to both a trust re-assessment and a contract adaptation. VO members will also need to enforce security at their local site. For example, providing access to services and adapting to changes and the violations.

Evolution is actually part of the operational phase: as participants in every distributed application may fail completely or behave inappropriately, the need arises to dynamically change the VO structure and replace such partners. This involves identifying new, alternative business partner(s) and service(s), as well as re-negotiating terms and providing configuration information as during identification, respectively formation phase. Obviously one of the main problems involved with evolution consists in re-configuring the existing VO structure so as to seamlessly integrate the new partner, possibly even unnoticed by other participants. Ideally, one would like the new service to take over the replaced partners' task at the point of its leaving without interruption and without having to reset the state of operation. There may other reasons for participants joining or leaving the VO, mostly related to the overall business process, which might require specific services only for a limited period of time - since it is not sensible to provide an unused, yet particularly configured service to the VO for its whole lifetime, the partner may request to enter or leave the VO when not needed.

VO Dissolution

During the dissolution phase, the VO structure is dissolved and final operations are performed to annul all contractual binding of the partners. This involves the billing process for used services and an assessment of the respective participants' (or more specifically their resources) performances, like amount of SLA violations and the like. The latter may of particular interest for further interactions respectively for other potential customers. Additionally it is required to revoke all security tokens, access rights, etc. in order to avoid that a participant may (mis)use its particular privileges. Generally the inverse actions of the formation phase have to be performed during Termination. Obviously partial termination operations are performed during evolution steps of the VO's operation phase.

3.4 Quality of Service in VO Formation, Monitoring, Policy

When forming a VO we search for appropriate services within organizations that could execute our workflow. Besides their appropriateness (i.e., their ability to execute our tasks) we are also interested in their reputation – can we trust them to meet the required performance (in terms of execution times, availability, etc.) constraints. To this end we employ the SLAs, i.e., the required Quality of Service.

To ensure that the Quality of Service is a) met, b) that the organizations are not advertising services with misleading properties and c) that the users of the services are not exceeding the agreed QoS, we need to implement security measures which prevent tampering with them.

This is done via ensuring secure way of storing:

- The properties of the available services
- The properties are set by the organization who owns the service.
- The reputation of the organizations/services is updated securely. They do not have the access to the reputation data about their services.
- The monitoring process is secure. Any tampering with this process must be detectable.
- The services' policy and the corresponding services (e.g., PEP) must be secured.

The above described measures enforce the availability of the real data which is to be used in finding the appropriate services for the new VO. When running a VO, the tampering within the services policy is also prevented, hence preventing the users to misuse them by enforcing their policies. These security measures also prevent widespread breaches with use of rogue services which only collect sensitive data.

Another view on Quality of Service in VO Formation, Monitoring and Policy is to look at the trustworthiness of different organizations. When forming a new VO, all the participating organizations need to be trusted – their reputation must be above certain level. When these are sufficient and with security measures, defined above, we can be reasonably certain that our requirements will be met and that our resources won't be misused.

When using complex systems that heavily rely on the authentication, authorization and delegation, we must always assume that these security systems may break. It is possible that they are under DoS attack or that the node (nodes) broke down. In these cases the agreed upon QoS will obviously fail, hence we need a forensic tool for matching which service consequently failed and notify the corresponding users. The same applies when services don't have proper authorizations.

3.5 Dynamic VOs

The definition of the VO at the start of this chapter stated that:

> The coalition can be temporary or permanent. The entities can be individuals, groups, organisational units or entire organisations and are normally geographically dispersed.

It may appear that much of the discussion since then of risks, topologies and lifecycles and limited this definition to large organisations with long lifecycles. This is not the intention.

Whether a VO exists for a month or 30 years similar high level risks exist in the relationship and need to be managed. The basic mechanisms of management will require knowing what is to be done, and monitoring whether it is being done. A general design must provide the mechanisms to support the policies for the management of the VO.

4. Trust and Security

In normal personal life, *trust* is exemplified by cinema clichés such as:

1 Parent: "Where are you going tonight?"
2 Adolescent child: "So you don't trust me!"
3 Parent: "I care about you."

The implication in this cliché is that when one party trusts another they assume that their high level *intentions* are similar, or at least certainly not in conflict, and that they have *competence* to achieve those intentions. When trust is not sufficient, then monitoring – in this example questioning of actions – is increased, so that undesirable actions can be averted or immediately corrected. A second implication (from statement 2) of this cliché is that *monitoring* is in conflict with *privacy* for the individual. For organisations *confidentiality* is the equivalent of privacy for the individual. A third implication (from statement 3 as a justification) is that perceived *risk* increases the *confidence* required in both the *intentions* held and the *competence*.

In the psychology, sociology and economics literature *trust* is contrasted with *trust substitutes*. Security measures, monitoring, records of past performance, contracts, service level agreements are all substitutes for trust in that they provide agreements on what is expected be done and data as to whether it is being done as expected. Contracts also state the limits on monitoring by *confidentiality* between the parties. Organisations implement trust substitutes since they define the risks, and the mechanisms to monitor and manage them without relying on trust alone. Trust substitutes merely constrain the risks, they do not eliminate them. Even in a well defined business relationship, trust is still required when things occur that were not planned for the initial contracts.

In business the method used for developing the trust that is required to address the unforeseen risks relies upon human relationships. When organisations are involved in a relationship there are often personal relationships between individuals in each organisation. Frequently these relationships are between senior staff who have established social networks that provide values outside business which can be drawn on when trust is required in these unforeseen circumstances. No automated equivalent to these human relationships has been defined in the world of computer mediated business relationships.

In the computing world there is a desire to make trust a single binary choice – is an entity trusted or not? The decision can be based on data of past performance of an entity, on recommendations as to the trustworthiness of an entity, or a combination of the two. Abstractions of past performance are used to determine the *reputation* of an entity, which contributes to the judgement of trust in this case. Reputation in this sense is a gross simplification of the procedures that most business organisations use for *supplier qualification* evaluation [Riorden96].

The notions of *trust* and *competence* are often confounded in this choice, particularly when third party *recommendation* is introduced as a basis for making the decision. The *scope* of any judgement of *competence* is essential to its application – for example, a national medical body will identify those doctors which are deemed competent to practice, but, further refinements of medical speciality will be defined by other specialist registers, e.g. a register of surgeons. It is often common that a selection is needed of a supplier of VO partner to perform a role which does not match any for which past data of competence is available. In this case, the selection will be based on an inference that because an agent was competent at something before, then their competence will transfer to the new role. In this case the argument is complex as to which set of competences are required, but an element of *trust* is also required, in as far as the new partner would monitor their own performance and either modify their own actions or report their failure if necessary. It is usual that the confounding of *trust* and *competence* is appropriate when the scope of the new *role* is in any way different from the roles previously performed.

In the extreme computing case, *trust* refers to the binary choice of whether the credentials of a user that have been issued by a certification authority (CA) should be accepted. In this example, if the CA is trusted, then the credentials are accepted, whereas the credentials are rejected if the authority is not trusted. The CA will provide credentials when a user is tested to meet a certification policy (e.g. [EuroPKI004]). The CA may issue credentials in accordance with its policy itself, or this role may be delegated to a registration authority to ensure that the duties are separated in order to increase security. In either case the certificate states that certification policy has been applied, and then the decision as to whether to trust a CA, becomes one of whether the certification policy of that CA meets the requirements of the person who is deciding whether to trust that CA. Certification policies themselves need to be inspected to determine trust in this case. Therefore certification policies can themselves be stated in a machine readable form by a CA to be compared with certification requirements by a VO in order to determine if a CA will be trusted. Once trusted then the CA is listed as a trusted authority until the CA either changes the policy, or the performance of the CA shows that they are not acting in accordance with the published certification policy. At this point in the process the more general judgement of trust of a CA is required in terms of their intentions (as stated in the certification policy) and their competence (as assessed from a record of their performance).

In the Internet world, trust has been recognised as an important aspect of decision making for electronic commerce [Gra00, Jos05]. Customers must trust that sellers will provide the services they advertise, and will not disclose private customer information (name, address, credit card details, etc). Trust in the supplier's competence and honesty will influence the customer's decision as to which supplier to use. Sellers must trust that the buyer is able to pay for goods or services, is authorised to make purchases on behalf of an organisation or is not underage for accessing service or purchasing certain goods.

How is the situation in the Grid? Fundamental to the Grid definition is the idea of *resource sharing* [Fos01]. The Grid was initiated as a way of supporting scientific collaboration, where many of the participants knew each other. In this case, there is an implicit trust relation, all partners have a common objective –for instance to realise a scientific experiment- and it is assumed that resources would be provided and used within some defined and respected boundaries. However, when the Grid is intended to be used for business purposes, it is necessary to share resources with unknown parties. Such interactions may involve some degree of risk since the resource user cannot distinguish between high and low quality resource providers on the Grid. The inefficiency resulting from this asymmetry of information can be mitigated through trust mechanisms.

This section analyses the concept of trust and its relation with security. There is a vast source of information on the theory and application of trust, For instance [Cas00, Wai02, Nix03, Jen04, Her05]. Here we visit the main definitions of trust and study the relation between trust and security.

4.2 Trust Definitions

This report focuses on trust in the context of networked and distributed computing systems. In this context, the remote system needs to be trusted, as well as interactions over underlying services such as communication services. As expressed by Grandison and Sloman [Gra00], the significance of incorporating trust in distributed systems is that trust is an enabling technology. Its inclusion will enable secure electronic transactions.

There is not consensus in the literature on what trust is [McK96]; it is recognised as an important and complex subject relating honesty, truthfulness, competence, reliability, etc. of the trusted person or service.

One of the influential works towards a practical definition of trust is given by Gambetta [Gam00b]: "*When we say we trust someone or that someone is trustworthy, we implicitly mean that the probability that he will perform an action that is beneficial or at least not detrimental to us is high enough for us to consider engaging in some form of cooperation with him. Correspondingly, when we say that someone is untrustworthy, we imply that that probability is low enough for us to refrain from doing so.*" Gambetta's definition stresses that trust is fundamentally a belief or estimation, which has inspired the use of subjective logic as a way of measuring trust [Jos99]. Castelfranchi and Falcone [Cas98] extend Gambetta's definition to include the notion of competence along with predictability.

Kini and Choobineh [Kin98] examine trust from the perspectives of personality theory, sociology, economics and social psychology. They highlight the implications of these definitions and combine their results to create their definition of trust in a system. They define trust as: *"a belief that is influenced by the individual's opinion about certain critical system features".* Their analysis covers various aspects of human trust in computer dependent systems but they do not address the issue of trust between parties (humans or processes) involved in e-commerce transactions.

In the Trust-EC[1] project of the European Commission Joint Research Centre (ECJRC), Jones [Jon99] defines trust as *"the property of a business relationship, such that reliance can be placed on the business partners and the business transactions developed with them".* Jones states as relevant issues such as the *identification* and *reliability* of business partners; the *confidentiality* of sensitive information; the *integrity* of valuable information; the prevention of *unauthorised* copying and use of information; the guaranteed *quality* of digital goods; the *availability* of critical information; the management of *risks* to critical information; and the *dependability* of computer services and systems.

Grandison and Sloman [Gra00] survey various definitions of trust. Following a brief analysis of these definitions, they build their own one as *"the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context".* They argue that trust is a composition of many different attributes - *reliability, dependability, honesty, truthfulness, security, competence* and *timeliness* - which may have to be considered and defined depending on the environment in which trust is being specified.

Dimitrakos [Dim01] has defined trust as follows: *"Trust of a party A in a party B for a service X is the measurable belief of A in B behaving dependably for a specified period within a specified context in relation to X".* In his definition, a *party* can be an individual entity, a collective of humans or processes, or a system; the term *service* is used in a deliberately broad

[1] http://dsa-isis.jcr.it/TrustEC

sense to include transactions, recommendations, issuing certificates, underwriting, etc; *dependability* is used broadly to include security, safety, reliability, timeliness, and maintainability; a *period* may be the duration of the service, refers to the past, future (a scheduled or forecasted critical time slot), or always; finally, the term *context* refers to the relevant service agreements, service history, technology infrastructure, legislative and regulatory frameworks that may apply.

Josang, Ismail and Boyd [Jos05] define trust as *"the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of a relative security, even though negative consequences are possible"*. They argue that their definition includes aspects such as *dependence* on the trusted entity or party; the *reliability* of the trusted entity or party; *utility* in the sense that positive utility will result from a positive outcome, and negative utility will result from a negative outcome; and a certain *risk attitude* in the sense that the trusting party is willing to accept the situational risk resulting from the previous elements.

Some aspects of these definitions are common, other are complementary. For example, [Gam00b] emphasises that trust is in part subjective, a characteristic present in other definitions such as [Gra00], [Dim01] and [Jos05]. [Gra00] underlines that trust is a belief in the competence of an entity within a specified context, while [Kin98] lay stress on that the entity that manifests trust (the "trustor") is the human - not the system. The definition in [Jon99] focuses on the aspect that in commerce *trust is relative to a business relationship*. One entity may trust another entity for one specific business and not in general. Such business relationship can be seen as the context of [Gra00] definition. Finally, the definition in [Dim01] highlights an important point, trust evolves in time and is measurable.

We do not intent to provide a definition of trust, rather to show the diversity of definitions and those points in common: subjective, context and evolution in time, among others. In the next part we analyse how trust is related to security, for the case of distributed systems.

4.3 Relating Trust to Security

In general, the purpose of security mechanisms is to provide protection against malicious parties. Traditional security mechanisms typically protect resources from malicious users by restricting access to only authorised users. However, in many situations within distributed applications one has to protect oneself from those who offer resources so that the problem is in fact reversed. For instance, a resource providing information can act deceitfully by providing false or misleading information, and traditional security mechanisms are unable to protect against this type of threat. As noted in [Jos05], trust systems can provide protection against such threats. The difference between these two approaches to security was first described by Rasmusson and Janssen in [Ras96] who used the term hard security for traditional mechanisms like authentication and access control, and soft security for what they called social control mechanisms, of which trust is an example.

Grandison and Sloman [Gra00] have defined a trust classification as a useful way of categorising the literature relating to trust in Internet services. We have found such taxonomy helpful in linking trust and security for the purpose of this work. Trust is specified in terms of a relation between a *trustor*, the subject that trusts a target entity, and a *trustee*, the entity that is trusted. [Gra00] defines the following classes of trust.

- **Service Provision Trust** describes the relying party's trust in a service or resource provider. The trustor trusts the trustee to provide a service that does not involve access to the trustor's resources.

This type of trust is essential for Grids, and can be seen as a minimal trust requirement in dynamic Virtual Organisations (VOs). Many Grid applications assume this type of trust implicitly; a partner in a VO presupposes a service provision trust as a result of participating in VO, although the VO does not provide mechanisms to enforce it. The EU project TrustCoM [Dim04] is developing mechanisms to enforce this type of trust.

In general, service provision trust is related to the *reliability* or the *integrity* of the trustee. For instance, in e-banking the customer trusts the bank to support mechanisms that will ensure that passwords are not divulgated, and to maintain the *privacy* of any information such as name, address and credit card number. The Liberty Alliance Project[2] uses the term "business trust" to describe a provision trust, a mutual trust between companies emerging from contract agreements that regulate interaction between them [Boe03]. Mobile code and mobile agent-based applications also include service provision trust; the mobile code trusts the execution environment provided by the remote system.

- **Resource Access Trust** describes trust in principals for the purpose of accessing resources owned by the relying party. A trustor trusts a trustee to use resources that he own or controls. Resource access trust has been the focus of security research for many decades [Abr95], particularly on mechanisms supporting access control. Generally, resource access trust forms the basis for specifying authorisation policies, which then are implemented using access control mechanisms, firewall rules, etc.

  [Gra00] highlights the distinction between trusting an entity to read or write a file on your server and trusting an entity to execute code within your workstation. Simple file access requires that the trustee will follow the correct protocol, will not divulge information read, and will write only correct data, etc. Allowing an entity to execute code on your workstation implies much higher level of trust. The code is expected not to damage the trustor's resources, to terminate within reasonable finite time and not to exceed some defined resource limits with respect to memory, processor time, local file space, etc. [Sur02] has also drawn the attention to the case of trusting an entity to execute remote code in Grids; it shows practical examples of the possible consequences how to minimise dangers.

- **Delegation Trust** denotes the case when a trustor trusts a trustee to make decisions on his behalf, with respect to a resource or service that the trustor owns or controls.

  Although delegation is conceptually simple, designing and deploying it within a Grid environment has proved to introduce problems regarding security. Such security implications have been analysed by Broadfoot and Lowe in [Bro03a], work carried out in the context of the EU DataGrid project. A point that is addressed is the level of trust assumed when delegation is employed, in particular the effect of having *onwards delegation*. They also investigate all the security implications for two delegation mechanisms widely used in Grids: delegation chaining [Gas90] and call-back delegation [Fos98].

- **Certification Trust** is based on the certification of the trustworthiness of the trustee by a third party, so trust would be based on a criteria relating to the set of certificates presented by the trustee to the trustor.

  Trust systems that derive certification trust are typically authentication schemes such as X.509 and PGP [Zim95]. This class of trust is called "authentication trust" in Liberty Alliance [Boe03] and "identity trust" in [Jos05]. Grandison [Gra00] views certification trust as a special form of service provision trust, since the certification authority is in fact providing a trust certification service; however Josang [Jos05] views certification trust and service provision trust as two layers on top of each other, where provision trust normally cannot exist without certification trust; in the absence of certification trust, it is only possible to have a baseline provision trust in an entity.

  Certification trust has played an important role in Grid environments; it is present with the inclusion of certification authorities, which play a central role in the Grid Security Infrastructure [Nag03] and have been exploited in production Grids [Joh03].

- **Context Trust** describes the extend to which the relying party believes that the necessary systems and institutions are in place in order to support the transaction and provide a safety net in case something should go wrong. It refers to the base context that the trustor must trust. This type of trust is called infrastructure trust in [Gra00], here we prefer to use the broader term of context trust used by [Jos05], which also involves social and legal factors such as insurance and legal system and law enforcement.

[2] http://www.projectliberty.org

The main motivation of Grandison and Sloman’s classification is to define classes of high-level trust specifications, which may be refined to low-level implementation policies, such as policies about access control, authentication and encryption [Gra03]. Gambetta [Gam00a] has highlighted that to make a society prosper, one needs rules (both written and unwritten), understanding of good and bad behaviour with its consequences and accountabilities, initial trust and earned trust, identification of the risks associated with transactions, and so on. As mentioned in [Sie05], a similar view should be taken if we want to achieve a secure Grid society. Many of the rules of the secure Grid society can be expressed in the form of trust specifications, which can consequently be refined into policies.

5 Trust and Security in a Grid Environment

5.1 Trust and Security Requirements in the Grid

The Virtual Organisation (VO) is a key concept in the Grid community. A VO can be seen as a temporary or permanent coalition of geographically dispersed individuals, groups, organisational units or entire organisations that pool resources, capabilities and information to achieve common objectives. Depending on the context, dynamic ensembles of the resources, services, and people that comprise a scientific or business VO can be small or large, short- or long-lived, single- or multi-institutional, and homogeneous or heterogeneous. Trust and security challenges within the Grid environment are driven by the need to support scalable, dynamic distributed VO [Fos01].

The GGF has initiated the definition of the next-generation of Grid middleware by extending the emerging Web services technology that is currently being developed across the IT industry, under the umbrella of the Open Grid Services Architecture (OGSA). Trust and security requirements can be analysed from different perspectives. This section analyses requirements as defined by the GGF OGSA Security Workgroup, as well as through the different phases of a Virtual Organisation.

Security Challenges According to GGF

The GGF OGSA Working Group has submitted a memo proposing a strategy for addressing security with OGSA [Nag03]. According to the group, the security challenges faced in a Grid environment can be grouped into three categories:

- integration solutions where existing services needs to be used, and interfaces should be abstracted to provide an extensible architecture;
- interoperability solutions so that services hosted in different virtual organizations that have different security mechanisms and policies will be able to invoke each other; and
- solutions to define, manage and enforce trust policies within a dynamic Grid environment.

A solution within a given category will often depend on a solution in another category. For example, any solution for federating credentials to achieve interoperability will be dependent on the trust models defined within the participating domains and the level of integration of the services within a domain. Defining a trust model is the basis for interoperability but trust model is independent of interoperability characteristics. Similarly level of integration implies a level of trust as well as a bearing on interoperability.

In a Grid environment, where identities are organized in VOs that transcend normal organizational boundaries, security threats are not easily divided by such boundaries. Identities may act as members of the same VO at one moment and as members of different VOs the next, depending on the tasks they perform at a given time. Thus, while the security threats to OGSA fall into the usual categories (snooping, man-in-the-middle, intrusion, denial of service, theft of service, viruses and Trojan horses, etc.) the malicious entity could be anyone. An additional risk is introduced, when multiple VOs share a virtualized resource (such as a server or storage system) where each of participating VOs may not trust each other and therefore, may not be able to validate the usage and integrity of the shared resource.

The Integration Challenge

For both technical and pragmatic reasons, it is unreasonable to expect that a single security technology can be defined that will both address all Grid security challenges and be adopted in every hosting environment. Existing security infrastructures cannot be replaced overnight. For example, each domain in a Grid environment is likely to have one or more registries in which user accounts are maintained (e.g., LDAP directories); such registries are unlikely to be shared with other organizations or domains. Similarly, authentication mechanisms deployed in an existing environment that is reputed secure and reliable will continue to be used. Each domain typically has its own authorization infrastructure that is deployed, managed and supported. It will not typically be acceptable to replace any of these technologies in favour of a single model or mechanism.

The Interoperability Challenge
Services that traverse multiple domains and hosting environments need to be able to interact with each other, thus introducing the need for interoperability at multiple levels. At the *protocol level*, it is required mechanisms that allow domains to exchange messages; this can be achieved, for instance, via SOAP/HTTP. At the *policy level*, secure interoperability requires that each party be able to specify any policy it may wish in order to engage in a secure conversation—and that policies expressed by different parties can be made mutually comprehensible. Only then can the parties attempt to establish a secure communication channel and security context upon mutual authentication, trust relationships, and adherence to each other's policy. At the *identity level*, it is required mechanisms for identifying a user from one domain in another domain.

The Trust Relationship Challenge
The VOs that underlie collaborative work within Grids may form quickly, evolve over time and span organisations; as discussed before, their effective operation depends on trust. In the simple case, personal knowledge between parties in the VO allows policies to be derived from identifiable trust "anchors" (parties vouching for other parties). An example in current Grid systems is the use of certificate authorities to root certificate-based identity mechanisms. For these to work, one must "know" about the trustworthiness of the certificate authority used to establish the identity of a party in order to bind it to specific usage policies. However, personal knowledge does not scale for the case on non-trivial VOs, which are most of the VOs, and it is required other technologies such as reputation management [Res00] to create and monitor relationships.

5.2 Security Technologies in the Grid

This section presents the traditional security areas that play an important role in defining security for the Grids and the associated technologies. We build this analysis on top of previous surveys on security for the Grids [Sur02, Bro03b].

Authentication

Authentication deals with verification of the identity of an entity within a network. An entity may be a user, a resource or a service provided as part of the Grid. Authentication is one of the mechanisms helpful in implementing certification trust.

One of the technologies playing a central role in authentication is Public Key Infrastructure (PKI), which defines message formats and protocols that allow entities to securely communicate claims and statements. The most used assertions are those that bind identity and attributes statements to keys. The most popular PKI is defined by the IETF's PKIX working group, which defines a security system used for identifying entities (users and resources) through the use of X.509 identity certificates. In this PKI, highly trusted entities know as certificate authorities (CA) issue X.509 certificates where essentially a unique identity name and the public key of an entity are bound through the digital signature of that CA.

One of the challenges encountered in key management include the need of users of having different credential, since users may play different roles or be part of several projects which have elected to trust different CAs. While PKI could handle this situation by signing the same public key into several different certificates, in practice the user may end up with numerous key pairs to manage. To link these different identities, the notion of federated identities has been developed, as shown in the Liberty Alliance project [Boe03].

Revocation is vital for authentication, for example when a key is compromised or when a user's project ends. PKI relies upon the periodic distribution of Certificate Revocation Lists (CRLs) in order to allow those relying upon certificate to gain confidence in their present validity. The use of CRLs needs careful management, particularly in relation to the frequency of updates.

Authorisation

Authorisation deals with the verification of an action that an entity can perform after authentication was performed successfully. In a grid, resource owners will require the ability to grant or deny access based on identity, membership of groups or virtual organisations, and other dynamic considerations. Thus policies must be established that determine the capabilities of allowed actions. Authorisation is closely related to access control trust. A good description of the current state of authorisation in Grid computing appears in [Cha05].

There are several architectural proposals for handling authorisation in Grids. One of the earliest attempts at providing authorisation in VOs was in the form of the Globus Toolkit Gridmap file. This file simply holds a list of the authenticated distinguished names of the Grid users and the equivalent local user account names that they are to be mapped into. Access control to a resource is then left up to the local operating system and application access control mechanisms. As can be seen, this neither allows the local resource administrator to set a policy for who is allowed to do what, nor does it minimise his/her workload. The Community Authorisation Service (CAS) [Pea02] was the next attempt by the Globus team to improve upon the manageability of user authorisation. CAS allows a resource owner to grant access to a portion of his/her resource to a VO (or community hence the name CAS), and then let the community determine who can use this allocation. The resource owner thus partially delegates the allocation of authorisation rights to the community. This is achieved by having a CAS server, which acts as a trusted intermediary between VO users and resources. Users first contact the CAS asking for permission to use a Grid resource. The CAS consults its policy (which specifies who has permission to do what on which resources) and if granted, returns a digitally self-signed capability to the user optionally containing policy details about what the user is allowed to do. The user then contacts the resource and presents this capability. The resource checks that the capability is signed by a known and trusted CAS and if so maps the CAS's distinguished name into a local user account name via the Gridmap file.

The EU DataGrid and DataTAG projects developed the Virtual Organisation Membership Service (VOMS) [Alf03] as a way of delegating the authorisation of users to managers in the VO. VOMS has gone through a number of iterations in its development. Initially it was a system for dynamically creating Gridmap files from LDAP directories containing details about VO users. Resources could pull a Gridmap file from this periodically. Thus the resource owner never had to actually create or manage the Gridmap file. This system, however, was not scalable. Work within these EU projects then evolved into a push system in which the VOMS server digitally signed a ‘‘pseudo-certificate’’ for the VO user to present to the resource. This pseudo-certificate could contain a local user account name, in which case no Gridmap file would be needed, or it could contain other privileges or group membership details, in which case software would be needed by the resource to interpret this information and grant appropriate rights. The software they developed for this is called the Local Centre Authorisation Service (LCAS) [Ste03]. LCAS makes its authorisation decision based upon the user's certificate and the job specification, which is written in job description language (JDL) format.

Confidentiality

The data being processed in a Grid may be subject to considerable confidentiality constraints, either due to privacy concerns or issues of intellectual property. For instance, grid applications may involve medical data [Bra03], bioinformatics and genomic databases [Cro05] and industrial design information [Wes05].

As mentioned in [Bro03b], confidentiality is usually associated with the encryption of data only, however there are other aspects to be considered for the case of Grids. The use of Grids implies that confidential data is stored in online accessible databases. Access to their interfaces must be carefully controlled, both to allow access only to appropriate users, and also to allow queries and simulations to run over these highly confidential data without that data being compromised or revealed. If the database is to be shared in a Grid, it might need to be operated by a trusted third party. A further novelty of Grid applications is that they may entail running confidential code or using confidential data on a remote resource; running a job on a dynamically-selected cluster according to load may be good resource management, but the data owner may know nothing about the trust status of the cluster selected by the grid software. Confidentiality also extents to the privacy requirements of the actual users and resources. Users are protected under privacy laws and these must be adhered by all components of proposed Grid technology.

5.3 Emerging Trust and Security Technologies

Service-oriented architectures provide the shared organising principles that underpin the collaborative operation of services in open dynamic distributed systems. In this section we review the main Web Services Security standards, proposed by standardisation bodies such as W3C and OASIS. Then, we review OGSA Security Model.

Web Services Security

Web services offer an interoperable framework for stateless, message-based and loosely coupled interaction between software entities. These entities can be spread across different companies and organisations, can be implemented on different platforms, and can reside in different computing infrastructures. Web services expose functionality via XML messages, which are exchanged through the SOAP protocol. The interface of a Web service is described in detail in an XML document using the "Web Service Description Language" (WSDL).

In order to provide security, reliability, transaction abilities and other features, additional specifications exist on top of the XML/SOAP stack. The creation of the specifications is a cross-industry effort, with the participation of standardisation bodies such as W3C and OASIS. A key element in the Web services specifications is the so-called combinability. Web services specifications are being created in such a way that they are mostly independent of each other, however they can be combined to achieve more powerful and complex solutions. In this section we describe some individual specifications, specifically focusing on those dealing with secure and reliable transactions. A complete description of the specifications and its usage is presented in [Geu05].

Reliability

The WS-ReliableMessaging specification describes a protocol for reliable delivery of SOAP messages in the presence of system or network failures. To do so, the initial sender retrieves a unique sequence identifier from the ultimate receiver of the sequence to be sent. Each message in the sequence is uniquely bound to that identifier, together with a sequence number. The receiver of the sequence acknowledges the sender what messages have already been received, thus enabling the sender to determine based on the sequence number which messages have to be retransmitted. WS-ReliableMessaging should be used in conjunction with WS-Security, WS-Secure-Conversation and WS-Trust in order to provide security against attackers at the network layer.

Policies

The Web Services Policy Framework, WS-Policy, provides a general-purpose model to describe web service related policies. WS-Policy by itself only provides a framework to describe logical relationships between policy assertions, without specifying any assertion. WS-PolicyAttachment attaches policies to different subjects. A policy can be attached to an XML element by embedding the policy itself or a link to the policy inside the element or by linking from the policy to the subject that is described by the policy. WS-PolicyAttachment also defines how policies can be referenced from WSDL documents and how policies can be attached to UDDI entities and stored inside a UDDI repository. WS-MetadataExchange defines protocols to retrieve metadata associated with a particular web services endpoint. For example, a WS-Policy document can be retrieved from a SOAP node using WS-Metadata. WS-PolicyAssertions specifies some common WS-Policy assertions, related to text encoding, required SOAP protocol version and so-called ‘MessagePredicate’ assertions that can be used to enforce that a particular header combination exists in a given SOAP message.

Security

WS-SecurityPolicy defines certain security-related assertions that fit into the WS-Policy framework. These assertions are utilised by WS-Security, WS-Trust and WS-SecureConversation. Integrity and confidentiality assertions identify the message parts that have to be protected and it defines what algorithms are permitted. For instance, the ‘SecurityToken’ assertion tells a requestor what security tokens are required to call a given Web service. Visibility assertions identify what particular message parts have to remain unencrypted in order to let SOAP nodes along the message path being able to operate on these parts. The ‘MessageAge’ assertion enables entities to constrain after what time a message is to be treated as expired.

The WS-Security specification defines mechanisms for integrity and confidentiality protection, and data origin authentication for SOAP messages and selected parts thereof. The cryptographic mechanisms are utilized by describing how XML Signature and XML Encryption are applied to parts of a SOAP message. That includes processing rules so that a SOAP node (intermediaries and ultimate receivers) can determine the order in which parts of the message have to be validated or decrypted. These cryptographic properties are described using a specific header field, the <wsse:Security> header. This header provides a mechanism for attaching security-related information to a SOAP message, whereas multiple <wsse:Security> header may exist inside a single message. Each of these headers is intended for consumption by a different SOAP intermediary. This property enables intermediaries to encrypt or decrypt specific parts of a message before forwarding it or enforces that certain parts of the message must be validated before the message is processed further.

Besides the cryptographic processing rules for handling a message, WS-Security defines a generic mechanism for associating security tokens with the message. Tokens generally are either identification or cryptographic material or it may be expressions of capabilities (e.g.

signed authorization statements). WS-Security 1.0 does only define a simple user name token, a container for arbitrary binary tokens (base64 encoded) and a container for XML-formatted tokens. Additional specifications define various ‘token profiles’ that introduce special token formats. For instance, the ‘WS-Security X.509 Certificate Token profile’ defines how X.509 certificates, certificate chains or PKCS#7 certificate revocation lists may be used in conjunction with WS-Security.

The WS-Trust specification introduces the concept of ‘security token services’ (STS). A security token service is a Web service that can issue and validate security tokens. For instance, a Kerberos ticket granting server would be an STS in the non-XML world. A security token service offers functionality to issue new security tokens, to re-new existing tokens that are expiring and to check the validity of existing tokens. Additionally, a security token service can convert one security token into a different security token, thus brokering trust between two trust domains. WS-Trust defines protocols including challenge-and-response protocols to obtain the requested security tokens, thus enabling the mitigation of man-in-the-middle and message replay attacks. The WS-Trust specification also permits that a requestor may need a security token to implement some delegation of rights to a third party. For instance, a requestor could request an authorization token for a colleague that may be valid for a given time interval.

WS-Trust utilises WS-Security for signing and encrypting parts of SOAP messages as well as WS-Policy/SecurityPolicy to express and determine what particular security tokens may be consumed by a given Web service. WS-Trust is a basic building block that can be used to rebuild many of the already existing security protocols and make them fit directly in the web services world by using Web service protocols and data structures.

WS-Federation introduces mechanisms to manage and broker trust relationships in a heterogeneous and federated environment. This includes support for federated identities, attributes and pseudonyms. ‘Federation’ refers to the concept that two or more security domains agree to interact with each other, specifically letting users of the other security domain accessing services in the own security domain. For instance, two companies that have a collaboration agreement may decide that employees from the other company may invoke specific web services. These scenarios with access across security boundaries are called ‘federated environments’ or ‘federations’. Each security domain has its own security token service(s), and each service inside these domains may have individual security policies. WS-Federation uses the WS-Security, WS-SecurityPolicy and WS-Trust specifications to specify scenarios to allow requesters from the one domain to obtain security tokens in the other domain, thus subsequently getting access to the services in the other domain.

Web Services Specification in Implementing the VO Lifecycle

Some of the requirements presented in the analysis of requirement through the VO lifecycle can be met by application of Web services specification, as shown in [Are05].

The identification phase includes defining VO wide policies as well as selecting potential business partners who are both capable of providing the required services and of fulfilling the trustworthiness requirements of the VO. The selection of potential business partners involves looking at repositories, which can realize. The usual Web service technology to be applied is WSDL/UDDI, WSDL describes messages and operations while UDDI offers a discovery mechanism. To include the provision of SLA, “Web Service Level Agreements” (WSLA) has been developed, a XML language for specifying and monitoring SLA for Web Services, which is complementary to WSDL. Determining the required service providers and a proper negotiation requires secure communication. The WS-Security specification and data origin authentication for SOAP messages can be used between the entities to secure the communication.

The realisation of the VO requires the creation of federations, where two or more security domains agree to interact with each other, specifically letting users of the other security domain accessing services in the own security domain. The WS-Federation specification deals with federations by providing mechanism to manage and broker trust relationships in a heterogeneous and federated environment. This includes making use of WS-Trust to support for federated identities, attributes and pseudonyms. The dissemination of configuration information requires secure communication as provided by the WS-Security specification.

Throughout the operation of the VO, service performance will be monitored. This will be used as evidence when constructing the reputation of the service providers. Any violation – e.g. an unauthorised access detected by the access control systems- and security threats –e.g. an event detected by an intrusion detection system- need to be notified to other members in order to take appropriate actions. VO members will also need to enforce security at their local site. For example, providing access to services and adapting to changes and the violations. Monitoring can be supported by event management and notification mechanisms using the WS-Eventing and WS-Notification specifications. This allows the monitoring service partner to receive messages when events occur in other partners. A mechanism for registering interest is needed because the set of Web services interested in receiving such messages is often unknown in advance or will change over time.

OGSA Security

To address the Grid specific security requirements of OGSA, the OGSA Security Group has proposed an architecture leveraging as much as possible from the Web Services Security specifications [Nag03].

As we mentioned previously, secure operation in a Grid environment requires that applications and services be able to support a variety of security functionalities, such as authentication, authorization, credential conversion, auditing and delegation. These functionalities are based on mechanisms that may evolve over time as new devices are developed or policies change. As suggested in [Sie03], Grid applications must avoid embedding security mechanisms statically.

Exposing security functionalities as services (i.e., with a WSDL definition) achieves a level of abstraction that helps provide an integrated, secure Grid environment. An OGSA infrastructure may use a set of primitive security functions in the form of services themselves. [Nag03] suggest the following security services:

- An authentication service: An authentication service is concerned with verifying proof of an asserted identity. One example is the evaluation of a User ID and password combination, in which a service requestor supplies the appropriate password for an asserted user ID. Another example involves a service requestor authenticating through a Kerberos mechanism, and a ticket being passed to the service provider’s hosting environment, which determines the authenticity of the ticket before the service is instantiated.
- Identity mapping service: The identity mapping service provides the capability of transforming an identity that exists in one identity domain into an identity within another identity domain. The identity mapping service is not concerned with the authentication of the service requestor; rather it is strictly a policy driven name mapping service
- Authorization service: The authorization service is concerned with resolving a policy based access control decision. The authorization service consumes as input a credential that embodies the identity of an authenticated service requestor and for the resource that the service requestor

requests, resolves based on policy, whether or not the service requestor is authorized to access the resource. It is expected that the hosting environment for OGSA compliant services will provide access control functions, and it is appropriate to further expose an abstract authorization service depending on the granularity of the access control policy that is being enforced.

- VO Policy service: The VO policy service is concerned with the management of policies. The aggregation of the policies contained within and managed by the policy service comprises a VO's policy set. The policy service may be thought of as another primitive service, which is used by the authorization, audit, identity mapping and other services as needed.
- Credential Conversion service: The credential conversion service provides credential conversion between one type of credential to another type or form of credential. This may include such tasks as reconciling group membership, privileges, attributes and assertions associated with entities (service requestors and service providers). For example, the credential conversion service may convert a Kerberos credential to a form that is required by the authorization service. The policy driven credential conversion service facilitates the interoperability of differing credential types, which may be consumed by services. It is expected that the credential conversion service would use the identity mapping service. WS-Trust defines such a service.
- Audit Service: The audit service similarly to the identity mapping and authorization services is policy driven. The audit service is responsible for producing records, which track security relevant events. The resulting audit records may be reduced and examined as to determine if the desired security policy is being enforced. Auditing and subsequently reduction tooling are used by the security administrators within a VO to determine the VO's adherence to the stated access control and authentication policies.
- Profile Service: The profile service is concerned with managing service requestor's preferences and data which may not be directly consumed by the authorization service. This may be service requestor specific personalization data, which for example can be used to tailor or customize the service requestor's experience (if incorporated into an application which interfaces with end-users.) It is expected that primarily this data will be used by applications that interface with a person.
- Privacy Service: The privacy service is primarily concerned with the policy driven classification of personally identifiable information (PII). Service providers and service requestors may store personally identifiable information using the Privacy Service. Such a service can be used to articulate and enforce a VO's privacy policy.

Grid Security Infrastructure

The Grid Security Infrastructure (GSI) is a specific implementation of an OGSA-based Grid security architecture that include as part of the Globus Toolkit Version 3 (GT3) [Wel03]. Given the prominent use of Globus within the Grid community, let us briefly revise such implementation.

- Authentication. GSI defines a credential format based on X.509 identity certification. An X.509 certificate, in conjunction with an associated private key, forms a unique credential set that a Grid entity (requestor or service provider) uses to authenticate itself to other Grid entities (e.g., through a challenge-response protocol such as TLS).

- Identity Federation. GSI uses gateways to translate between X.509-based identity credential and other mechanisms. For example, the Kerberos Certificate Authority (CKA) and SSLK5/PKNIT provide translation from Kerberos to GSI and vice versa, respectively. These mechanisms allow a site with an existing Kerberos infrastructure to convert credentials between Kerberos and GSI as needed.

- Dynamic Entities and Delegation. GSI introduces X.509 proxy certificates, an extension to X.509 identity certificates that allows a user to assign dynamically a new X.509 identity to an entity and then delegate some subset of their rights to that identity.

- Message Level Security. Globus Toolkit Version 3 (GT3) uses the Web Services Security specifications to allow security messages and secured messages to be transported, understood and manipulated by standard Web services tools and software.

In relation to stateful and secured communication, GSI supports the establishment of a security context that authenticates two parties to each other and allows for the exchange of secured messages between the two parties. GT3 achieves security context establishment by implementing preliminary versions of WS-SecurityConversation and WS-Trust specifications. Once the security context is established, GIS implements message protection using the Web Services standards for secured messages XML-Signature and XML-Encription.

To allow for communication without the initial establishment of a security context, GT3 offers the ability to sign messages independent of any established security context, by using XML-Signature specification.

- Trust Domains. The requirement for overlaid trust domains to establish VOs is satisfied by using both proxy certificates and security services such as CAS. GSI has an implicit policy that any two entities bearing proxy certificates issued by the same user will inherently trust each other. This policy allows users to create trust domains dynamically by issuing proxy certificates to any services that they want to interoperate.

6 Trust and Security in Other Grid Projects

This section reviews EU projects working on trust and security in Grids.

6.1 EU Grid Concertation Technical Group TG6 – Trust and Security

TG6 [Sur05] is one of eight groups formed to work on technical concertation between EC FP5 and FP6 Grid projects, focusing on the area of trust and security. It comprises several FP6 projects tackling trust and security in Grids. TG6 has identified some topics related to either a gap in the technology or a gap in know-how that could be addressed through inter-project exchange. The topics are the following.

- Review of Web Service Security specifications: what are they, how do they relate to each other, how do they address user requirements, and where are the gaps that might need further development to fill? Most projects are planning to use web service technologies, but few had a clear picture of how these can meet which of their requirements.
- Virtual Organisation models: what business models are appropriate within VO, how do these relate to trust, and what requirements do they place on lower-level security technologies? It is clear that several projects are developing concepts related to different kinds of VO, and can benefit from exchanging ideas, requirements and potentially technologies for VO classification and VO operation.
- Privacy issues: what does privacy mean, and how does it impact Grid construction and Grid operation? Several projects are concerned with applications where privacy must be maintained. There is an opportunity to transfer results and know-how from this FP5 project into several ongoing FP6 activities that have privacy issues to address.
- Mobile network security: how does network-level security impact the design of Grids, and specifically how can Grids operate over mobile links without compromising its own security models?
- Operational best practice in e-Science: there is considerable experience in Grid operations from e-Science activities such as EDG and EGEE. The goal is to capture best practice in operational security from these projects and identify where it is applicable in industrial FP6 projects.

Below we present a summary of the projects participating in TG6 and the main trust and security challenges that are being considered.

- **Akogrimo**[3]. Akogrimo will bring together the Grid world with the mobile Internet. Within this context it should be mentioned that a lot of currently deployed security mechanisms provided by the network have not been developed for the mobile Internet where e.g. a user might change the Internet Protocol address e.g. once each 10 seconds. In the current Grid world a lot of security mechanisms have been deployed and are under development which do no directly communicate and interact with security mechanisms from the lower layer.

  Within Akogrimo a cross layer security framework will be developed providing the security support for users connected to a "commercial" mobile Internet and accessing commercial Grid services in a dynamic way. The potential contribution of Akogrimo to the related Grid projects in the community are first the provision of new requirements coming from a commercial mobile Internet which immediately come to the concept of Mobile Virtual Dynamic Organizations (MVDOs) and the distribution of overall security features across the overall protocol stack.

- **Daidalos**[4]. Daidalos is an IP focussing on network infrastructure but also with service aspects. It is driven by operators and already incorporates rather new and emerging concepts like mobility and context-awareness. Security and privacy are inherent parts from the beginning on.

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

[4] http://www.ist-daidalos.org/

Grid systems have to rely heavily on communication. Moreover, they are in need of a huge infrastructure being potentially provided by operators that need to earn money with it. Therefore, a close interaction of Grid systems with the network is necessary. Daidalos can raise awareness of network and operator aspects to FP6 Grid projects, doing both restricting and supporting Grid systems.

- **EGEE**[5]. The EGEE security activities comprise three independent but interrelated topics: global trust establishment for authentication, operational security responsibilities and incident procedures, and increasing the robustness and deployability of grid middleware security mechanisms.

  Global trust building is accomplished through the European Grid authentication policy management authority for e-Science (EUGridPMA for short). This body defines common guidelines for authenticating entities in the Grid, and accredits authentication authorities according to those guidelines. EGEE has established a Joint (Operational) Security Group to consider other operational aspects such as authorisation responsibilities, common Acceptable Usage Policies (AUPs), and distributed security incident response. Finally, EGEE is also re-engineering its current middleware to use a service oriented architecture (SOA) built using Web Services. This includes a new Authorisation model in which delegation is tokenised and no longer depends on user identity authentication.

- **HPC4U**[6]. The objective of the HPC4U project is to expand the potential of the Grid approach to Complex Problems Solving through the development of software components for a dependable and reliable Grid environments and combining this with Service Level Agreements (SLA) and commodity-based clusters providing Quality of Service (QoS). Development of HPC4U will take place in a Grid context following standards of the Global Grid Forum (GGF).

  HPC4U will not focus on developing security mechanisms, but leverages trust and security work of other projects to achieve reliability, predictability and dependability.

- **NextGRID**[7]. The goal of NextGRID is to develop architectural models and components that will lead to the emergence of the Next Generation Grid that is economically viable, and useful to business and society. To achieve its goals NextGRID has integrating activities covering Grid architecture, business and operational issues, applications and standards, and development activities covering Grid foundations and core services, Grid dynamics and federation models, and Grid user interaction models.

  Security and Trust are key issues in NextGRID, without which it cannot meet the needs of business or society. Privacy is also important to enable participation by the public. To address these issues, security will be built into the NextGRID architecture at all levels, and will be a focus for the architecture design activity from the beginning of the project. This will cover secure communication, authentication, authorization, roles, firewall management, and security policy enforcement. NextGRID addresses these aspects at the level of services (through its Foundations work) and in service federations (through its Dynamics work). The interaction between security and management (expressed through VO models), including decentralised and P2P management mechanisms, and VO lifecycles, is of considerable interest in dynamic federation scenarios.

  NextGRID is also concerned with operational security requirements from business, including mechanisms and policies internal to a site for protecting resources and recovery strategies following a breach. This work will focus on extending risk management methods to uses of the Grid, and using this to generate operational policies that are relevant to business and societal (as distinct from research) scenarios.

- **EU-Provenance**[8]. The overarching aim of the Provenance project is to design, conceive and develop an industrial-strength, open provenance architecture for grid systems, and to deploy and evaluate it in complex grid applications, namely aerospace engineering and organ transplant management. This support includes a scalable and secure architecture, an open proposal for standardising the protocols and data structures, a set of tools for configuring and using the provenance architecture, an open source reference implementation, and a deployment and validation in industrial context.

  Expected contribution to TG6 from EU-Provenance:

[5] http://www.eu-egee.org/
[6] http://www.hpc4u.org/
[7] http://www.nextgrid.org/
[8] http://www.gridprovenance.org/

  - Security architecture generally requires an audit of actions performed by authorized individuals in the system. This audit trail can be subsequently processed and analyzed towards various ends (e.g. dynamically fine-tuning security policies). It is possible to formulate the gathering and analysis of audit information as a provenance activity. As such, the Provenance project could analyze briefly the audit storage and processing requirements of other relevant projects, and illustrate a possible way these requirements can be mapped to the provenance architecture using the various interfaces and tools provided as part of this project. This will take the form of a documented case study and will not involve the development of additional software or interfaces outside the purview of EU Provenance.
  - Experience in using open source software for implementing federation management and open source software with standards like SAML and XACML.

- **SIMDAT**[9]. The goals of SIMDAT are to test and enhance data grid technology to enable and support product and process design and service provision across four important industrial sectors: automotive, aerospace, pharmaceuticals and meteorology. The main outputs will be a set of generic application enabling tools produced through transfer of technology between sectors, and from underlying Grid developments, applied to enable Grid applications in the target sectors.

  Trust and security are fundamental to SIMDAT, as they provide the basis for federating resources (including data and knowledge) between collaborating organisations in these highly competitive industrial sectors. The bulk of the work on Trust will focus on how to represent and manage Trust in the context of VOs. It is expected that this will be stimulated by the aero application sector, where collaboration is well established even from the early design stages for a new product. Security technology will be developed mainly at the Grid infrastructure and Data access and integration levels, and in the work needed to support analysis services based on commercial application software.

- **TrustCoM**[10]. TrustCoM is developing an integrated framework for trust, security and contract management for collaborative business processing in dynamically-evolving Virtual Organisations (VOs). A realisation of the TrustCoM framework will be delivered by means of open-standards web services based specifications and a reference implementation. Validation will take place within testbeds in the areas of collaborative engineering (CE) and provision of ad-hoc, dynamic processes for aggregated electronic services (AS).

  TrustCoM addresses trust and security issues across the complete VO life-cycle, including discovery and justified identification of credible, trusted partners (VO Identification), establishment of trust between VO members (VO Formation), maintenance of trust, autonomic security management, adaptive deployment of security policies (VO Operation and Evolution), and termination of trust relationships and maintenance of trust knowledge (VO Dissolution).

- **UniGridS**[11]. Security, the protection of sites and users from malicious users, and delegation, users authorising servers to perform actions on their behalf, are of fundamental importance to Grid Computing. An effective Grid infrastructure will strike the appropriate balance between good security and flexible delegation.

  The UNICORE approach to security and delegation is known to be strong, but this strength creates a tension with the flexible deployment of OGSA based Web Services. For example, the Generic Service Portal will create a job description for a user but, under the current model, is unable to obtain the explicit authorisation of the work that is required by the UNICORE servers. UniGridS will extend the UNICORE security architecture to support explicit statements of trust, to give the level of flexibility needed to support dynamic delegation, but without undermining the basic UNICORE security architecture. This increased flexibility will also facilitate the incorporation of emerging standards in Web Service and Grid security.

[9] http://www.simdat.org
[10] http://www.eu-trustcom.com
[11] http://www.unigrids.org

7 Conclusions

This document presents a survey on trust and security in Grid systems.

Trust and security have proved over the years to be extremely difficult to achieve; this is palpable by the millions of pounds that disappear every year through Internet fraud. The problem remains a challenge for Grids, given their scale and complexity. Trust and security are socio-technical topics, and any solution should take into account this multidisciplinary dimension. Where technical solutions exists, such as the case of PKI technology, issues such as how to enable users to manage keys effectively remains unclear.

For the XtreemOS project there is a requirement to develop the trust and security mechanisms that are needed to operate VOs. The description of VOs in this review shows that they vary in size, topology and the security requirements between members. XtreemOS goes further in that it requires that the Linux kernel be modified to accommodate the Grid middleware which would usually operate these security mechanisms.

References

[Abr95] M.D. Abrams, M.V. Joyce. Trusted Computing Update. Computers and Security, 14(1): 57-68. 1995.

[Alf03] R. Alfieri et al. VOMS: An Authorization System for Virtual Organizations. In Proceedings of 1st European Across Grids Conference, Santiago de Compostela, 2003. Available from: http://grid-auth.infn.it/docs/VOMS-Santiago.pdf.

[Are05] A.E.Arenas, I. Djordjevic, T. Dimitrakos, L. Titkov, J. Claessens, C. Geuer-Pollman, E.C. Lupu, N. Tuptuk, S. Wesner, L. Schubert. Towards Web Services Profiles for Trust and Security in Virtual Organisations**.** IFIP Working Conference on Virtual Enterprises – PRO-VE'05, Valencia, Spain. 2005.

[Boe03] S. Boeyen et al. Liberty Trust Models Guidelines. In J. Linn (editor), Liberty Alliance Project. Liberty Alliance, draft version 1.0, 2003.

[Bra03] M. Brady, D. Gavaghan et al. eDiamond: A Grid-Enabled Federated Database for Annotated Mammograms. In F. Berman, G. Fox, T. Hey (editors), Grid Computing: Making the Global Infrastructure a Reality, Wiley, 2003

[Bra03a] J. Bradshaw, A. Uszok, R. Jeffers, et al. Representation and reasoning about DAML-based policy and domain services in KAoS. In Proc. of The 2nd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems (AAMAS2003). 2003.

[Bro03a] P.J. Broadfoot, G. Lowe. Architectures for Secure Delegation within Grids. Oxford University Computing Laboratory Technical Report, PRG-RR-03-19, 2003.

[Bro03b] P.J. Broadfoot, A.P. Martin. A Critical Survey of Grid Security Requirements and Technologies. Oxford University Computing Laboratory Technical Report, PRG-RR-03-15, 2003.
[Bur99] JM Burn, P Marshall, M Wild. Managing Changes in the Virtual Organisation. Proceedings of the Seventh European Conference on Information Systems 40-54, Copenhagen Business School, Copenhagen, 1999.

[Cam03] L.M. Camarinha-Matos, H. Afsarmanesh. A Roadmap for Strategic Research on Virtual Organizations. Proceedings of IFIP Working Conference on Virtual Enterprises - PRO-VE'03, Lugano, Switzerland, pages 33-46, 2003.

[Cas98] C. Castelfranchi, R. Falcone. Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification. In Y. Demazeau (editor), Proceedings of the Third International Conference on Multi-Agent Systems. IEEE C.S., Los Alamitos, 1998.

[Cas00] C. Castelfranchi, R. Falcone, B. Sadighi, Y-H Tain. Guest Editorial. Applied Artificial Intelligence, 14(9), Taylor & Frances, 2000.

[Cha05] D. Chadwick. Authorisation in Grid Computing. . Information Security Technical Report, Elsevier, 10(1)33:40, 2005.

[Cro05] S. Crompton, B. Matthews, A. Gray, A. Jones, R. White. Data Integration in Bioinformatics using OGSA-DAI. In Proceedings of Fourth All Hands Meeting, AHM2005, UK, 2005.

[Den03] G. Denker, L. Kagal, T. Finin, M. Paolucci and K. Sycara. Security for DAML Web Services: Annotation and Matchmaking. In D. Fensel, K. Sycara, & J.Mylopoulos (Ed.), The Semantic Web—ISWC 2003. Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA, October 2003, LNCS 2870, 2003.

[Dim01] T. Dimitrakos. System Models, e-Risk and e-Trust. Towards Bridging the Gap? in *Towards the E-Society: E-Business, E-Commerce, and E-Government*, eds. B. Schmid, K. Stanoevska-Slabeva, V. Tschammer. Kluwer Academic Publishers, 2001.

[Dim04] T. Dimitrakos, D. Golby P. Kearney. Towards a Trust and Contract Management Framework for Dynamic Virtual Organisations. In *eAdoption and the Knowledge Economy: eChallenges 2004*. Vienna, Austria, 2004.
[EuroPKI004] EuroPKI Certificate Policy, *VERSION 1.1,* January 2004,
http://www.europki.org/ca/root/cps/en_cp.pdf

[Fos98] I. Foster, C. Kesselman, G. Tsudki, S. Tuecke. A Security Architecture for Computational Grids. In Proceedings of 5th ACM Conference on Computer and Communication Security, 1998.

[Fos01] I. Foster, C Kesselman, S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputing Applications 15(3), 200-222, 2001.

[Fos03] I. Foster, C. Kesselman. The Grid: Blue Print for a New Computing Infrastructure. Morgan Kauffmann, 2003.

[Gam00a] D. Gambetta (editor). Trust: Making and Breaking Cooperative Relations. Department of Sociology, University of Oxford, 2000. Available at http://www.sociology.ox.ac.uk/papers/trustbook.html .

[Gam00b] D. Gambetta. Can We Trust Trust? In [Gam00a], chapter 13, 2000.

[Gas90] M. Gasser, E. McDermott. An Architecture for Practical Delegation in a Distributed System. IEEE Symposium on Research in Security and Privacy, 1990.

[Gra00] T. Grandison, M. Sloman. A Survey of Trust in Internet Applications. IEEE Communications Survey and Tutorials, 3, 2000.

[Gra03] T. Grandison, M. Sloman. Trust Management Tools for Internet Applications. In [Nix03], 2003.

[Gue05] C. Geuer-Pollmann, J. Claessens. Web Services and Web Service Security Standards. Information Security Technical Report, Elsevier, 10(1)15:24, 2005.

[Her05] P. Hermann, V. Issarny, S. Shue (editors). Third International Conference on Trust Management. Lecture Notes in Computer Science, vol. 3477, Springer, 2005.

[Jen04]. C.D. Jensen, S. Poslad, T. Dimitrakos (editors). Second International Conference on Trust Management. Lecture Notes in Computer Science, vol. 2995, Springer, 2004.

[Joh03] W.E. Johnston, J.M. Brooke, R. Butler, D. Foster and M. Mazzucato. Production Deployment: Experiences and Recommendations. In [Fos03], 2003.

[Joh03a] M. Johnson, P. Chang, R. Jeffers et al. KAoS semantic policy and domain services: An application of DAML to Web services-based grid architectures. Proceedings of the AAMAS 03 Workshop on Web Services and Agent-Based Engineering. Melbourne, Australia, 2003.

[Jon99] S. Jones. TRUST-EC: Requirements for Trust and Confidence in E-Commerce. European Commission Joint Research Centre, 1999.

[Jos99] A. Josang. An Algebra for Assessing Trust in Certification Chains. In J. Kochmar (editor), Proceedings of the Network and Distributed Systems Security Symposium (NDSS'99). The Internet Society, 1999.

[Jos05] A. Josang, R. Ismail, C. Boyd. A Survey of Trust and Reputation Systems for Online Service Provision. To appear in Decision Support Systems, 2005.

[Kag03] L. Kagal, T. Finin, J. Anupam. A Logical Policy Language for a Pervasive Computing Environment., 4th IEEE Int. Workshop on Policies for Distributed Systems and Networks, Lake Como, 4-6 June, 2003.
[KatXX] BR Katzy, C Zhang, H loeh. Reference Models for Virtual Organizations. Working Paper No 2704, Working Paper Series, CeTIM.

[Kin98] A. Kini, J. Choobineh. Trust in Electronic Commerce: Definition and Theoretical Consideration. Proceedings of 31st International Conference on System Sciences, IEEE, 1998.

[McK96] D.H. McKnight, N.L. Chervany. The Meaning of Trust. Technical Report MISRC Working Paper Series 96-04, University of Minnesota. Management Information Systems Research Center, 1996.

[Nag03] N. Nagaratnam, P. Janson, J. Dayka, A. Nadalin, F. Siebenlist, V. Welch, S. Tuecke, I. Foster. Security Architecture for Open Grid Services. Available at http://forge.gridforum.org/projects/ogsa-sec-wg/.

[Nix03] P. Nixon, S. Terzis (editors). First International Conference on Trust Management. Lecture Notes in Computer Science, vol. 2692, Springer, 2003.

[Pea02] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke. A Community Authorization Service for Group Collaboration. In Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks, 2002.

[Ras96] L. Rasmusson, S. Janssen. Simulated Social Control for Secure Internet Commerce. In C. Meadows, editor, Proceedings of the 1996 New Security Paradigms Workshop. ACM, 1996.

[Res00] P. Resnick, P. Zeckhauser, R. Friedman, K. Kuwabara. Reputation Systems. Communications of the ACM, 43(12):45-48, December 2000.

[Riorden96] MH Riordan, Contracting with Qualified Suppliers, International Economic Review, 1996 - Vol. 37, No. 1 (Feb., 1996), pp. 115-128, doi:10.2307/2527249

[Sie03] F. Siebenlist, N. Nagaratnam, V. Welch, C. Neuman. Security for Virtual Organizations: Federating Trust and Policy Domains. In [Fos03].

[Ste03] M. Steenbakkers. Guide to LCAS v.1.1.16, September 2003. Available at http://www.dutchgrid.nl/DataGrid/wp4/lcas/edg-lcas-1.1.

[Sur02] M. Surridge. A Rough Guide to Grid Security. Technical Report, IT Innovation Centre, V1.1a, 2002.

[Sur05] M. Surridge, J. Claessens. TG6 Trust and Security - White Paper on State of the Art and Planned Developments in the Context of FP6 Grid Projects.

[Ton03] G. Tonti, J. Bradshaw et al. (2003). Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In D. Fensel, K. Sycara, & J.Mylopoulos (Eds.), The Semantic Web—ISWC 2003. Proc. of the 2nd Int. Semantic Web Conf., Sanibel Island, Florida, USA, October 2003, LNCS 2870

[Wai02] M. Waidner (editor). Ercim News, Special Theme: Information Security. No 49, 2002.

[Wel03] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski et al. Security for Grid Services. In Proceedings of 12th IEEE International Symposium on High Performace Distributed Computing. IEEE Computer Society Press, 2003.

[Wes05] S. Wesner, L. Schubert, T. Dimitrakos. Dynamic Virtual Organizations in Engineering. In Proceedings of German-Russian Workshop, 2005.

[Zim95] P.R. Zimmermann. The Official PGP User's Guide. MIT Press, 1995.