EUROPEAN MIDDLEWARE INITIATIVE

EMI Security Architecture

http://openaire.cern.ch/record/5959
10.5281/ZENODO.5959

April 2013

EMI is partially funded by the European Commission under Grant Agreement RI-261611

TABLE OF CONTENTS

I. DOCUMENT AMENDMENT PROCEDURE

This document can be amended by the authors further to any feedback from other teams or people. Minor changes, such as spelling corrections, content formatting or minor text re-organization not affecting the content and meaning of the document can be applied by the authors without peer review. Other changes must be submitted for peer review and to the EMI PEB for approval.

When the document is modified for any reason, its version number shall be incremented accordingly. The document version number shall follow the standard EMI conventions for document versioning. The document shall be maintained in the CERN CDS repository and be made accessible through the OpenAIRE portal.

II. GLOSSARY

Acronym	Long Name
AA	Attribute Authority. See http://www.ietf.org/rfc/rfc3820.txt
AAI	Authentication and Authorization Infrastructure
AC	Attribute Certificate: A special case of an X.509 certificate. See http://www.ietf.org/rfc/rfc3281.txt for details.
ACL	Access Control List: List of rules regulating access to a resource or entity
AP	Authentication Profile
ARC	Advanced Resource Connector
BDII	Berkeley Database Information Index: An LDAP-based information system, where information about the current state of the Grid is stored.
CA	Certificate Authority: An internal entity or trusted third party that issues, signs, revokes and manages digital certificates.
Certificate	Information issued by a trusted party. Used to identify an individual or system.
CE	Computing Element: a Grid-enabled computing resource
Credentials	Evidence asserting the user's right to access certain systems (e.g. username, password, etc)
CRL	Certificate Revocation List
CSR	Certificate Signing Request
CREAM	Computing Resource Execution and Management: The gLite CE framework developed by INFN.
DN	Distinguished Name to uniquely denote a user or entity. The DN is typically used in directories like X.500 and LDAP, but also within X.509 certificates.
DPM	Disk Pool Manager
EGEE	Enabling Grids for E-sciencE: EU funded Grid project
End Entity	System (individual, host, service) that receives a certificate expressing its identity
EES	Execution Environment Service: Service internal to the new authorization system, which is being implemented within EGEE-III.
EUGRIDP	International organization to coordinate the trust fabric for e-Science Grid

MA	authentication in Europe. See http://www.eugridpma.org for details.
Federated Identity	Management and use of identity information across security domains, e.g. between members of a federation. Federated identity inevitably deals with issues like liability, security, privacy and trust.
FQAN	Fully Qualified Attribute Name: A string containing VOMS group and (optionally) role information.
FTS	File Transfer Service: A service, which allows to transfer a collection of files between a source and a destination.
GA	Generic Attribute: VOMS attributes in the format of key-value pairs.
GACL	Grid Access Control List: a Grid specific ACL
GID	Group IDentifier in the Unix operating system
gLite	Middleware stack developed by the EGEE project
Globus	A Grid middleware stack. See http://www.globus.org for details.
GLUE	Grid Laboratory Uniform Environment: An abstract information model for the Grid. See http://forge.ogf.org/sf/projects/glue-wg for details.
GSI	Grid Security Infrastructure: the PKI infrastructure that has been customized and extended for the Grid by the Globus project.
IGTF	International Grid Trust Federation: Body with the goal to harmonize and synchronize PMAs policies to establish and maintain global trust relationships in e-Science. See http://www.igtf.org for details.
Internet2	A consortium, led by the research and education community, which promotes advanced networking in the USA.
ITU	International Telecommunication Union: International organization established to standardize and regulate international radio and telecommunication.
ITU-T	International Telecommunication Standardization Sector
LCAS	Local Centre Authorization Service: An authorization framework, which authorizes Grid users based on their X.509 credential.
LCMAPS	Local Centre MAPing Service: A framework, which maps Grid credentials to local accounts on a host attached to the Grid.
LFC	LHC File Catalogue: A file catalogue developed by the LCG project.
LRMS	Local Resource Management System: the batch system behind the Grid CE framework.
MICS	Member Integrated Credential Service: An IGTF profile for issuing (long-lived) X.509 certificates to End Entities based on an identity management system operated by an institution.
MWSG	Middleware Security Group: An informal working group for security architects and knowledgeable security individuals from EGEE, OSG, OMII-Europe and other Grid projects.
MyProxy	A Grid service used for storing and issuing certificates as well as renewing expiring certificates. It was developed by the Globus project. [R17]
NREN	National Research and Education Network
OSCP	Online Certificate Status Protocol

OSCT	Operational Security Coordination Team: Team that is responsible for ensuring the overall security coordination in EGEE.
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PKI	Public Key Infrastructure: Processes and technologies used to issue and manage digital certificates, enabling third parties to authenticate individual users, services and hosts.
PMA	Policy Management Authority: Body responsible for defining minimum standards for the CP/CPSs of a PKI infrastructures and accrediting against those standards
Principle	Participant in a security operation. It can be a service or a user.
Pseudonymity	Describes a state of disguised identity resulting from the consistent use of another, unknown name.
Resource Provider	Synonym for site.
SAML	Security Assertion Markup Language: XML standard for exchanging authentication and authorization data.
SCAS	Site Central Authorization Service
SE	Storage Element: a Grid enabled storage resource.
Shibboleth	Federated identity management solution from Internet2/MACE (Middleware Architecture Committee for Education). It is the name of the architecture as well as the name of the open source implementation.
Short-lived X.509 certificate	An X.509 certificate with a life time of less than 1 million seconds (approx. 11 days)
Site	Organization having administrative control of resources provided to the Grid. This may be at one physical location of spread across multiple locations.
SOAP	Simple Object Access Protocol
SRM	Storage Resource Management
StoRM	Storage Resource Manager
ssss	Shamir's secret sharing scheme: A scheme for distributing a secret among a group of participants. http://point-at-infinity.org/ssss/
STS	Security Token Service: A Web service that authenticates clients by validating credentials that are presented by a client. It can issue to a client a security token for a successfully authenticated client.
Trusted Third Party (TTP)	Entity, which facilitates interactions between two parties who both trust the third party. They use this trust to secure their own interactions. A CA is an example of a TTP.
UI	User Interface: host from where the user interacts with the Grid software in the gLite middleware environment.
UID	User IDentifier in the Unix operating system

VO	Virtual Organization
VOMS	VO Management Service (may sometimes also refer to the server hosting the service.)
WMS	Workload Management System: a central broker for job submission and management
WN	Worker Node: the entity where jobs get executed
X.509	A standard for public key infrastructures. It defines among other things standard formats for certificates. See http://www.ietf.org/rfc/rfc2459.txt for details.
XACML	eXtensible Access Control Markup Language - a declarative access control policy language implemented in XML

The complete EMI glossary is available at https://twiki.cern.ch/twiki/bin/view/EMI/EmiGlossary .

III. COPYRIGHT NOTICE

1. INTRODUCTION

This document describes the various architectures of the three middlewares that comprise the EMI software stack. It also outlines the common efforts in the security area that allow interoperability between these middlewares. The assessment of the EMI Security presented in this document was performed internally by members of the Security Area of the EMI project. This assessment was performed on a one-time basis specifically for this document and is given in the Section 10 "Assessment, Strengths, ideas for improvement".

2. OVERVIEW OF THE EMI SECURITY ARCHITECTURE

This section describes, at a high level, the components that provide the security for the EMI middleware stack. The current overall EMI middleware stack is the result of a combination of developments of previous Grid middleware projects: ARC, gLite and UNICORE and independent software products (dCache and StoRM). The underlying security models of the three software stacks remain unchanged and differ between each stack. The gLite security architecture is described in [R33], UNICORE in [R34] and ARC security documentation in [R35].

As such, there is no "EMI Security Architecture" that supersedes the security models of each stack. The EMI project has worked on extending and modifying some existing components and producing new components in order to facilitate the interoperability and improve the uniformity between the stacks. Profiles for Authentication and Authorization have been produced. These modifications and additions have been made with input from general security concepts of the software development on one hand and the actual experience of deploying and securely operating the EGI and WLCG infrastructures on the other hand.

It should also be kept in mind, that the security infrastructure must not only satisfy the security requirements, but it must also be practical and easy to use by site administrators, security personnel and users. During this period of development, the existing Grid infrastructure could not be interrupted and therefore the changes made are incremental and with the agreement of the operators.

In this chapter the main building blocks of the EMI security are presented. It starts with the definition of the term "security architecture" and defines the usage of the terms "trust", "authentication" and "authorization". It then continues with a description of the virtual organization and its relation to sites before enumerating the main security building blocks. In doing so, the document tries to strike a balance between presenting high-level security concepts and its actual implementation in EMI.

2.1. DEFINITION OF A SECURITY ARCHITECTURE

Within the context of this document, the term "security architecture"[1] is defined as "a set of features and services that tackles a set of security requirements and can handle a set of use cases".

The term service is used here in a broader sense than (web) service, which is the common jargon in other documents, and also includes infrastructure and user-driven services such as certification, auditing and incident response procedures.

According to [R2], the security architecture is composed of building blocks, called facilities. Each facility can be viewed as an atomic piece of functionality that facilitates some security requirement. The facilities should be factored orthogonally against each other, such that their use is optional, and keeping the number of interdependencies to a minimum, allowing for individual advancements of respective technologies and mechanisms used in the implementation of individual architectures.

The facilities, along with a short description, are listed in Table 1. The definitions were taken from [R2].

Functionality	Facility	Short Description
Logging and Auditing	Logging	A common infrastructure for the recording of system events for tracking, accountability, and auditing purposes.
	Auditing	The auditing system uses information recorded about system and user activity for the purposes

[1] As defined in [R1].

		of accountability and security assurance.
Authentication	Identity credentials and trust infrastructure	The identification of users, agents, hosts, and services, when they interact with each other.
	Short-lived credential services	Short-lived credential services (SLCS) leverage the organizational or federated authentication infrastructure and issue (short-lived) Grid credentials to their users.
	Enforcing validity constraints	Additional validation tests are required to assess that credentials (in particular proxy certificates) are adhering to established operational policies.
	Revocation	The process of invalidating a credential and the secure distributed propagation of such status change information.
	Certificate renewal	The automated, yet controlled and managed, renewal of short-lived credentials and authorization assertions.
	Anonymity, privacy, pseudonymity	The controlled protection of user identities and data, and escrow of the same for management and authorities.
	Bootstrapping authentication	A collective term for the initial security mechanisms that can be used to acquire a Grid credential.
	Credential store	The controlled secure management of user credentials, to permit the enforcement of key hygiene.
Key Management	Key hygiene	Enforcement of proper handling practices of user held credentials.
	Encryption key management	The secure management of cryptographic keys that in turn protect data stored in encrypted form.
Authorization	Authorization services	A collective term for the (centralized) services used to manage access control (typically one per VO)
	Mutual authorization	The process in which both parties authorize each other before engaging in a message exchange, typically to avoid information leakage.
	Authorization framework	A framework that collect and combines policies and information from several sources using pluggable extensions.
	Authorization interfaces to existing systems	An authorization interface to existing and legacy systems, which allows a combined and flexible decision-making process by taking into account information, assertions and policies from a variety of authorities.
Delegation	Delegation	The capability of transferring rights and privileges to allow for a principal (e.g. an application or a user) to act on your behalf.
Sandboxing	Securing the hosted to native interface	Isolating the (user-provided) applications from each other and from the local system as much as possible, while preserving the appearance of transparent access to shared remote resources.

	Network isolation	Dynamically adapting firewall policy to enforce strict rules yet being able to obey the connectivity needs of (some) users and applications.

Table 1: EMI Security Facilities

2.2. TRUST, AUTHENTICATION AND AUTHORIZATION: A TERMINOLOGY

The concepts of trust, authentication and authorization are often poorly defined or confused. This section provides a short description of these terms with a focus on X.509 credentials.

- Under *Trust* - the process of ensuring that the issuer of a credential, and the credential itself, is trustworthy.
- Under *Authentication* - the process of ensuring a credential is valid and belongs to the individual that presents it.
- Under *Authorization* - the process of checking that a person has the rights to perform an operation. Authorization can be issued based on several criteria, such as, for example, the identity of the person or attributes provided about the person by a trusted third party.

Note that trust and authentication are often meshed together, but as a matter of fact they should be treated separately. For example, one can establish trust without authentication. Authentication and authorization are often (wrongly) understood as being just two sides of the coin. Today, it is generally accepted that a good design clearly separates authentication and authorization.

Trust evaluation in a PKI infrastructure may include:

- Ensuring the certificate is not expired.
- Ensuring the signature of the certificate still matches.
- Ensuring the certificate is issued by a trusted CA (this includes checking the trust chain).

Authentication in a PKI infrastructure may include:

- Ensuring the user has the private key to the certificate that he provides. (Note: This does not mean that the holder of the private key is the intended recipient of the private key).
- Ensuring the user continues to provide the correct SSL session key.
- Performing a trust evaluation on a certificate.

Authorization in a Grid environment may include:

- Ensuring that the DN is present in a configuration file that lists all authorized users.
- Ensuring that an attribute[2] listed in the extension of the certificate has a certain value in order to perform an operation, e.g. installing software.

2.3. VIRTUAL ORGANIZATIONS, SITES AND COMMON SERVICES

In Grid environments users can be organized in so called *virtual organizations* (VO). They allow the management of users across different institutions without the need to observe intra-institutional management structures and policies.

[2] The user's proxy certificate may contain a so-called attribute certificate, which contains attributes describing the user as member in a VO (see Section 2.3).

Sites[3] on the other hand correspond to local installations of computing resources (clusters, storage, etc.). Agreements are made, between the individual sites and VOs that give the members of a VO access to the resources of the sites. In order to do so, sites must install the services of the middleware chosen by the VO. It is important to note that the support of VOs by sites does not mean that the site hands over some of its autonomy, particularly in security issues, to VOs. On the contrary, the local site autonomy must be preserved and respected by the VO and its members at all times.

Besides VO and sites, there are also a set of *common services* that act as a glue between VOs and sites. They can be considered as a part of the underlying Grid infrastructure and are typically operated by some of the larger sites on behalf of the infrastructure. Examples are information services, credential stores, management services for VOs etc.

Figure 1 shows the high level picture of the EMI components divided into the elements of VOs, sites and common services. The EMI middleware consists of the following software components:

- User Interface UI: Suite of components through which the user interacts with the Grid (submitting of jobs, querying component status, access to data stored on the Grid, etc.). The user needs to possess a X.509 certificate in order to interact with the Grid services, because the EMI security model is currently entirely PKI-based. The UI contains clients that interact with ARC, gLite and UNICORE services.
- CA: A certificate authority provides the X.509 credential to the user. The CAs are coordinated through the international Grid trust federation (IGTF) and are outside the purview of the EMI security project. Note that CAs also issue host or service certificates to identify hosts and services.
- VO services: Users are permitted to access the Grid due to their VO membership. Currently; ARC and gLite use the Virtual Organization Management Service (VOMS). UNICORE uses UNICORE Virtual Organization Service (UVOS).
- Common services: These are services that span the Grid and may serve one or more VOs at the same time. Examples are the information system, file catalogues and resource brokers.
- Credential services: The special class of services, which act either as:
  - An attribute authority (AA)[4]
  - A proxy certificate store and renewal service (MyProxy [R17]).
- Site-specific software components: These are the Grid services that the individual sites operate. They comprise of Compute Elements (CE) and Storage Element (SE) services. The user can either access these services directly or through resource brokers and/or file catalogues.

[3] The terminology of the Joint Security Policy Group has been used in this document; refer to the glossary for a definition of the term "site".

[4] VOMS is the AA in use.

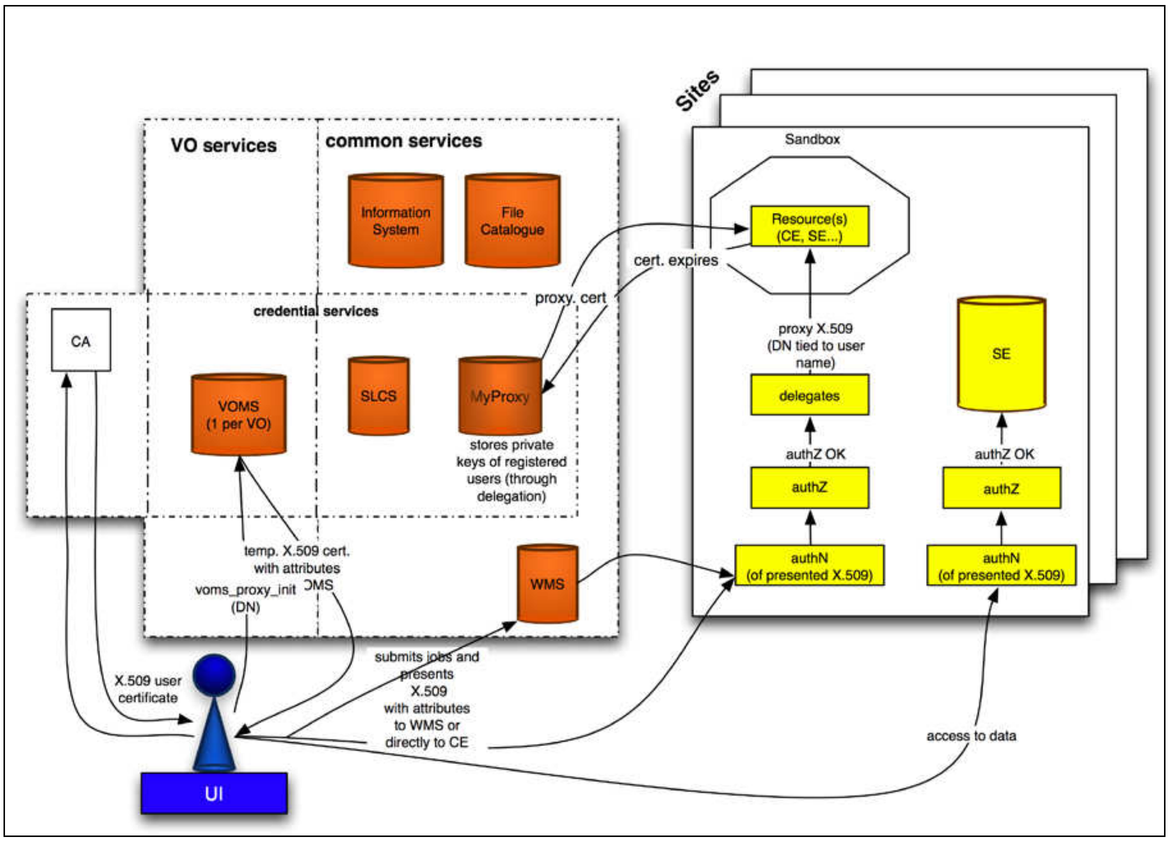

Figure 1 Overview of the EMI architecture

Table 2 lists the EMI-supported software components. Each of these components must support the needed security features as given by the above-mentioned facilities.

Class of Service	Software Component	EMI supported instance
VO services	VO management system	VOMS and UVOS
	User ID protection	pseudonymity
Common services	Information system	Berkeley Database Information Index (BDII), EMI Registry
	File catalogues	LCG File Catalogue (LFC)
	Store for proxy certificates	MyProxy
	Short-lived (site-integrated) credential services	Security Token Service (STS)
	Resource broker	Workload Management Service (WMS)
	File Transfer Service	File Transfer Service (FTS)
Site-specific services	Compute Element (CE)	Compute Resource Execution and Management Service (CREAM), ARC A-REX
	Storage Element (SE)	Disk Pool Manager (DPM), dCache, StoRM, Hydra

	Authorization service	Argus
	User identity switching	gLExec
	Virtual Machine provisioning	Argus-EES

Table 2: EMI Security Components

3. AUTHENTICATION

Authentication[5] (AuthN) is concerned with identifying entities (users, agents, and services) when establishing a context for message exchange between principles. One of the key aims for Grid authentication is enabling single sign-on for the user – using a single identity credential with "universal" value across many different infrastructures, different communities and virtual organizations and multiple applications. As such, attention must be paid to the fact that the same identity is also to be used for other purposes: accessing non-Grid resources such as networks and web resources; or in interaction with government and administration.

The authentication model for EMI is largely based on the EGEE model that uses the concept of *trusted third parties* (TTPs): entities that are not related to any *relying party* except through a trust relationship. Underlying that trust relationship is the digital signature of the TTP, based on conventional asymmetric cryptography. The TTP will bind the cryptographic key pair to one or more identifiers that represent the entity. Although theoretically a single TTP could service the entire community, in practice a mesh of TTPs exist. The mesh defines a common authentication domain, by grouping the resources, users, and services that agree to use a common set of TTPs for authentication. The authentication domain, however, does neither imply common rights-of-access nor does it constitute an "infrastructure" of sorts. Although the EGEE project assumes a set of TTPs is used, it does not suggest that all entities accept this set. This introduces an additional failure mode that higher-level services should cover anticipate and handle.

The strength of an authentication credential issued by a TTP is dependent on three things:

- The trust on the TTP, in particular the TTP's operations, procedures and general conduct.
- The quality of the original identity vetting. No amount of technology can overcome weak identity checking at the source. The use of appropriately qualified authorities in the credential issuing process is important. The vetting can be performed as part of the issuing process when a photo-ID is shown in-person – a system which is supporting of thinly-spread users or relatively small constituencies. Alternatively, it can leverage pre-existing assured credentials, such as those maintained by the subscribers' employer or university, and made available through an authentication and authorization federation (AAI federation). In the latter case, it should be based on in-person vetting at some stage of the process or established to an equivalent assurance level like Kantara LoA 2, but the issuance can pursue automatically without human intervention at that time. This setup requires a mature federation and AAI identity providers, but scales better to large constituencies (examples include the TERENA Certificate Service deployed widely in Europe, or the CILogon service leveraging the InCommon federation in the US).
- The security of the private data needed to prove possession of the credential. This is further discussed in Section 4.

The International Grid Trust Federation (IGTF) and its European constituent EuGridPMA have been established and expanded over the course of the EGEE project. They are the orchestration bodies for the TTPs and define Authentication Profiles (AP) for identity provisioning. The advantages of such a standardized approach across the many partners of EGI as well as across several Grid middlewares and Grid infrastructures cannot be overemphasized.

[5] Large sections of this chapter have been taken from [R1] and adapted to the current situation. This reflects the mature status of the authentication in the security architecture. See also Section 2.2 for a description of the terms trust and authentication.

3.1. IDENTITY CREDENTIAL FORMATS

In accordance with the standards as defined by IGTF, X.509v3 public key certificates [R7] are used to express identity assertions. These certificates are issued by Certification Authorities (CA), which are accredited by the IGTF. Different types of CAs exist (see below), as well as different means of delivering certificates to end-entities. Whatever the delivery mechanism or operational mode of the CA, the authentication is based on at least the distinguished name (DN) of the subject contained therein.

While the distinguished name may contain information about the user such as name, organizational affiliation and email address, this document does not make any assumption on these fields. The distinguished name is only considered as a unique identifier, which bears no semantics. In order to facilitate a single-sign on for Grid resources as part of the authorization process, EMI security components also support *proxy certificates* as defined in [R8]. Proxy certificates are normal X.509 identity certificates, but equipped with an extension that ensures their rejection by applications that do not support proxy certificates.

- Those proxies are used to perform two tasks: single sign-on AND delegation.
- Proxies are a necessary artefact for gLite, ARC and dCache. UNICORE does not rely on proxies for sign-on or delegation. Therefore the all explanations and discussions on proxies are not relevant for the UNICORE stack.
- As a result of some EMI development, UNICORE may be accessed using a proxy certificate. In this case the proxy is not used for delegation; it is used for single sign-on and UNICORE authenticates the credential.

Two different types of certificates can be distinguished based on their lifetime:

1. Long-lived (typically one year) certificates, which are issued according to the “classic” AP or the “Member Integrated Credential Service” (MICS) AP. A MICS certificate is issued by an automated system based on a pre-existing identity management system maintained by an organization or federation.
2. Short-lived (i.e. less than 1 million seconds, approx. 11 days) certificates, which are issued according to the Short-lived Credential Service (SLCS) AP.

It should be noted that most CAs accord to the “classic” AP, with MICS and SLCS based CAs only gaining attention recently.

Over the past few years, security architectures, which do not use X.509 certificates as identity credentials, have gained significant interest. Authentication and Authorization Infrastructures[6] (AAI) based on the concept of federated identity have been established in several European countries, often coordinated by National Research and Education Networks (NRENs). They typically comprise tens or even hundreds of thousands of users. These AAIs use SAML assertions to express authentication information and user attribute values. Whereas the EMI Security team does not foresee replacing PKI as the basis of the security architecture over the next years, it does expect interoperability issues to gain in importance. The key concept to implement is the generic support of security tokens without requiring a specific token format. Security Token Services (STS) can then be used to exchange a given token into one of a different format.

3.2. BOOTSTRAPPING AUTHENTICATION

The process of bootstrapping authentication is well established for services and users. For services the

[6] There exist several implementations of AAIs. Shibboleth of Internet2 has gained the widest acceptance in Europe.

administrator requests a service certificate from an accredited CA, which they install on the host. Typically, the Grid service must then be configured to use this host credential.

For users, it consists of obtaining a certificate from an accredited CA. Before accessing Grid services, the user must create[7] a proxy certificate, which contains a VO specific extension, the VOMS Attribute Certificate (AC). The AC contains the set of attributes that the user has within the context of the VO. (See Section 5 and [R14] on the usage of these attributes in authorization decisions). For the UNICORE case the authentication is bootstrapped with classic SSL/TLS using the regular X.509 client certificate.

3.3. ENFORCING VALIDITY CONSTRAINTS

Proxy certificates are typically stored with a weaker protection level (stored in clear text and safeguarded only by local file system privileges). As a consequence, security policy often declares that proxy certificates should not be trusted if issued with a longer validity: A lifetime of 24 hours is normally requested, with longer-lived proxy certificates mandated to be stored in a keystore such as MyProxy [R17]. Currently, there are unresolved issue that should be resolved by the various operators, users and VOs of the Grid.

3.4. EMI COMMON AUTHENTICATION LIBRARIES

The EMI Common AuthenticatioN Libraries (CANl) are a set of libraries implemented along common principles, following common API architecture, and introducing shared error codes. Multiple language versions are available, namely C, C++ and Java, each implemented in parallel but given the shared development effort, different language versions on server/client side are mutually compatible. On top of the common authentication features, each language version can offer additional functions required in the application area.

3.4.1 C library

The C version of CANl provides basic support for communication management channel establishment and message protection (creation of secured sockets on server and client side). It is done via core API without a direct SSL/X.509 dependency. Thus CANl C is easily portable outside the PKI world.

A higher level API provides support for SSL and X.509 (SSL specifics for connections can be set). CANl C handles certificate management (proxy certificates are fully supported), i.e.:

- Proxy certificate signing request generation (including key pair generation with some restrictions on key length with respect to generated certificate lifetime).
- Certificate signing and verification (CRL and OCSP are supported).
- CANl C also supports usage of hardware security tokens – PKCS #11.

The CANl C library has already been successfully integrated into the gridsite software package. The reduction and simplification of the gridsite code was significant. The adoption of CANl C in L&B (Logging & Bookkeeping) and other ANSI C-based products is also planned.

3.4.2 C++ library

The C++ EMI common authentication library provides the C++ interface for credential manipulation and secure communication.

The credential manipulation includes CSR (certificate signing request) generation, proxy certificate signing, EEC (end entity certificate) signing (useful for short lived certificate signing), and certificate verification. The certificate verification supports the CA namespace constraints policies. Moreover, the certificate verification supports the checking of certificate status through certificate revocation list

[7] This process is described in detail in [R14]

(CRL) or online certificate status protocol (OCSP) [R11]. For certificate status checking with OCSP, both OCSP and its alternative approach - OCSP stapling (formally known as the TLS Certificate Status Request extension [R30]) are supported. The OCSP stapling is supported in the secure communication and invoked once the peer certificate is verified.

For the repository of certificates, besides the de-facto file-based credential, C++ library also supports the NSS database-based credential through PKCS11 standardization.

The C++ EMI common authentication library has currently been integrated into the pre-production version of one of the ARC client utilities - arcproxy. Therefore, users can create proxy certificate by directly using their credentials from NSS credential database, e.g., credentials used in their Mozilla Firefox web browser.

3.4.3 Java library

The Java version of the EMI X.509 Common Authentication Library provides support for:

- Off-line certificate validation
- Creation of SSL sockets (both server and client side)
- Handling certificate DNs, also in text format
- Usage of proxy certificates: from proxy generation to validation
- Support for CRLs and OCSP

The library provides implementation of an easy to use and flexible validation logic, which can take advantage of different trust material sources. Proxy certificates, which are commonly used in the grid environment, are fully supported.

The library can be easily integrated with the standard JSSE stack. Therefore the library does not handle the SSL/TLS communication by itself, as this functionality is provided by the standard Java library. Moreover the library can be easily integrated with all popular Java communication solutions as HTTP client libraries, JMS libraries, Web Service stacks and others.

Another unique feature of the Java EMI Common authentication library is the support for different formats of trust stores (collections of trusted root certificates) such as Openssl-like, plain directory with certificates or native Java keystores. The support for multiple formats of trust stores is required to maintain backwards compatibility for applications adopting the Java EMI common authentication library. PKCS11 and support for hardware security tokens is currently not provided as this requires platform dependent drivers. The counter to the fundamental Java principle of portability and therefore in the many cases would not be feasible to use.

3.5. REVOCATION

There are good reasons why the binding between the public part of a key pair and an identifier, or a set of identifiers, should be revoked. These include the compromise of the associated private key or invalidation of one or more identifiers in the binding. The longer the key-to-identifier binding is considered valid, the higher the probability that such a binding will become invalid.

The TTP is responsible for revoking credentials it has issued, but it is up to the relying parties (both services and requesters in case of mutual authentication) to ensure that this revocation information is consulted. The basic revocation information is distributed as a Certificate Revocation List (CRL) [R7]. All authorities recognized as TTPs by the major EMI stakeholders issue revocation information at least through such CRLs (although local TTPs in individual installations may not provide this feature and thus the software needs to be resilient to a lack of CRL information).

However CRLs are neither the most efficient nor the most secure way to distribute revocation data. On the one hand CRLs are pre-fetched by relying parties regardless of their actual use (not every RP will be continuously validating credentials from all TTPs), and this periodic retrieval is burdensome on both the RP bandwidth as well as the TTP CRL distribution infrastructure, especially for large TTPs. On the other hand, the periodic retrieval introduced delays in revocations becoming effective, given the period between subsequent downloads. Timely identity revocation is needed to prevent exploitation of credentials that have been compromised. The allowed response time as specified by sites is in the order of 10-60 minutes. However, this time cannot be achieved through the periodic distribution of CRLs to all parties at a pan-European (or larger) scale.

Therefore, any software component that performs certificate validation should preferably be able to check the validity of the credentials in real-time. The Online Certificate Status Protocol [R11] (OCSP) is a widely used protocol that facilitates the validation or credentials in real-time. Large-scale deployments of OCSP, after it had been introduced in major web browsers and for high-volume web sites, have however demonstrated the scaling limitations of traditional OCSP responses. Revised profiles of the OCSP standard, in particular light-weight OCSP [R11b], offer better scalability and facilitate deployment by the TTP infrastructure. Although the profiling of light-weight OCSP for deployment in the e-Infrastructure is still in progress in the Open Grid Forum, EMI has developed implementations addressing the initial use of OCSP. At this time, only few TTPs have made OCSP end-points available as a production service and included the appropriate meta-data in the issued certificates, but the expected to change with wider availability of OCSP capable software.

The fact that OCSP is a real-time protocol does introduce operational dependencies on the availability of the OCSP responders. This has a potential impact on service availability unless responses are cached and fall-back to other mechanisms such as CRL parsing is implemented. Fall back to CRL is set by default in all EMI validation code, and caching mechanisms are being developed.
This availability issue (with the corresponding security implications of a deliberate denial-of-service against the TTP) is being addressed by the IETF TLS working group via “OCSP Stapling” as part of the TLS protocol. At this time, the stapling work is not yet sufficiently mature to have been implemented in the server-side code for endpoints.

As such CRLs remain for the foreseeable future the main revocation mechanism.

3.6. CERTIFICATE RENEWAL

Certificates are equipped with a validity time-stamp and, as such, they expire and need to be renewed. There are several techniques that a CA may use to facilitate a trusted remote credential renewal, which are not covered in this document. For instance, the CA may choose to: a) simply issue a new certificate to the existing user-held key pair; b) require the user to generate a new key pair; c) countersign the new key with the old and so on.

3.7. DELEGATION

Grid computing is somewhat unique in using X.509 credentials for both server and end-user identity assertions. In contrast, most commercial systems use X.509 only for identifying servers, with users identifying themselves with some ostensibly lighter-weight solution, such as user-name or email address and password. However, identifying a user via X.509 provides some advantages, including as a natural method of allowing identity delegation via X.509 proxy certificates.

Often, grid users need to allow a remote service to conduct some activity on their behalf. For example, a user may need to transfer data from a remote storage service to some distinct, remote computational service to analyze that data. To achieve this, either there needs to be a trust relationship between these services or one of these services must act on behalf of the grid user for the data transfer. As such actions may be difficult to predict and grids tend to have a large number of services, it becomes prohibitively expensive to establish pair-wise trust relationships between the services or delegate the necessary credentials ahead of time. Therefore a dynamic mechanism to allow delegation is needed.

An important security aspect of delegation is the principle of least privilege: one should delegate only as much privileges as is necessary for the desired operation. RFC3820 describes a mechanism for constraining delegation by allowing policy elements to be embedded in the credential; however, this has proven hard use and is currently not supported. The exception is support for "limited proxy certificate", which are delegated certificates that inherit all rights of its parent except for job submission.

There is currently no agreed standard process by which a client may undergo identity delegation with a server: no standards body has developed a protocol to allow X.509 delegation. However, there is a common need for delegation and, as a result, there now exist several methods for delegating within different communities (even within a single middleware), each of which is incompatible with the others.

Within EMI there was a consultation process to decide on a common approach to delegation. This process resulted in a decision to use GridSite Delegation (GSD). This was due to the existence of client and server support for both Java and native binary libraries. This level of support means that GSD is easy to adopt by any EMI packages that need X.509 delegation.

Figure 2 shows an example of GSD in use. The client wants a server to undertake some operation that will involve it interacting with some remote, third-party server. Using GSD, the client first checks whether the first server already has an appropriate delegated credential. On discovering no suitable credential exists, the client starts the delegation process, in which the first server generates a key-pair and a corresponding certificate signing request (CSR), delivers the CSR to the client. The client then signs the request and returns it to the server. This process results in a new delegated credential available from the credential storage at the server. When triggering some activity that requires this delegated credential, the client passes a reference to the delegated credential. The server makes the request to the remote server using the referenced credential. Once the operation completes, the client instructs the server to destroy the delegated credential to reduce the risk that the delegated credential is compromised.

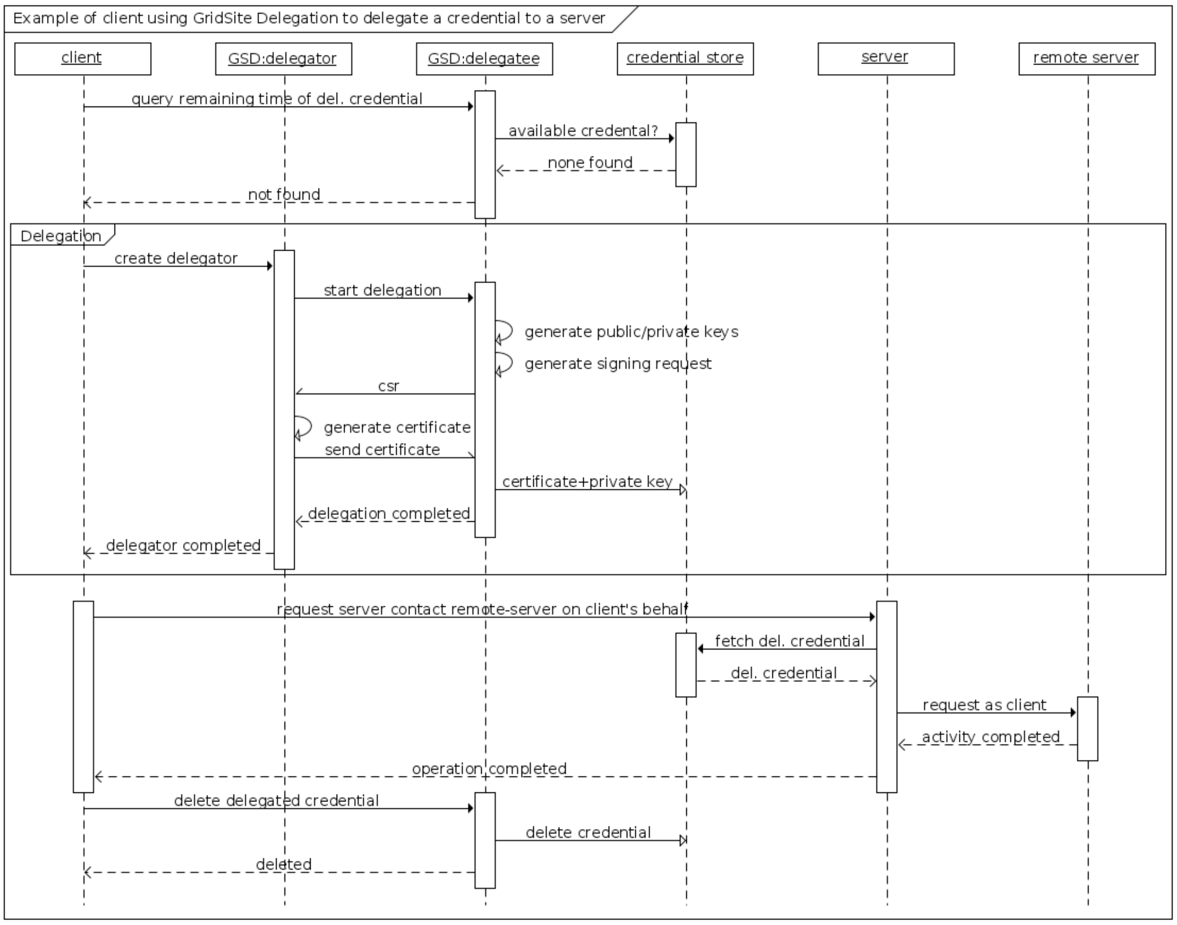

Figure 2: Gridsite delegation process

The agreement within EMI to use GSD was codified in a formal document [R39]. The document describes both delegation in general and GSD specifically. It outlines the responsibilities of the project teams to adopt GSD. The document also describes some of the deficiency of GSD, principally some problems with the existing technical description of GSD, which conflates protocol definition with implementation-specific decisions. The agreement document also includes plans for correcting these problems.

As mentioned above, UNICORE does not use proxy certificates for delegation, in order to eliminate all proxy-related problems as difficult tracing of invocation chain (proxy certificate impersonates the initial issuer) and the problems described in Section 3.3. Instead, in UNICORE the delegation process is fully separated from authentication and is performed by issuing special delegation assertions, signed by the issuer. This model called Explicit Trust Delegation is described in details in [R34]. As the delegation assertions are designed in such a way to be usable only by otherwise properly authenticated users, they may be public and there is no strong security requirement on reducing their validity period to couple of hours as in case of proxy certificates. This also eliminates the need for their renewal, which, in the case of proxy certificates, is described in the next section.

3.8. RENEWAL OF PROXY CERTIFICATES

The renewal of proxy certificates must be considered separately. It is necessary in order to support long running jobs on the Grid.

The renewal procedure (see Figure 3) starts with delegating (see Section 3.7) a user proxy certificate of the user into an online keystore such as MyProxy [R17]. The delegation procedure will ensure that a newly created public and private key pair has cryptographically become a part of the certificate chain stored in a MyProxy service. This stored delegated proxy must be of a longer-lived type, which means that it should have a maximum lifetime in the order of two weeks. The user then typically submits jobs using a short-lived proxy with a validity length in the order of one day.

When the short-lived proxy is about to expire, a renewal service[8] or daemon must detect this and prior to expiration it tries to request a new delegation from the previously used online keystore. The renewal request to the keystore must be authorized using the almost-expired delegated proxy and can only be accepted from pre-configured trusted services. The keystore will use the stored long-lived delegated proxy to create a new short-lived delegation to trusted renewers. This procedure can continue as long as a valid proxy resides in the keystore. The entire certificate chain needs to be valid at each level in the chain.

In a VOMS-enabled (see Section 5) deployed Grid infrastructures such as gLite or ARC, a few extra steps have to be taken when renewing the proxy certificate, as not only the proxy certificate but also the VOMS AC must be renewed. This could be taken care of by the keystore itself. However, the currently deployed keystore MyProxy does not offer this functionality. Therefore the renewal service must contact the user's VOMS server and assemble the VOMS extension into the proxy.

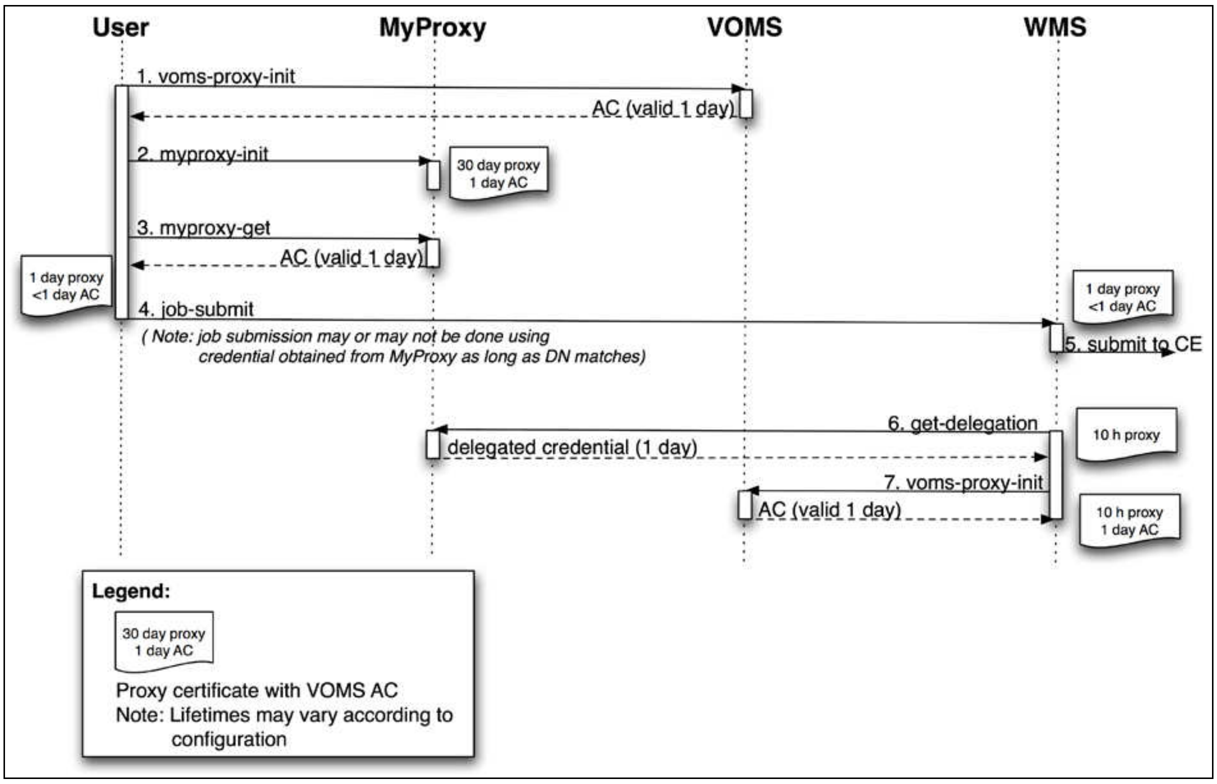

Figure 3: Proxy Renewal Mechanism

[8] This functionality is currently been provided by the renewal service, which is a part of the gLite Workload Management Service (WMS). The FTS service also supports a renewal mechanism for long lasting transfers.

3.9. ANONYMITY, PRIVACY, PSEUDONYMITY

Application and user requirements[9] call for the possibility of anonymous use of the Grid system: an outsider should not be able to deduce a particular user's activities, such as how much of the resources the user consumes or what applications are run. Such *information creep* is of serious concern to applications in areas of highly competitive research, such as biomedicine.

Grids are open-ended distributed systems. As such, protecting ourselves against information creep is prohibitive. Even if all information and message exchanges were made on encrypted and authenticated connections, analyzing the message exchange patterns would still allow an adversary to deduce information on how the system is being used. In fact, true anonymity can only be achieved by all parties sending continuous streams of (bogus) data to all other interacting parties at all time: the common tactic in the military and intelligence world.

Likewise, the only way in which you can obtain true privacy is by not sharing information or data at all, which is contradictory to the very nature and basis for Grid computing. In addition, many sites attached to the Grid require in their local site policy the full disclosure of the identity of the user regardless whether the local user of the site or a remote user accessing the site through the Grid. Thus, for these requirements, EMI provides best-effort solutions by the addition of a pseudo-anonymity (pseudonymity) service and defer on privacy and anonymity.

The pseudonymity service swaps the real identity of a Grid user for a pseudonym, thus hiding it from immediate exposure in logs and on the network. The pseudonymity service acts in all regards as another TTP, with the addition that it is also trusted to maintain the relationship between the pseudonym and real identity in confidential and secret manner. This trust has to be also kept unless law enforcement or a similar legitimate body requires it as part of, for example, an investigation on malicious use.

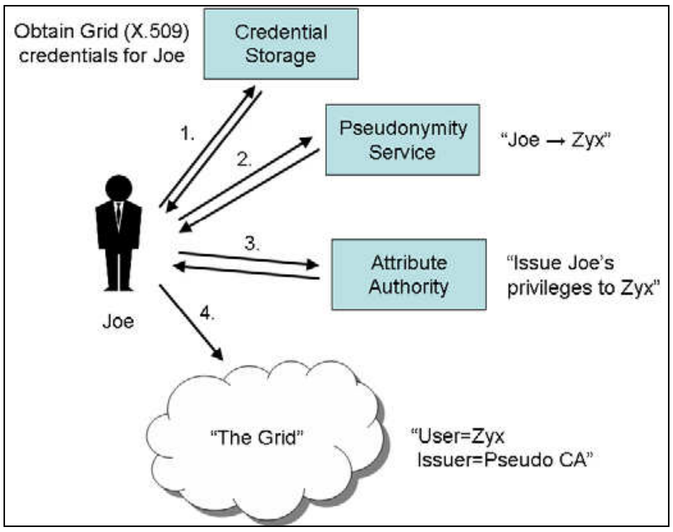
Obtain Grid (X.509) credentials for Joe
Credential Storage
1.
2.
Pseudonymity Service
"Joe → Zyx"
3.
Attribute Authority
"Issue Joe's privileges to Zyx"
Joe
4.
"The Grid"
"User=Zyx Issuer=Pseudo CA"

Figure 4 Use of the pseudonymity service

To acquire a pseudonym identity, a user performs the following steps (see Figure 4):

1. The user obtains their normal authentication credential, for example: Joe.
2. The user authenticates using Joe to the pseudonymity service, which then issues a short-lived identity, Zyx, to the user.

[9] Note that national as well as EU legislation also set requirements on privacy and anonymity.

3a. The pseudonymity service registers the pseudonym, Zyx, to the attribute authority as an alias to the original identity, Joe.

3b. The user authenticates to the attribute authority using Zyx credentials and obtains the attributes originally issued to Joe.

4. The user is now bootstrapped with a pseudonym credential Zyx and the necessary access privileges bound to that credential to make use of Grid services.

In the Grid VO model, the user's real identity is revealed to the pseudonymity service (the authentication trust anchor) and the VO service (the authorization trust anchor). The assumption is there is no need to hide the true identity of a user from the other members of the same VO.

The pseudonymity service has been implemented during EGEE-II and is part of the EMI software release.

4. FEDERATED IDENTITIES

The experiences of the past years have shown that the management of private keys by the end user is a significant entry barrier for the wide adoption of Grid technology. PKI technology has turned out to be not easily understood by many users, and the average user often ends up leaving weakly protected copies of this certificate on various untrusted systems.

Several other technologies were investigated during the all phases of the EGEE projects, but no final conclusions had been reached by that point [R1, R2, and R28].

In the EMI project a new general purpose service, Security Token Service (STS) has been designed based on the responses given in the "AAI needs of the DCIs" workshop [R29].

As Security Assertion Markup Language (SAML) can be considered as the de-facto standard in the national and international federations, federated identities are supported in the EMI by transforming the SAML assertions into the Grid identities using STS.

4.1. SECURITY TOKEN SERVICE

The Security Token Service (STS) is a partial implementation of the OASIS WS-Trust specification. It is a service that can be used for transforming an existing security token into another security token format. A security token is defined in the WS-Security specifications as a collection of claims that can be attached into a Web Service message. The WS-Trust specification employs an open content model within its request and response messages. This provides maximal extensibility but also means there is a theoretically infinite number of messages that could be produced and yet remain compliant with the specification. EMI STS uses a WS-Trust interoperability profile which attempts to strike a balance between the extensibility offered by the WS-Trust specification and the need to scope that functionality into a manageable set.

The token formats that will be supported by EMI STS implementation include Username/Password token, X.509 certificate, X.509 proxy certificate and SAML assertion. Existing external sources such as online Certificate Authorities, online credential repositories, VOMS and SAML authorities can be exploited by STS for aggregating the required information for the security tokens issued. By enabling the token transformation, STS can establish a trust relationship between different security and application domains. The current implementation of the STS focuses on a subset of all the possible functionality outlined with the WS-Trust specification. Specifically it implements the issuance of X.509 certificates and proxy certificates by exploiting the supported security tokens.

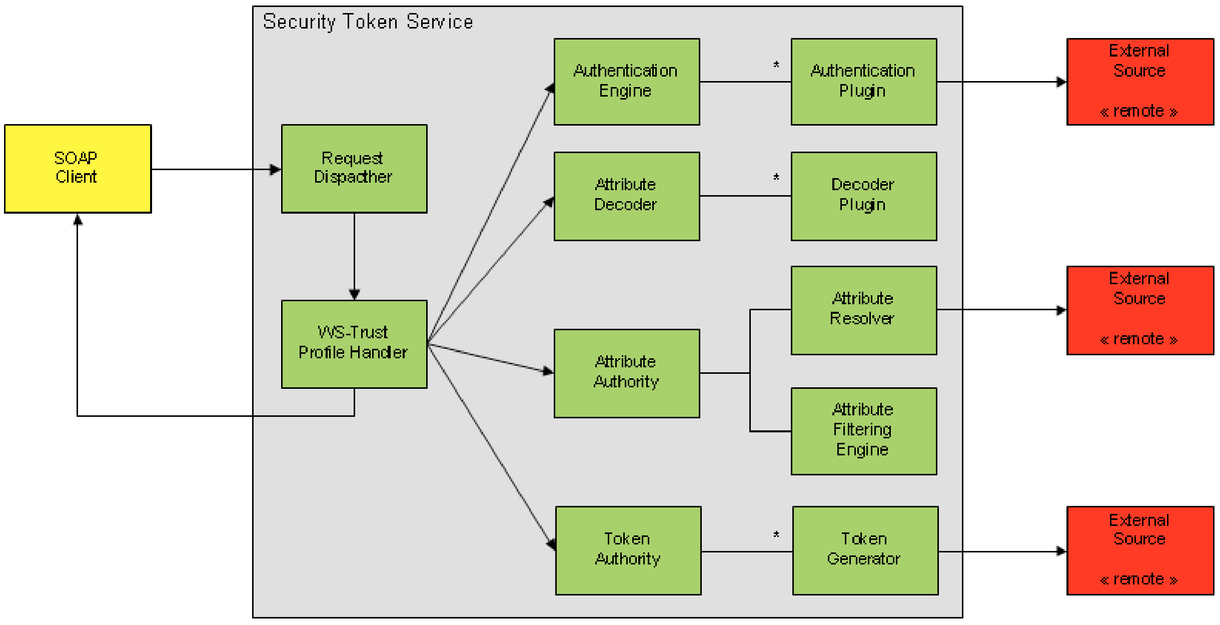

Figure 5: Overview of STS components

From the point of view of the client the STS is a Web Service that, like any other Web Services, is accessed using the SOAP protocol. This means that any party capable of producing specified request messages and understanding response messages can act as a client for the STS. As the SOAP protocol is extremely widely adopted, many building blocks exist for implementing a client to STS. This was not dependent on the platform, programming language or even deployment: the STS client can be for instance a simple command-line tool, a local heavy client with rich GUI, or a Web portal. Figure 5 illustrates the overall architecture of STS. The architecture is modular and support for additional security token formats can be easily added.

5. AUTHORIZATION

5.1. INTRODUCTION

Authorization (AuthZ) is concerned with allowing or denying access to services based on *policies*. The core problem with authorization in the Grid environment is how to handle the overlay of policies from multiple administrative domains (VO policies, operational procedures, policies of the local sites) and how to combine them.
Early Grid deployments based authorization on the identity of the user (e.g. the Distinguished Name of the certificate). Whereas this approach is very simple and intuitive, it does not scale or support more sophisticated authorization policies.

Attribute-based authorization overcomes the limitations of identity-based authorization and is the authorization mechanism that has been chosen for EMI. It requires two kinds of components:

- Attribute Authorities (AA), which associate a user with a set of attributes in a trusted manner to a relying party by way of digitally signed assertions. Note, that several attribute authorities in a Grid deployment may assert attributes. EMI currently supports one two AA - the VO Membership Service (VOMS) [R12] and UVOS. VOMS is considered the primary solution as it may be used by all middleware stacks, while UVOS only by UNICORE. VOMS assigns two types of attributes[10] to the user: memberships in an arbitrary number of hierarchical groups, and roles that the user may or may not assert on a case-by-case basis. VOMS issues X.509 attribute certificates (AC) [R13], which contain a list of strings, so-called Fully Qualified Attribute Names (FQAN), corresponding to the role and group membership associated with a particular user. These ACs are then embedded in the proxy certificate of the user and are thus available to the Grid resources at authorization time. A detailed description of the group and role mechanism and the role of AC can be found in [R14].
- The relying party (the resource) evaluates the attribute assertions and includes them as "evidence" added to a *context*, which in turn is used when evaluating an access request against local policy. As an example, a CE evaluates the FQANs and authorizes the user based on their values (see [R14] for details).

5.2. VOMS AND UVOS

As mentioned in Section 2.3, in Grid environments users are generally organized into *virtual organizations* (VOs). A VO is defined as a group of users and resources (computing) that come together for a period of time to perform a particular task. A VO allows the management of users across different institutions without the need to observe intra-institutional management structures and policies. In practical terms, the management of a VO is facilitated by a Virtual Organization Management System (VOMS). In gLite/ARC and UNICORE these are gLite VOMS and UVOS respectively.

The EMI project made effort to provide unification in this area. In effect UNICORE middleware can use a VOMS server to obtain user attributes and VO membership information.

5.2.1 UNICORE Virtual Organization System (UVOS)

UVOS is a VO-aware attribute information point that is an alternative to the gLite VOMS. UVOS provides several minor features related to attribute management and deployment scenarios which are not available in VOMS. These features are necessary for the internal security model of UNICORE. A diagram of UVOS in UNICORE can be seen in Figure 6:

[10] VOMS also supports a third type of attributes: the Generic Attributes (GA). They are arbitrary key-value pairs that the VOMS administrator defines. Thus, they are VO-specific and have no meaning for the generic infrastructure. Therefore, they are not used by EMI services nor are policies for their usage defined. These are not considered in this document any further.

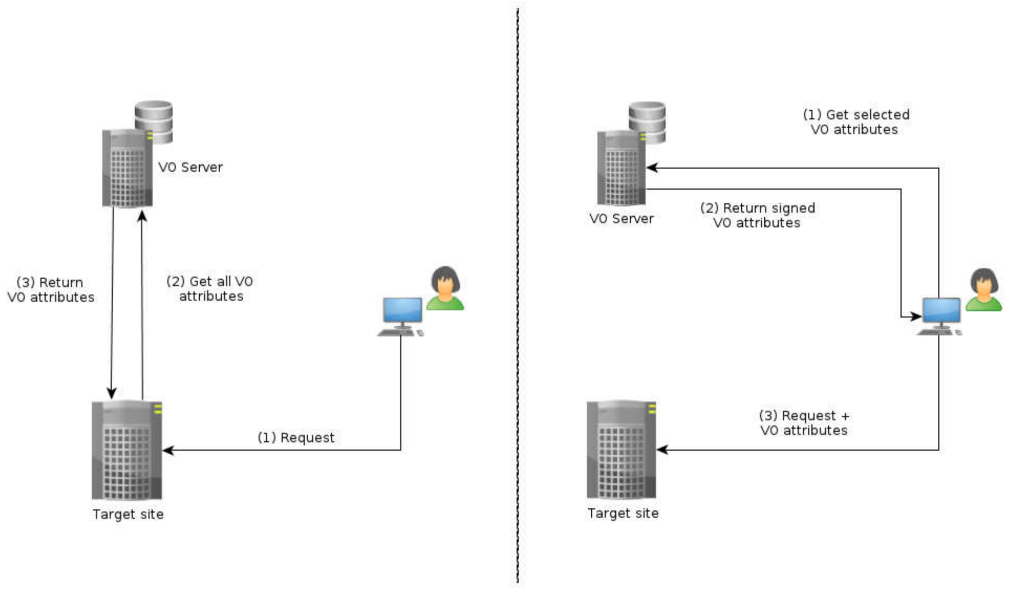
VO Server
(3) Return VO attributes
(2) Get all VO attributes
(1) Request
Target site
(1) Get selected VO attributes
VO Server
(2) Return signed VO attributes
(3) Request + VO attributes
Target site

Figure 6: UNICORE UVOS

UVOS offers only the SAML interface to query for attributes. Therefore UVOS can be used exclusively with the UNICORE middleware stack as gLite and ARC may only consume attributes embedded in proxy certificates. The EMI project maintains UVOS for existing UNICORE deployments requiring its specific functionality, but UVOS is not pro-actively enhanced. UVOS has also an alternative use case: it can be used as an authentication service allowing for integration of its users with SAML-based federations.

5.2.2 Virtual Organization Management System (VOMS)

The Virtual Organization Membership Service (VOMS) is an attribute authority which serves as the central repository for VO user authorization information. VOMS provides support for sorting users into group hierarchies, keeping track of their roles and other attributes in order to issue trusted attribute certificates and SAML assertions used in the Grid environment for authorization purposes.

The main idea of VOMS is to augment a user's credentials with additional attributes, thus allowing services to retrieve them and take authorization decisions on their basis. In a typical Grid job submission scenario, a Grid user requests their bag of VOMS trusted attributes from the VOMS service (typically included in the form of an X.509 Attribute Certificate embedded in an X.509 proxy certificate) and submits a job to the Grid together with the X.509 proxy certificate. These attributes will then be used to authorize operations according to the user position in the VO context, see Figure 7.

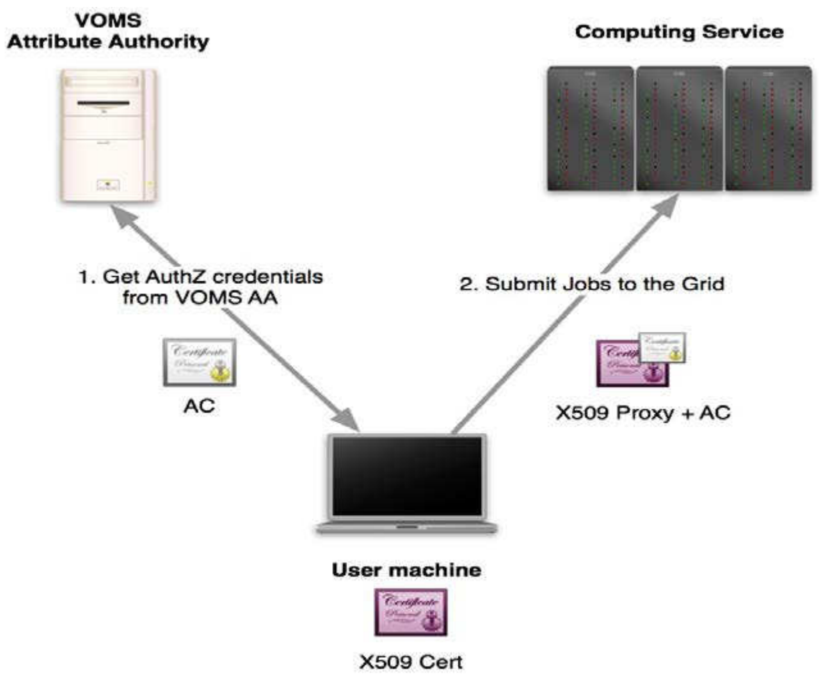

Figure 7: VOMS concept

The VOMS Attribute Authority can issue the following types of attributes:

- **Groups** reflect the VO internal structure by defining a tree where membership in a subgroup implies membership in a parent group.
- **Roles** are used to assign special privileges to users in the context of a VOMS group. Roles are issued only on explicit user's request.
- **Generic attributes** are (name, value) pairs which can be used to represent user attributes that do not directly map to the organizational structure of the VO (its groups and roles) but identify some other property on which authorization decisions can be taken.

The groups and roles are encoded using Fully Qualified Attribute Name [1] (FQAN) attributes. Each FQAN contains the user membership information regarding the VO and groups and also the roles the user may assume. An alternative SAML encoding of the groups and role membership has been developed during the EMI project to support the Common Virtual Organization Attribute profile [R38].

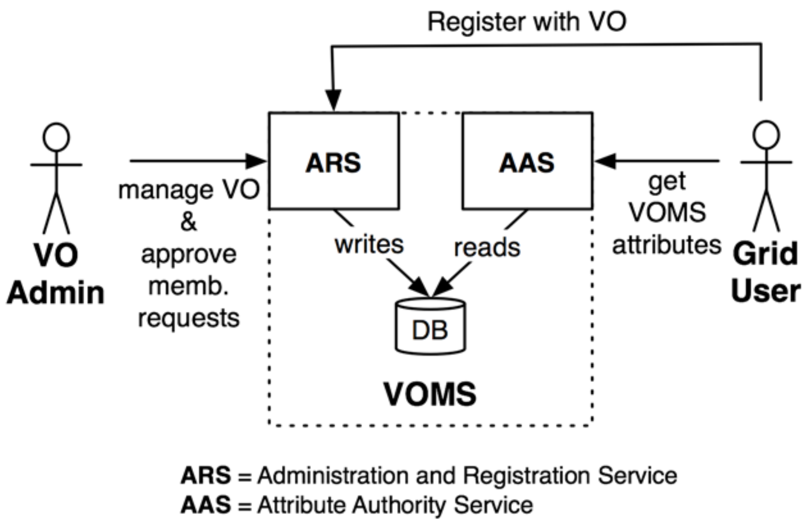
Register with VO
VO Admin
manage VO & approve memb. requests
ARS
AAS
writes
reads
DB
VOMS
get VOMS attributes
Grid User
ARS = Administration and Registration Service
AAS = Attribute Authority Service

Figure 8: The VOMS Architecture

VOMS is composed of two main components:

- The VOMS core service, which issues attribute certificates to authenticated clients;
- The VOMS Admin service, which is used by VO manager to administer VOs and manage user membership details and also implements a SAML attribute authority.

Besides the above components, VOMS provides Java and C++ APIs that are included in all major Grid services to retrieve and validate VOMS attributes.

5.3. PROFILES FOR POLICY DEFINITION

The authorization service enforces *access policies.* These policies are specified in a language such as the XML-based XACML [R36] (the eXtensible Access Control Markup Language). XACML is very generic - it merely specifies the syntactic structure of access rules and their combinations. In order to support a given application like ours (controlling access to Grid resources), XACML needs to be enriched with a "vocabulary" of *attributes* to be used in access rules. Such vocabularies are called *profiles.* Profiles define not only the names, types, and allowed cardinalities of attributes, but also, importantly, textual descriptions of the semantics, i.e. how the attributes should be set and interpreted. Two such profiles have been developed in EMI:

- The *Common XACML Authorization Profile* [R37] defines a common set of attributes for authorization decisions. This includes attributes for "subject" principals and their means of authentication, Virtual Organizations (VOs) and groups within them, resource owners, and different *actions* to which access is desired. The profile also includes attributes which can enrich the *outcome* of the authorization decision beyond a simple "yes" or "no" response: These take the form of *obligations* (to the consumer of the decision), which can contain POSIX user/group IDs and environment variables to be used in the resource's local execution environment. See Section 5.4 below on how this can be used.
- The *Common Virtual Organization Attribute Profile* [R38] specifies attributes for expressing a user's VO memberships, as well as group memberships and roles within those VOs. This profile is intended for use with *SAML assertions.* It provides an important bridging function between those middleware "dialects" that use X.509 proxy certificates based on VO information, and others that use SAML to assert attributes about users.

5.4. ARGUS AUTHORIZATION SERVICE

The Argus authorization service was developed in the EGEE-III project to address the lack of a consistent and accurate application of authorization policy across the Grid. It has been continued and adopted by EMI as the standard Authorization service. The initial primary focus of the authorization service is limited to the problem space of access control to services in general or service operations in specific. Other specific issues like data access authorization have not been addressed yet, although every effort has been made to ensure that the service is sufficiently general to apply to other areas as well.

5.4.1 What is a Policy?

The core concept of the authorization service is that of a policy. Policies, however, are nothing new in Grid middleware. The gridmap files currently used in gLite are an example of a very simple policy. However some deployers may wish to communicate more robust policies, for example:

1. Do not allow jobs from anyone on the site's banned list.
2. Do not allow jobs from anyone on a Grid-wide banned list[11].
3. Allow only jobs that execute programs present on the site's application white list.
4. Allow individuals from the site to submit work to site resources and, from 8.00 – 17.00, give their work a priority of “highest”.
5. Allow any individual to submit work to site resources. From 8.00 – 17.00 given their work a priority of “low” and a priority of “normal” any other time.
6. All jobs started by a pilot job[12] must use a credential associated with the same VO that from which the pilot job originated.

As can be seen from the examples, policies may pull in information from other sources (e.g. a Grid-wide banned list), be based on submitter, environment (e.g. time), and job (e.g. program to execute) properties, and require certain responsibilities (e.g. giving work a specific priority) for an affirmative decision to be valid.

5.4.2 Service Goals

The primary goal of the authorization service is to accurately and consistently apply authorization policies across Grid resources. This process starts by allowing authoritative individuals to write and maintain their policies thus increasing the chance that the policy will be accurate. These policies are then automatically distributed across the Grid in a relatively short period of time (i.e. minutes to hours) ensuring that all services are eventually operating with a consistent, up-to-date, set of policies. Finally, all services use the same policy evaluation engine and so always end up with consistent results.

At a more technical level the service also attempts to provide very good tools and information to deployment and troubleshooting staff. For example, robust audit logs that may be used to track users and simple command line tools for adding new policies or determining an effective, point-in-time policy. Additionally, the components of the service are designed to be very scalable and highly resistant to failure, when deployed in a network configuration. The authorization service client is kept as small and simple as possible to allow for easier deployment (i.e. to avoid library dependency conflicts) and lower the barrier of entry in creating clients in other languages. Lastly, a great deal of effort is invested to make sure that components can be deployed in numerous models in order to allow a balancing of factors like complexity, latency, and response time.

5.4.3 Authorization Service Architecture

The authorization service is made up of four individual components (see *Figure* 9). The Policy Administration Point (PAP) is the repository for policies. Generally speaking, each organization that contains an author of policies will run a PAP. The Policy Decision Point (PDP) is the policy evaluation

[11] E.g. the Operational Site Coordination Team (OSCT) of EGEE could operate such a Grid-wide banning list.

[12] Pilot jobs are explained in Section 5.5.

engine. The Policy Enforcement Point (PEP) acts as the client to the authorization service. It sends authorization decision requests to the PDP and provides the resulting information back to the invoking code (e.g. CREAM, glexec). The Execution Environment Service (EES) works with the PDP to provide the constraints, or context, under which an authorization decision is valid (e.g. run the job under this UID/GID).

The communication between components is performed using the eXtensible Access Control Markup Language (XACML) profile for the Security Assertion Markup Language (SAML). The policies exchanged between PAPs are expressed in the XACML policy language.

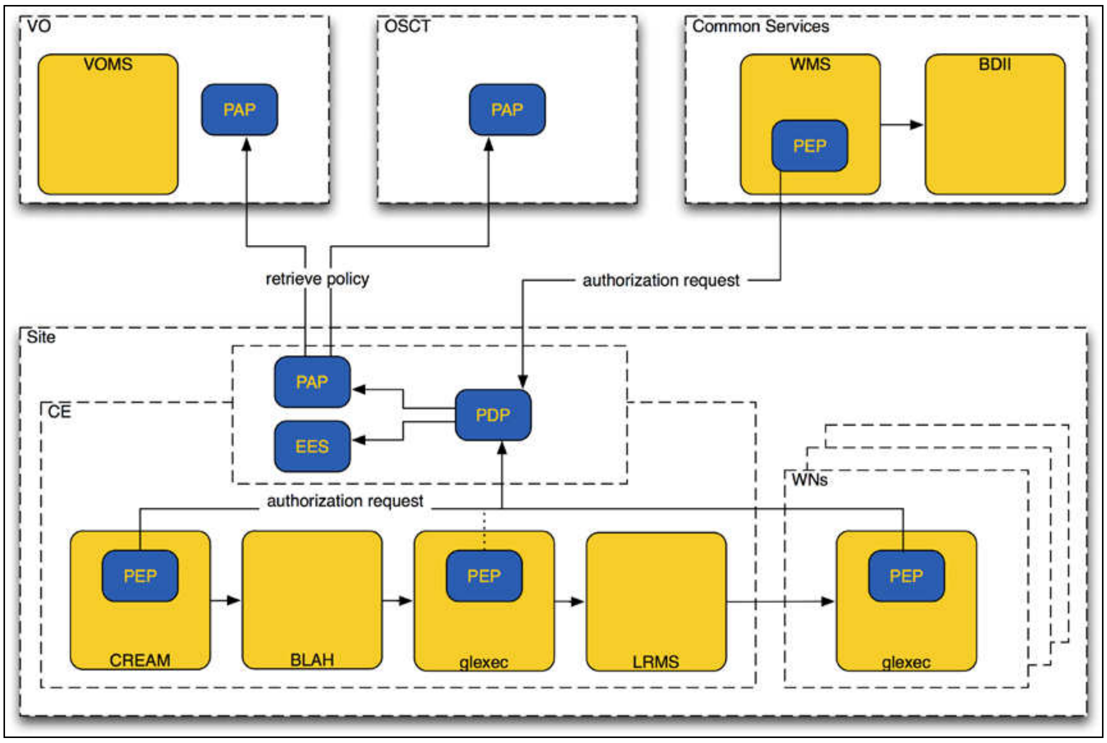

Figure 9: The Argus Architecture

The effective policy for a given authorization decision is the composition of a site's local policy as well as any remote policies. The site has final control over which remote policies are pulled and how those policies affect the overall policy decision. For example, a site may indicate that if any policy, local or remote, indicates that the request is allowed than the returned result should be "permit". Alternatively a site may indicate that its policy should overrule remote policies, or vice versa.

Lastly, the deployment model depicted in *Figure* 9 is likely to be the model used in most cases. However, deploying the PAP, PDP, and EES components is not without some amount of work. An organization may choose to use an alternative deployment model if they wish. PAP, PDP, and EES components may be hosted and run by a third party. Therefore an organization may outsource any, or all, of these components as a means of reducing the complexity at their site at a cost of increased request/response latency.

5.5. IDENTITY SWITCHING ON THE WORKER NODES

Users in the EMI system are identified with globally unique names, either derived from their X.509 credential subject name or other subject attributes. However, many local services, in particular all computing services offered on Unix [5] and Unix-like platforms, and some data storage systems that employ posix style file systems, use a different native representation of the user and group concepts. In the Unix domain, these are expressed as (numeric) identifiers, where each user is assigned a user identifier (uid) and one or more group identifiers (gid). At any one time, a single gid will be the 'primary' gid (pgid) of a particular process. The pgid will be used as the gid under which any new files are created, and for group-level accounting for processes and in batch systems. The uid and gid representation is local to each administrative domain.

In storage systems, back-ends have implemented either a static mapping (largely due to the fine-grained control policies associated with data) or perform access control based on the external attributes. On the Unix job execution side, the need for a system-local credential cannot be evaded, since the isolation and security properties of Unix rely on unique and distinct uids and gids. Unless distinct credentials are used, jobs running within the same operating system environment cannot be isolated using portable cross-platform mechanisms.

In the most straight-forward case, a request to execute a Grid job at a given site is mapped directly at by the computing service to a local identity, under which the job will be executed on the WN as scheduled by the local resource management system. The authorized service can return a local identity (using the EES or equivalent local mechanisms) and if the compute service runs with sufficient privileges in the execution domain, the resulting job process can be launched with the mapped user credentials. In this scenario, once the local identity has been determined, it would not be changed, until the requested job has ended.

There are however two cases where an additional component is needed to perform the identity switch: in case the computing service does run in a context with the appropriate system privileges (e.g. it is run in a Java virtual machine, or as an otherwise non-privileged user; and in case the identity switch is not performed on the service end, but elsewhere in the business chain of the job.

gLite and ARC support the execution of *pilot jobs,* a special kind of job that is submitted to a computing resource but which acts as a 'container' placeholder to retrieve other workload, potentially owned by a different entity. Once this *pilot job* starts to run on a worker node, it pulls one or more jobs from a dedicated service outside the control of the site's local compute service and executes them one after the other. The advantage of the pilot job concept is that it allows selecting the next job at the time, when a slot on a site has become available. In other words, the user's job will not be assigned to a given site early and risks to remain waiting for a long time until a slot has become available in the Local Resource Management System (LRMS). Instead, the user's job is waiting in a special scheduler, which takes the scheduling decision at the time a slot on the Grid is available. They are called *late binding*, in contrast to *early binding*.

Pilot jobs offer the option of either running the user's job, called payload job, under the identity of the pilot job, or under the identity corresponding to the payload job's proxy certificate. The latter option requires that the identity must be switched on the WN.

Both use cases are supported by a single component: gLExec [R20]. gLExec is a program to make the required mapping between the grid world and the Unix notion of users and groups, and has the capacity to enforce that mapping by resetting the uid and gids of running processes. For a service running under a 'generic' uid, such as a web services container, it provides the way to escape from this container uid. It may be used in a similar by externally managed services run on a site's edge. Lastly, in a late-binding scenario, the identity of the workload owner can be set at the instance the job starts executing.

The actual mapping of external credentials and attributes to a system-local credential is performed by gLExec using the LCMAPS credential mapping service. LCMAPS can make a local mapping decision, in large deployments; it may be advantageous to use a single, centralized point where all policy and credential mapping state is kept. They especially useful in case multiple end-points are used for like services, or when the compute and Grid storage services share a common set of POSIX-style file systems that rely on uid and gid access controls, e.g. the Network File System NFS. Similarly, this mapping consistency is relevant for the 'gLExec on Worker Node' scenario, where consistency in uids and gids on the worker nodes and on the batch server is advantageous, but keeping consistency in mapping amongst the worker nodes is necessarily a central function.

The LCAS and LCMAPS services, as well as the PEP client employed via LCMAPS, are common elements are incorporated in various tools. Through the PEP call-out, the full flexibility of the Argus authorization service is available, and gLExec passes all incoming attributes to the authorization service through the PEP client. Services such as the EES (described above) then can make central mappings that are communicated back to gLExec through the LCMAPS PEP client. Similar clients are available for authorization services based on the SAML2XACML2 standard interface, such as those profiled by the collaboration of EGEE, Open Science Grid VO Privilege project and Globus.

Optionally, some rudimentary authorization and access control can be done through the LCAS system, but using VOMS FQAN and identity subject attributes from the pushed-credentials only, but delegating such decisions to the authorization service via the LCMAPS PEP client allows more flexibility and better integration with other services.

6. DATA MANAGEMENT

6.1. UNENCRYPTED DATA STORAGE

Several data storage systems that store the data unencrypted are supported by EMI - the Disk Pool Manager (DPM), dCache and StoRM.

6.1.1 Disk Pool Manager (DPM) and LHC File Catalogue (LFC)

All the files managed by DPM on a given installation are owned by the same Unix account, which serves as a management account. Every DPM instance contains a name service (a local database), where the actual file name, ownership and access information resides. The services exposed by DPM consult this name service for obtaining authorization information. Internally to DPM, every user and group is represented in this local database by a virtual UID and virtual GID, which is independent of any Unix UID or GID for the hosting operating system. These virtual UID/GIDs are just representations of individual users and groups of users.

The authorization mechanisms in DPM are based on how DPM maps the user credentials (proxy certificate with or without VOMS AC) to the virtual UID and GIDs. The mechanism is depends whether the proxy certificates contains a VOMS AC or not. In both cases the virtual UID is derived from the DN and the virtual GID either from the FQANs or a data management specific map file. LFC, a file catalogue, is using the same model, whereas its virtual UID and GIDs are not correlated to the values used by DPM. More information on the DPM security model can be found in [R14].

6.1.2 The dCache Security Model (gPlazma)

dCache is using the second version of the “grid-aware PLuggable AuthoriZation Management” system (gPlazma2) for authentication, credential mapping and related operations. Whenever a request enters the system, gPlazma performs 4 steps, which are 'authentication', 'mapping', 'account' and 'session'. The configuration is similar to the UNIX PAM (Pluggable Authentication Modules) Subsystem allowing particular steps to be declared optional, mandatory etc. The pluggable mechanism allows dCache authentication to be customized to the requirements of the security infrastructure of sites running dCache. gPlazma modules can be provided or modified without deep knowledge of dCache and can be applied to a dCache installation 'in vivo'. At the time being, EMI dCache releases are shipped with gPlazma modules supporting Kerberos, X.509 Certificates, X.509 Proxies and User/Password authentication, ARGUS blacklisting and LDAP (ActiveDirectory)/NIS and GridMap file user mappings as well as GUMS, the Open Science Grid User Management Service.

6.1.3 StoRM Security Model

The Storage Resource Manager (StoRM) exposes a SRM Web Service interface as the only user interface. This interface is exposed by the StoRM “FrontEnd” component, a service that may be deployed on a host that does not access the SE storage.

When a user invokes the SRM Web Service they must provide a X.509 Proxy certificate with VOMS extensions, afterwards the communication between the endpoints is performed via http protocol over GSI. The first authorization step is performed in the FrontEnd where the X.509 proxy and VOMS FQANs are validated. Subsequently the user DN identified in the X.509 proxy is checked for blacklisting via Argus. The Argus blacklisting verification can be disabled by the site administrator.

This pre-validated request is then sent to StoRM “BackEnd” component, a service that resides on a host with full access to the SE storage. The communication between FrontEnd and BackEnd is performed via XML-RPC over HTTP. The StoRM BackEnd is configured to receive such calls only from the IP address of the FrontEnd host. All the SE storage files managed by StoRM are owned by the Unix account of the user running the StoRM BackEnd service. StoRM considers all its files as residing in a certain Storage Area.

When the BackEnd receives a request for interactions with a file first of all it verifies that the user performing the request is allowed to approach the Storage Area hosting the request by inspecting their DN, VO and role. Upon validation, the user is mapped to a local Unix user using the information contained in the GridMap file and LCMAPS library. If a valid local user is found the request has to pass a further authorization step: the requested SRM operation is mapped to a file system-like operation and the authorization for the local user to perform this operation on the requested file is verified using the information stored in the Path Authorization file. The definition of the policies stored in this file is up to the site administrator.

When all the authorization steps are passed an extended ACL is stored on the requested file for the requested operation to the local Unix user in order to allow users to access StoRM resources via the file:// protocol. Depending on the configuration of the Storage Area this information can be transient (with a well-defined lifetime) or permanent.

6.1.4 The File Transfer Service (FTS)

The File Transfer Service (FTS) is a service that moves data from one SE to another over a defined channel between the sites. FTS operates on individual files as well as sets of files. It has dedicated interfaces for managing the network resources and to display statistics about transfers. Optionally, it also supports lookup and registration of files in catalogues. The authorization is based on defined roles, to which the user is mapped using their DN. Mapping based on FQAN will be supported in the future.

FTS renews expiring proxy certificates through the use of MyProxy, similar to the renewal mechanism as described in Section 3.8). More information on the FTS security model can be found in [R14].

6.2. ENCRYPTED DATA STORAGE

The encrypted data service solution provides a system that encrypts/decrypts files and stores/retrieves them from GFAL library-compliant storage systems. For additional security and redundancy, the encryption keys used to encrypt the files are split and stored in separate keystores (Hydra servers).

Hydra [R19] is the central element in the encrypted file storage solution. The Hydra service comprises at least one Hydra server and a set of command-line interfaces to perform the basic commands. The Hydra server consists of a MySQL database and a Tomcat server that acts to contain the actual Hydra software, written in Java. Communication between the Hydra services and clients is through SOAP.

The encrypted data service solution uses the Hydra servers to store the pieces of the encryption key and their associated Access Control List (ACL) information. Therefore the key pieces are only accessible by their respective owners. These encryption keys are split using a particular scheme, Shamir Secret-Sharing Scheme (SSSS) that provides another layer of protection. The encryption key may be split into "M" fragments and may be recovered with possession of "N" fragments, where N<M. These parameters "M" and "N" are configurable by the number of Hydra servers deployed and the SSSS algorithm respectively. In this case, the security and reliability of the overall service has been increased: an attacker would have to gain access to at least "N" Hydra keystores to recover and encryption key; a legitimate user can tolerate the loss of M-N keystores and still recover their key. In addition, a legitimate administrator of any single Hydra keystore would not have access to the full encryption key.

Encryption keys are generated in the Hydra system based upon the concept of a globally unique identity (GUID) of the file in question to be encrypted. This GUID can be retrieved from sources such as an LCG File Catalog (LFC) that contains the logical names for files inside a Disk Pool Manager (DPM) storage element. The GUID is necessary for retrieval and decryption of files.

7. LOGGING, TRACING AND AUDITING

Being able to trace and audit past actions is a prerequisite for operating an infrastructure in a secure and stable manner. Although the primary focus here is on the security aspects, traceability and logging is a prominent element of all software since it is the key element for tracing stability issues, finding bugs, and generally maintaining an operational service by system managers. As such, the (security) logging capabilities of the software are continuously exercised not for security purposes but for operational reasons.

However this is only possible if consistent and complete logging information is produced by all services. The EGI Security Policy group, leveraging work of the joint EGEE/OSG/WLCG security policy group (JSPG), have described the criteria for adequate logging: "The minimum level of traceability for Grid usage is to be able to identify the source of all actions (executables, file transfers, pilot jobs, portal jobs, etc.) and the individual who initiated them. […] The aim is to be able to answer the basic questions who, what, where, and when concerning any incident. This requires retaining all relevant information, including timestamps and the digital identity of the user, sufficient to identify, for each service instance, and for every security event including at least the following: connect, authenticate, authorize (including identity changes) and disconnect."[R21].

Especially is a distributed infrastructure, the 'hand-over' of work flows between services, maintaining forward (and where possible) backward references to each services log data, is important. To implement requirements such as those from the EGI SPG, the EGEE-II Middleware Security Group MWSG, in collaboration with its global partners, defined the "Middleware Security Audit Logging Guidelines" [R22].

Although the document today is not followed to the letter with regards to format, the spirit of the document has permeated all EMI services, and almost all of the relevant information today is logged using site-central mechanisms like syslog, and forward identifiers are kept. Traceability for job execution services has been assessed in operational scenarios and exercises run by EGI in 'security service challenges' [R23] and workflows can be followed across multiple service end points. The logging of storage services and correlation of storage and compute actions remains largely based on correlating identities and time stamps. There can be also be site-specific auditing mechanisms; since the end-user supplied code it used initiated elements of the workflow. Sites availing themselves of operating-system based auditing (e.g. for those OSes targeting Common Criteria compliance) have more correlation possibilities using those mechanisms.

Apart from operational feed-back there is no formal monitoring of compliance of code with the audit guidelines, nor does EMI have itself a single set of mandatory audit formats.

8. SECURITY MANAGEMENT AND THREATS HANDLING

The software (security and otherwise) provided by the EMI project is, of course, released for use by various Grid users or clients e.g. EGI and WLCG. As issues (vulnerabilities or bugs) are identified they are assessed and prioritized in a joint process between clients and EMI. These then arrive to the EMI developers as bugs to be fixed. Concurrently, the EMI project has a more formal program of evaluation of software components for security flaws.

8.1. SOFTWARE SECURITY MANAGEMENT

The project has a formal process for handing security vulnerabilities identified in the software. It has both a pro-active and a re-active method for dealing with potential vulnerabilities.

For the pro-active approach, EMI software which performs key functions that potentially affect security are identified and analyzed through a first principles vulnerability analysis [Kupsch, Miller, Heyman, Cesar, "First Principles Vulnerability Analysis" [R24]. Although by nature a laborious process, the first principles analysis is better than automated tools in finding *exploitable* vulnerabilities. This analysis has been done for selected EMI software products that are security sensitive, such as gLExec and VOMS. Vulnerabilities found have been addressed and mitigated, and disclosed using the industry-standard CVE process.

Reactive vulnerability management has been formalised with a set of defined response and resolution times, and agreed with major stakeholders such as EGI. Here resolution times have been agreed; with for example a 48 hours resolution time after reporting for extremely critical issues (resolution can either be a software fix, a configuration change, or a patch to deployed installations). Other low-priority vulnerabilities are either distributed as part of the regular software release cycle or made available as updates.

The risk ranking is not done solely by EMI, but through the Risk Assessment Team (RAT) jointly set up by EGI and the major European middleware providers and customers, including EMI, IGE, and WLCG and selected outside experts, under the aegis of EGI.eu. The RAT is responsible for ranking the risk of each vulnerability, taking into account the operational environment, according to the EGI Software Vulnerability Issue Handling Procedure [R25]. The RAT is managed by the EGI Software Vulnerability Group (SVG) [R26] with oversight from EGI, EMI, IGE and others.

Operational security issues and vulnerabilities which have already been disclosed are handled by the EGI CSIRT Incident Response Task Force [R27] who assesses the risk and impact of the issue. Here the RAT and EMI as a software provider are involved as experts, but have a more reactive role than in vulnerabilities that are responsibly disclosed and managed. However, the same service levels and response times apply.

8.2. BUG FIXES, EMERGENCY RELEASES

The EMI middleware stack is released to infrastructure projects such as WLCG and EGI. These projects, during the course of running the EMI stack, discover issues (bugs). These bugs are reported through a standard ticket processing management system, Global Grid User Support (GGUS) run by EGI. The bug reports are processed and the EMI developers are in the role as third-line support and receive bugs that cannot be closed by EGI. Subsequently, a bug is opened in the product team bug tracker and the issue is solved. These interactions are documented in more detail at [R51].

Software releases are regularly scheduled and are documented; see Annex EMI Maintenance and Release Processes. If the EGI RAT (see Section 8.1) assesses a security vulnerability as critical, then an emergency release can be made. This procedure is not limited to security software and is handled by the EMI SA1 work package.

8.3. GRID SERVICES SECURITY ASSESSMENT

Within the EMI project, effort has been devoted to perform a software and functional security analysis on various EMI components – these are independent software security experts who are not directly part of any development team. Inclusion of these experts in the EMI project is seen as the most efficient way to gain access to objective software security analyses.

These assessments followed the First Principles Vulnerability Assessment methodology (FPVA), which is a manual (analyst-centric) methodology aimed at finding critical vulnerabilities first. Critical vulnerabilities are those affecting high value assets. Given that FPVA is a manual methodology, it is slow and therefore expensive.

Ideally, vulnerability assessments should be performed on all components of the different software stacks within EMI. Unfortunately this is not realistic given the resources granted for this activity. So far in the EMI project the following EMI components from the gLite stack were assessed.

Component	Version	Code Base	Effort	Result
Argus	1.2	42k lines Java/C	5 months	0 vulnerabilities found.
gLExec	0.8	48k lines C	4 months	5 vulnerabilities found.
VOMS-Admin	2.0.18	140k lines Java, JSP, Javascript	2.4 months	4 vulnerabilities found
VOMS core	1.9.19-2	161k lines C/C++ exp and Java	6 months	1 vulnerability found
WMS				In progress...

Table 3: gLite Vulnerability Assessment

9. INTERNATIONAL COLLABORATIONS

9.1. OGF, IGTF, IGE

With security being a foundational layer of interoperability, the security area in Open Grid Forum has several working groups in which EMI participates. In the software standard activities EMI has a leading roles as co-chair of all security working groups (IDEL, VOMS-PROC and formerly OGSA-AuthZ), and EMI also is closely involved with the operational groups (CAOPS-WG and IGTF).

The working group on identity delegation (IDEL-WG) was fostered by and is co-chaired by EMI, with participation from IGE, XSEDE, OSG and the UK NES. This aligns with the EMI objective to revert to the use of a non-modified TLS protocol instead of GFD.078 [R31]. IDEL will consider out-of-band delegation protocols, based on SOAP and REST protocols, to propagate delegated credential to services in the workflow, with emphasis on identity delegation.

The format and syntax of the VOMS FQANs was documented through the OGSA-AuthZ-WG in GFD.182 [R32], but since this document concerns only the syntax and only addresses the interpretation of the attributes in a limited scope, the need for a comprehensive document on the ordering, interpretation and combination of VOMS attributes in both the PKI and SAML domains was needed. The VOMS-PROC WG, initiated by IGE in collaboration with EMI and XSEDE, addresses this issue through two documents: “VOMS Attribute Certificate Parsing Rules for Chained Identity Credentials” describing the combination of attributes through a single identity delegation chain, and “Understanding parsing rules for collated VOMS SAML space” which addresses the complementary elements when expressed using SAML. Both documents are currently in draft.

EMI also participates in the CAOPS-WG, addressing the technical foundation of trust in the identity management space. The CAOPS group, which liaises closely with the International Grid Trust Federation IGTF, defines not only the technical profiles of PKIX X.509 for use in the Grid, but also drives the move towards secure hash algorithms for certificates and the development of profiles for on-line credential status checking using (light-weight RFC 5019-style) OCSP, setting profiles for the credential-issuing authorities and proposing guidelines for the use of OCSP in clients. EMI, in collaboration with IGE and many national e-Infrastructure projects, contributes to the development of the profiles, which are closely aligned with industry – since both the web browser and CA industry and grid face the same problems at the same time.

10. ASSESSMENT, STRENGTHS, IDEAS FOR IMPROVEMENT

The Security architectures of the three middleware stacks have been devised and implemented in previous projects. These architectures reflect the requirements of the infrastructure operators and their users. As the architecture of ARC/gLite has been an iterative and evolutionary process over several previous projects and over the years there have been many components that have been created and abandoned or stopped. At this current moment the Grid infrastructure (EGI and WLCG) runs in continuous operation using X.509 certificates for Authentication and Authorization.

The EMI security stacks consist of components of ARC, gLite and UNICORE bound together loosely with some common services and libraries. The common efforts include: The Common Authentication libraries; Common Authorization Service (Argus); Common Attribute Authority (VOMS); Common Security Token Service (STS); Common delegation method (to transfer X.509 proxies).

10.1. STRENGTHS

There has never, to this date, been a security incident on the Grid due to a fault in the middleware (gLite or ARC) security components. All incidents have been traced back to the operating system vulnerabilities or well-known attack techniques e.g. phishing. UNICORE also can report the same status. The general strengths of the EMI stack are:

- The common security credential is based on solid industry-standard PKI technology (X.509), which is widely used in many critical systems such as in banks and large e-commerce systems. It also uses the current versions of standard libraries such as OpenSSL, BouncyCastle.
- The ability to support Grid deployments, i.e. the deployments where there is a minimal trust between participating sites. This is realized by the means of trust delegation (with proxies or ETD) and makes the architecture different from vast majority of solutions used in other distributed systems. (move up to second point).
- The deployment is highly scalable as the X.509 proxy functions as the AuthN and AuthZ token carrying all the necessary attributes. It is not necessary for all services to contact each other to form a web of trust.
- Common Authentication libraries available throughout the stack and also for external usage. Provides common error messages and AuthN responses.
- Separated AuthN and AuthZ in principle. Argus is not bound to specific AuthN mechanism and can use different technologies, e.g. SAML and UNICORE or even http.
- Convergence on a standard X.509 delegation interface, gridsite. This provides a standard method to pass X.509 proxies between services.
- Able to guarantee a traceable security chain from the user to the computing worker node. This is also true for multi-user pilot jobs when gLExec on the WN is deployed and correctly configured and used.
- Support for federated identities to access Grid services through the use of the STS and accredited CA.

10.2. WEAKNESSES

The process of X.509 proxy delegation results in the impersonation of the original user (submitter to Grid). In this process no information on the trust chain of the intermediate services that the proxy has passed through is included in the proxy credential. Therefore it is impossible for services further down the line to determine whether they can trust the credential and the actions it is requesting. Currently, this trust is assumed not expressed.

Any user credential must be renewed in case the tasks assigned within a Grid exceed the time limit of the credential validity. In the case of X.509 proxies the recommended validity is 24 hours. There order is to limit the amount of damage that can be caused in case a proxy is stolen. In general, if tasks exceed the 24 hour limit then the proxy must be renewed. This renewal process is unwieldy and is achieved using several components. These are not issues for the UNICORE middleware that does not use proxy certificates but the Explicit Trust Delegation solves the shortcomings of proxies.

An overall weakness can be postulated that, in general, the delegation of credentials between services and the renewal of credentials is an unwieldy procedure. These services include the CEs, Data storage and especially the centralized workload management system (WMS). This concentrates too much credential handling and renewal activity in one service. Included is the delegation of credentials from the UI to MyProxy servers. The delegation step itself is a weakness as currently all the attributes (and therefore rights) are passed in the step. Delegation of a reduced set of attributes is not possible.

Already, the WLCG experiments have shown that a WMS is not necessary for large-scale physics data processing (one of the main use-cases of the Grid). The large physics experiments have moved to a pilot-job service model that use pilot job frameworks existing outside the Grid middleware. Here a long-lived job, that uses a long-lived pilot credential, is injected into the Grid job management chain. Upon final arrival at a worker node this job signals the pilot service to start to send the analysis payloads. These analysis payloads are then executed with the credential of the submitter to the pilot service.

EMI security authentication uses PKI also for identifying end-users. The advantages of this approach are clear and were numerously mentioned in the document. However usage of X.509 certificates is troublesome for end-users:

- Most of end-users do not understand the PKI and therefore handling of certificates and private keys is a big concern. This may lead to security incidents with private keys left unencrypted or encrypted with a weak password on a publicly available machine.
- Handling of certificates and private keys is a big issue for roaming users, users using untrusted computers, using several trusted computers and having several identities (i.e. certificates issued by different CAs).

As the authentication (and in case of gLite and ARC also delegation) is tightly coupled with X.509 certificates, it is difficult to create web interfaces to the EMI middleware. Even with the help of STS the integration is very complicated. What is more it is currently impossible to apply even stronger authentication mechanisms as Multi Factor Authentication.

10.3. IMPROVEMENTS

Direct submission of pilot jobs to the site computing elements (CEs), rather than WMS, would eliminate the need for one stage of proxy handling (delegation) and renewal. This is the case presently for the CERN LHC experiments but is not generally the case for other EGI Grid communities.

From the security point of view, if a site detects a malicious payload or activity, then it may either:

- Ban the pilot credential, therefore protecting itself against further malicious payloads from the pilot service.
- Ban the payload credential, allowing other pilot-submitted payloads to continue.

By eliminating the WMS step, the site that runs the CE can set its own policy on credential lifetime and banning policy.

The overall ease of the middleware could be improved by full integration of the CANl. This would give standard error messages to users and administrators and allow standard analysis and monitoring tools to be built. Also, with the presence of STS, integration to various portals should be emphasized.

Security assessments of the EMI stack should be planned as part of the release process. Some NGIs are already preforming this work on code that will run in their infrastructures. Code reviews were performed during the EMI project and this work highlighted the labor-intensive nature of this work. A more uniform migration of the security code to public repositories such as github would make it easier for developers to contribute back-patches and examine the code and documentation.

As mentioned above the possibility of alternatives to the delegation step should be explored. Possibly by reducing the use of delegations to only where it is strictly needed. Another path could be SAML adoption within the EMI stack, not just at the interface with the user. This would move the trust from the user credential to the service credential. This proposition would require a large amount of work and a strong request from the user communities to justify such an improvement in the security of Grid infrastructures. The attractions of SAML and ETD are obvious but the reality of maintaining a production Grid infrastructure may rule this out. Another effect that is not well-understood is how scalable this option would be. The main Grid users are not asking for this feature and probably would not accept the disruption.

If the above proposal is too onerous then the possibility of delegating only some of the users rights could be explored. Currently it is too complicated for the majority of the Grid users (in gLite/ARC and UNICORE) to determine which attributes (rights) should be selected for the credential to be delegated. The only other option is to automate this selection. Therefore the attributes need to be selected at run-time. Today, this task has not been seriously considered as it has not been requested by users. The perceived difficulty of this task coupled with the lack of user requirements makes the potential return on investment seem low.

The UNICORE plans for improving their security architecture are focused on usability and integration with existing federations. To achieve this it is planned to support a fully certificate-less access to the Grid, what will eliminate all certificate related problems which are faced by end-users, especially in web access case.

The fundamental problem behind certificate less access, where end-user is unable to digitally sign data is related to trust delegation. Therefore the main focus is placed on extension of UNICORE Explicit Trust Delegation model, so it can be used without the end-user signing the initial trust delegation as it is now.

By definition, the security of commercial clouds is not addressed by the EMI security stack. whereas the very act of deploying services and data to resources that are essentially unknown and outside of the policies developed by Grid projects and VOs renders any services untrusted. The components and services of the EMI Security stacks of course technically can be deployed on virtual machines within a commercial "Cloud" infrastructure. Deployment on a "Private Cloud" i.e. a cluster controlled by virtualization technologies is no different to a deployment on a traditional cluster and would enjoy the same level of guarantees from Grid project and VO policies.

The delegation services could be deployed within a couple of commercial clouds in order to provide a loose federation of their infrastructures. The Argus AuthZ service could be deployed on a Cloud node in order to control access to other nodes that a client may have rented. A Cloud client could therefore shift their data processing or storage based on various metrics.

Finally, any improvements to the Security stacks result from requirements of the operators and users of the Grid infrastructures. Experience and requirements have shown that any improvements or changes must not impede or disrupt operations. Therefore the two scenarios for changes are that they are incremental or a complete radical change of direction. Any future projects, large or small, should be driven by the user requirements and be controlled by the user communities.

REFERENCES

R1	Global Security Architecture (EGEE-I EU Deliverable DJRA3.1) https://edms.cern.ch/document/487004/
R2	Global Security Architecture, rev1 (EGEE-I EU Deliverable DJRA3.3) https://edms.cern.ch/document/602183/1.3
R3	Security Architecture Assessment (EGEE-I EU Deliverable DJRA3.4) https://edms.cern.ch/document/686044/
R4	Shibboleth Interoperability through dedicated SICS (EGEE-II EU Deliverable MJRA1.4) https://edms.cern.ch/document/770102/1
R5	Shibboleth Interoperability with Attribute Retrieval through VOMS (EGEE-II EU Deliverable MJRA1.5) https://edms.cern.ch/document/807849/1
R6	Grid Components Re-engineering Workplan (EGEE-II EU Deliverable MJRA1.3) https://edms.cern.ch/document/756544/2
R7	R. Housley et al. RFC3280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. http://www.ietf.org/rfc/rfc3280.txt
R8	S. Tuecke et al. RFC3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. http://www.ietf.org/rfc/rfc3820.txt.
R9	J. Hahkala, H.Mikkonen, M.Silander, J.White Requirements and Initial Design of a Grid Pseudonymity System; Proceedings of the 2008 High Performance Computing and Simulation Conference (HPCS 2008), Nicosia, Cyprus, 3-6 June 2008.
R10	Authentication and Authorization Infrastructure (AAI) in a nutshell. http://switch.ch/aai/support/documents/#flyer
R11	M. Myers et al. RFC2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol (OCSP). http://www.ietf.org/rfc/rfc2560.txt
R12	Alfieri R. et al. VOMS, an Authorization System for Virtual Organizations. In *Grid Computing, First European Across Grids Conference*, 2004
R13	S. Farrell and R. Housley. RFC3281: An Internet Attribute Certificate Profile for Authorization. http://www.ietf.org/rfc/rfc3281.txt.
R14	Overview of gLite authorization mechanisms. EGEE-II MJRA1.7 https://edms.cern.ch/document/887174/1
R15	Globus Workspace project: http://workspace.globus.org/
R16	Grid Security Tracing and Logging Policy https://edms.cern.ch/document/428037
R17	MyProxy Credential Management Service: http://grid.ncsa.uiuc.edu/myproxy/
R18	Privilege Project: http://www.fnal.gov/docs/products/voprivilege/
R19	Hydra: https://twiki.cern.ch/twiki/bin/view/EMI/EMIHydraDocumentation
R20	gLExec: D Groep, O Koeroo and G Venekamp, "gLExec: gluing grid computing to the Unix world", *J. Phys.: Conf. Ser.* **119** 062032.
R21	JSPG: Kelsey et al., https://documents.egi.eu/document/81]
R22	MWSG: Groep et al., https://edms.cern.ch/document/793208/1].
R23	EGI: https://wiki.egi.eu/wiki/EGI_CSIRT:Security_challenges]
R24	"First principles vulnerability assessment", doi>10.1145/1866835.1866852
R25	https://documents.egi.eu/document/717
R26	https://wiki.egi.eu/wiki/SVG

R27	https://wiki.egi.eu/wiki/EGI_CSIRT:IRTF
R28	MJRA1.10 - EMI Security Workshop: http://cds.cern.ch/record/1277569?ln=en
R29	http://cdsweb.cern.ch/record/1277567?ln=en
R30	Transport Layer Security (TLS) Extensions, http://www.ietf.org/rfc/rfc4366.txt
R31	Von Welch, Grid Security Infrastructure Message Specification, http://www.gridforum.org/documents/GFD.78.pdf
R32	V. Ciaschini, V. Venturi, A. Ceccanti, The VOMS Attribute Certificate Format, http://www.gridforum.org/documents/GFD.182.pdf
R33	gLite: https://edms.cern.ch/document/935451/2
R34	UNICORE: http://unicore-dev.zam.kfa-juelich.de/release-candidates/docs/security/output/sec-arch.html
R35	ARC: http://www.nordugrid.org/documents/arc-security-documentation.pdf
R36	XACML standard: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf
R37	XCAML profile: https://cds.cern.ch/record/1449146?ln=en
R38	SAML profile. https://cds.cern.ch/record/1451532?ln=en
R39	EMI delegation document: http://discordia.desy.de/paul/Delegation/20121024-emi-delegation-agreement-v1.1.pdf
R50	DSA1.4.2 Annual Maintenance and Support Report (April 2012) https://cds.cern.ch/record/1277558?ln=en DSA1.4.1 Annual Maintenance and Support Report (April 2011) https://cds.cern.ch/record/1277557?ln=en
R51	DSA1.1 Software Maintenance and Support Plan https://cds.cern.ch/record/1277556?ln=en

ANNEX: EMI MAINTENANCE AND RELEASE PROCESSES

The EMI Maintenance organization follows the guidelines of the ISO/IEC 14764:2006 (Software Engineering - Software Life Cycle Processes - Maintenance) standard, and is documented in the [R50]. The process is briefly summarized here.

In EMI, the maintenance and evolution of the software is implemented inside **Product Teams (PTs).** PTs are small teams of software developers, fully responsible for the successful release of a particular software product (or group of tightly related products) compliant with agreed sets of technical specification and acceptance criteria.

Each PT has a **Product Team Leader (PTL)** who is responsible for coordinating the design, development, support and maintenance in the context of the PT.

The **Release Manager (RM)** is responsible for the overall coordination of the release process. Main duties include steering the release process, planning and managing EMI releases, chairing the EMT ensuring that all relevant issues are being followed.

The **Engineering Management Team (EMT)** is led by the Release Manager and composed of the PTLs (or duly appointed representatives), a QA representative, a Security representative, representatives of the operations teams of the major infrastructures (like EGI and PRACE) and invited experts as necessary. The EMT manages the release process and the approval of “standard” request for changes, that is all the changes that are pre-approved based on established policies and do not need to be individually approved by other project boards (i.e. the Project Technical Board (PTB) or the higher level Project Execution Board (PEB)). This includes software defects triaging and prioritization, technical management of releases, integration issues, and user support request analysis.

Component releases

The EMI release process is centered on four types of releases:

Major Release A major release for a component is characterized by a well-defined interface and behavior, potentially incompatible with the interface or behavior of a previous release. New major releases of a component can be introduced only in a new major release of EMI.

Minor Release A minor release of a component includes significant interface or behaviour changes that are backwards-compatible with those of the corresponding major release. New minor releases of a component can be introduced in an existing major release of EMI.

Revision Releases A revision release of a component includes changes fixing specific defects found in production and represents the typical kind of release of a component during the lifetime of an EMI major release.

Emergency Releases An emergency release of a component includes changes fixing only Immediate-priority defects found in production, typically security-related.

Maintenance and support schedule

EMI major releases will be supported and maintained for two years after the release date. The availability of a new major release of EMI does not automatically obsolete the previous ones and multiple major releases may be supported at the same time according to their negotiated end-of-life policies. It is foreseen, however, that only the latest two EMI major releases are supported at any time. Within an EMI major release only the latest version of a component is supported.

The maintenance schedule is organized in the following periods:

Full maintenance period: during this period updates are released to address issues in the code and provide new features are provided for each supported EMI major release (12 months).

Standard maintenance period: during this period updates are released to address issues in the code, but no new feature is introduced for each supported EMI major release (6 months).

Security updates period: during this period only updates targeting security vulnerabilities are provided for each supported EMI major release (6 months).

End-of-life period: in this period no updates or support is provided any more. The end-of-life period starts after the end of the security updates period.

Specific exceptions to the above maintenance periods can be negotiated between the users and the PTs depending on various criteria that include critical run-time of experiments or projects that do not allow upgrades to new versions, particularly complex deployment and migration conditions, etc.

Problem analysis

Request For Changes (RfCs) are classified according to priority, where the priority is a composition of the factors like severity, impact, urgency and cost. The evaluation of the priority of an RfC results in one of four possible logical values: Immediate, High, Medium and Low. Each level implies a very specific behavior for the management of that RfC. In particular, the behaviours are:

- **Immediate:** the RfC needs to be addressed as soon as possible, in all affected EMI major releases on all supported platforms. A release containing Immediate-priority changes can contain **only** Immediate-priority changes. Multiple Immediate-priority changes can be included in the same release, provided that any change does not delay the release significantly. This constraint minimizes the risk of introducing new defects in the new release of the affected component and allows the adoption of special accelerated procedures for its release. Moreover it will not force site administrators to deploy lower-priority changes during an emergency update.
- **High:** The RfC will be addressed in a next release of the affected component, in all affected EMI major releases on all supported platforms.
- **Medium:** The RfC will be addressed in the release of the affected component that will be shipped with the next EMI major release.
- **Low:** There is no target date for addressing the RfC.

Security vulnerabilities, according to their risk assessed by the EGI-SVG Risk Assessment Team (RAT) and the affected PT leader, are mapped to RfCs tracked in EMI trackers in accordance with the SVG Software Vulnerability Handling Process. Vulnerabilities risk can be categorized as Critical, High, Moderate or Low. A target solution date is defined for each risk when the risk is assessed, and represents the time by which a fix for the vulnerability should be available in production repositories. The target date for the risk categories is defined as follows:

- **Critical:** 3 days
- **High:** 6 weeks
- **Moderate:** 4 months
- **Low:** 1 year

Once the risk has been assessed the EMI RM, in agreement with the affected PT, schedules a release in compliance with the target date. The progress on the vulnerability handling is followed up in a private issue tracker provided by EGI, where all the appropriate parties are contacted and represented.

Once the issue is resolved in the code an advisory is issued by the SVG. This advisory is referenced by the release notes of the affected products so that the disclosure of information regarding the vulnerability is properly controlled.

SVG defines a special procedure for the Critical risk vulnerabilities which aims at involving as quickly as possible all the relevant experts to assess the nature and exploitability of the vulnerability and to produce a patch as quickly as possible or provide other recommendations to contain the vulnerability as effectively as possible. For more details, see the Software Vulnerability Handling Process [R25].

No critical vulnerabilities have been discovered in EMI products during the EMI project.