Genesis II Security Architecture[1]

Duane Merrill and Andrew Grimshaw

Department of Computer Science
University of Virginia

1 Overview

The security of grid infrastructure is its ability to protect its assets (information, data, services, etc) from various sources of misuse (i.e., threat). Following the Open Systems Interconnect threat model [ituOsi], the types of security threat to which networked resources are vulnerable include disclosure or theft of resources, modification (including destruction) of resources, and resource service interruption. The purpose of Grid security architecture is to define the security services and related mechanisms that can be appropriately applied to mitigate such threat. The goal of these services and mechanisms is to make the costs of unauthorized behavior greater than the potential value of doing so. Without a strong commitment to meaningful security, many potential adopters would be unable to participate because of undue risk and/or legal restrictions.

Grid security architecture is complicated by the reality that participants from different administrative domains will "come to the table" with distinct trust and security infrastructures. This presents challenges for interoperability, even if all resource interfaces and security mechanisms are well-specified by de facto community standards. For example, although the service implementations of a particular application (e.g., a job execution service) may conform to the same interface, each provider is free to choose the security mechanisms that satisfy its particular security requirements. The orthogonality between service interface and security has two important implications: security tokens from one domain may not carry any syntactic or semantic meaning within another, and different peers may require different compositions of secure communication mechanisms (e.g., specific encryption or signature actions).

With the paradigm of site-autonomy in mind, OGSA-compliant architectures such as Genesis II are designed with flexible security architecture in order to support the integration of existing trust infrastructures rather than force users to adopt a new, singular security model. Genesis II provides services and mechanisms to federate and broker existing identity, to provide suitable forms of new identity as necessary, and to configure and discover the secure communication requirements of communicating peers. This is fundamental to realizing the vision of resource sharing in an environment lacking a single source of authority.

Although Genesis II is implementation agnostic and uses standard mechanisms, it guarantees neither the complete interoperability nor the security of all participants. By design, it does not establish a "lowest common denominator" set of security mechanisms that must be supported by all compliant resources, nor does it govern the establishment and maintenance of the trust relationships that are ultimately necessary to provide assurances of authorized usage. Domain administrators are responsible for undertaking threat assessment for new Grid resources and then using these requirements to select appropriate security mechanisms. Administrators must also negotiate trust with their counterparts in other domains in order to configure token brokering services to federate their security infrastructures.

In the following sections, we present the Genesis II models for identity, identity brokering, access control, and secure communication.

[1] Technical Report CS2009-07, Department of Computer Science, University of Virginia. January 2009.

2 Identity

Identity is one of the most important issues in Grid computing. It is necessary for determining who the remote party is (authentication), who is allowed to perform what actions (authorization), and who did what and when (auditing). Genesis II supports a number of identity standards that are defined for Web Services. These currently include X.509 digital certificate [ituX509, wsX509-1.1], UsernameToken [wsUT-1.1], and SAML token [saml-2.0, wsSAML-1.1] profiles.

When considering the client-server model of Web services, we often think in terms of *resource* identities and *user-principal* identities. We associate resource identities with virtualized Grid resources (i.e., callees) and user-principal identities with actors (i.e., callers). Although there are different challenges for managing resource identities versus user-principal identities, such classification of identity is not mutually exclusive: Genesis II resources may actively call other resources.

2.1 Resource Identity

A Genesis II resource is a logical entity to which messages can be delivered. In many cases, such entities are local resources (e.g., files, database tables, job queues, etc.) that have been "provisioned" into the Grid. Unfortunately, classic resource identities (e.g., file i-nodes, RDMS table names, PBS queue URLs, etc.) are often unsuitable for Grid use because they are not guaranteed to be universally unique and/or they are not cryptographically authenticatable.

All Genesis II Grid resources are given X.509 identities, each consisting of an X.509 digital certificate and a corresponding public/private keypair. These cryptographic identities can be used by clients to ensure that they are indeed communicating with the intended Grid resource (as opposed to being spoofed).

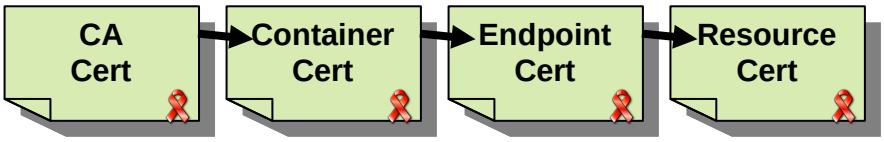

Figure 1: X.509 certificate chain of trust for Genesis II resource identities

In many cases, Grid resources and their identities are automatically generated and destroyed at sufficient rates and quantities as to preclude human involvement in the trust process of vetting identity. The depth of the certificate chain for automatically generated resource identities reflects this dilution of trust. A typical certificate chain of trust for a Genesis II resource has four entries:

1. The automatically generated end-entity certificate identifying the logical resource
2. The automatically generated certificate of its parent Web Service instance (e.g., the Web Service instance for RNS, ByteIO, BES, etc.), which is also a Genesis II resource
3. The certificate of the hosting Genesis II Container (which is also used for HTTPS transport connections), configured by the domain administrator
4. A global Certificate Authority (CA) that conspiring domain administrators configure to be "trusted" by all Grid participants.

2.2 User-principal Identity

User-principal tokens are security credentials that state different facts about the caller. For example, one security credential may claim that the caller is *John Smith* as vetted by State University, and another that indicates they are a member of the Astronomy Department. When using Genesis II, a caller can convey arbitrarily many identity tokens.

User-principal tokens are used by Genesis II resources to perform access-control. When performing an operation on a Genesis II resource, the user-principal credentials conveyed within the request message are checked against that resource's security policy to decide whether to allow or deny the operation.

2.2.1 Existing Identities

Genesis II is very flexible with regard to user-principal tokens. In many cases, Grid participants will already possess identities supplied by their organization. The Genesis II login commands make it simple to acquire these credentials for Grid use, and security tools allow for the inclusion of these credentials within resource authorization policies.

For example, users can use the login tool to acquire an existing X.509 credential stored within the Windows keystore or other popular keystore formats (e.g., PKCS12, JKS, etc.). The corresponding X.509 digital certificate is their identity and is conveyed within outgoing messages; the private key is used for signature and never leaves the calling process's address space. The login command also allows users to convey UsernameToken credentials within outgoing messages.

Alternatively, users may have identities that are managed by directory systems such as NIS/YP, LDAP, etc. Genesis II can integrate with these systems to virtualize these identities into the Grid: Genesis II login commands can authenticate to these services, and their identity entries can be specified within resource authorization policies.

2.2.2 Delegated Identities

Although Genesis II allows clients to use a specific X.509 identity for message/transport-level authentication, the ability to delegate identities/roles to a single shorter-lived, transient identity has several advantages. Delegation allows users to obtain their roles/identities when and where they are needed without exposing the associated private keys to the Grid client. Delegation also allows clients to convey multiple identities/capabilities that have been cryptographically delegated to the single X.509 certificate/keypair used for authentication in message/transport-level protocols.

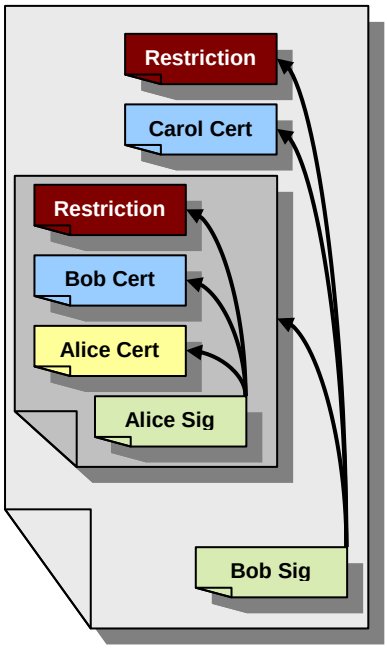

Figure 2: Nested SAML assertions used for cryptographic identity delegation: "*Alice says Bob can be Alice, Bob says Carol can be Bob*".

For example, the default process of acquiring an X.509 identity (e.g., *John Smith*) during login incorporates a signing request to delegate that identity to the Grid Shell process (or IFS process, or OGRSH process), which itself has its own X.509 identity. Their private key is used to sign a SAML document asserting that "*John Smith says Grid Shell XYZ can be John Smith for 24 hour*s". In this fashion, a Genesis II process such as the Grid Shell can use a single X.509 certificate/keypair for authentication within message/transport-level protocols, yet still convey multiple identities/capabilities that have been cryptographically delegated to it.

Although SAML assertions can support delegation chains of arbitrary depth, the login tool has options that limit the duration and number of subsequent delegations allowable for an identity credential. For example, suppose the Grid Shell wishes to further delegate John Smith's identity to a job queue resource. The identity conveyed in the job-submission request is an enclosing SAML document with nested, signed statements asserting that "*John Smith says Grid Shell XYZ can be John Smith for 24 hours, Grid Shell XYZ says Queue 123 can be Grid Shell XYZ*". Hence Queue 123 can perform duties on behalf of John Smith, such as scheduling the job on an available BES when the time is appropriate.

The primary advantage of performing delegation via SAML assertions of this form, as opposed to using X.509 Proxy Certificates [x509Proxy], is that the identity of the delegatee is always known. X.509 Proxy certificates do not carry the subject of the delegatee; only the delegator.

3 Identity Provider Resources

3.1 IDPs for New Identities

New Grid identities can be created and managed using Genesis II Identity Provider (IDP) resources. An IDP is a Grid resource that delegates security credential(s) to the caller. IDP resources are exposed via a Genesis II implementation of the WS-Trust [wsTrust-1.3] Security Token Service (STS) Web service interface. Each Genesis II Container hosts an STS Web service instance that can be used to create and manage multiple IDP resources.

By creating new IDP resources, one can create new security credentials that can be used to indicate facts about a user-principal, such as identities (e.g., *Jane Smith*), roles (e.g., *Faculty*), privileges (e.g., *Level 5*), etc. IDP resources are typically given names and linked into the RNS namespace.

3.2 IDPs for Delegating Existing Identities

Many users are loath to let the private key for an existing security credential leave their machine/device. For example, a user-principal may wish to operate as *John Smith*, but may not want to expose the corresponding private key to the local Geneisis II installation, portal, etc. Such scenarios arise for mobile users who may be accessing the Grid from multiple clients over a period of time. In this case, such a user can create an IDP resource *JSmithProxy* around a delegated credential for their *John Smith* identity: "*John Smith says IDP resource JSmithProxy can be John Smith for 30 days*". By configuring login access to the IDP resource *JSmithProxy* to allow callers who carry a particular UsernameToken credential, the user-principal can request a restricted *John Smith* credential from any location using the login tool. The IDP's credential is then further delegated to the Grid client: "*John Smith says IDP resource JSmithProxy can be John Smith for 30 days, JSmithProxy says Grid Shell 123 can be JSmithProxy for 24 hours*".

3.3 IDPs for NIS, LDAP, etc.

Genesis II also allows administrators to configure identity provider service endpoints that virtualize the identities managed within an external JNDI-compatible directory system (e.g., NIS, LDAP, etc.).

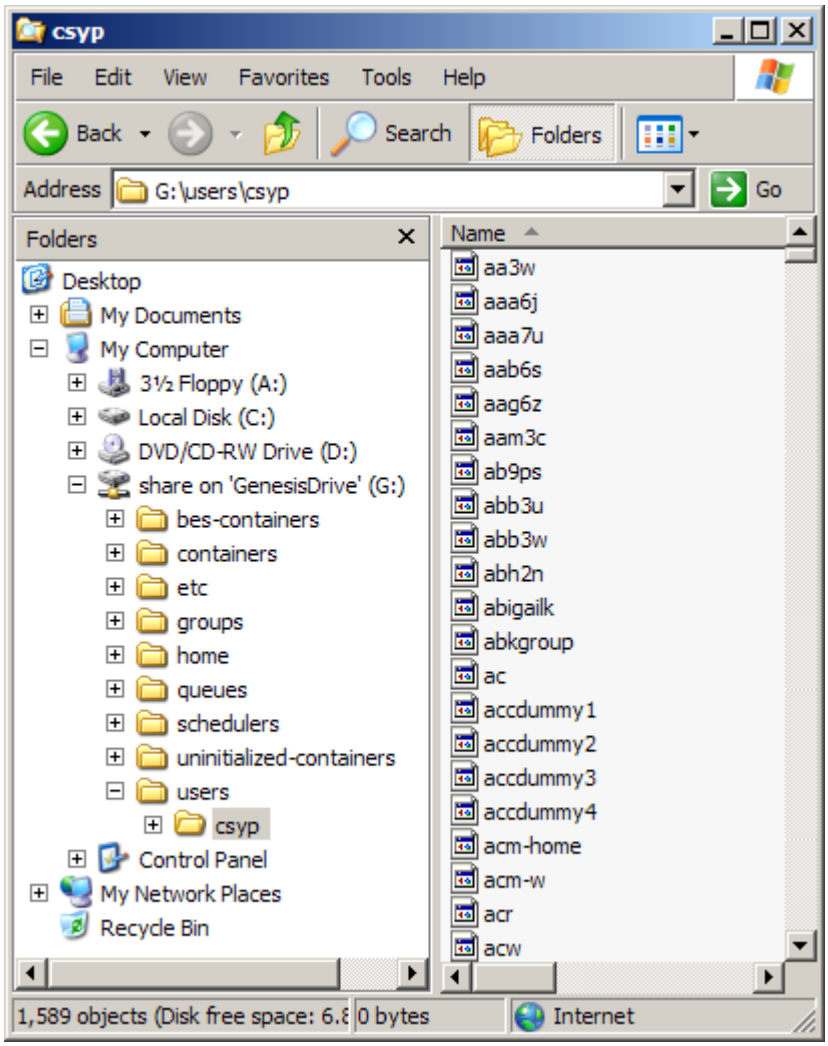

Figure 3: The "CSYP" STS shown here virtualizes the UVA Computer Science NIS directory as a collection of identity-provider resources

As an example, the interface of a NIS-configured STS endpoint is an RNS directory whose entries are IDP resources for the individuals who have NIS accounts in the specified NIS domain. A NIS IDP resource is similar to an IDP for a new identity in that a new X.509 certificate for the virtualized NIS account resource is created. This certificate identifies the user, the user's NIS UID, and the Grid NIS endpoint. In order to acquire a delegated security credential from a NIS IDP (i.e., login to it), the user must carry the specific UsernameToken token that can be used to authenticate the password hash of their NIS account.

3.4 Using IDPs for Single-Sign-On

In many cases, a user may wish to acquire multiple security credentials for their Grid session. It may be tedious and inconvenient to individually use the login command to obtain delegated credentials from every identity, group, and role IDP resource that the user-principal belongs to. To facilitate single-sign-on capability, our IDP resources implement the RNS directory interface. For example, RNS links to IDP resources such as *FacultyGroup* and *QueueAdministrators* can be placed within the directory exposed by the *JSmithProxy* IDP resource. This configuration signifies that a user-principal logging into *JSmithProxy* will automatically obtain delegated credentials for all three restricted identities.

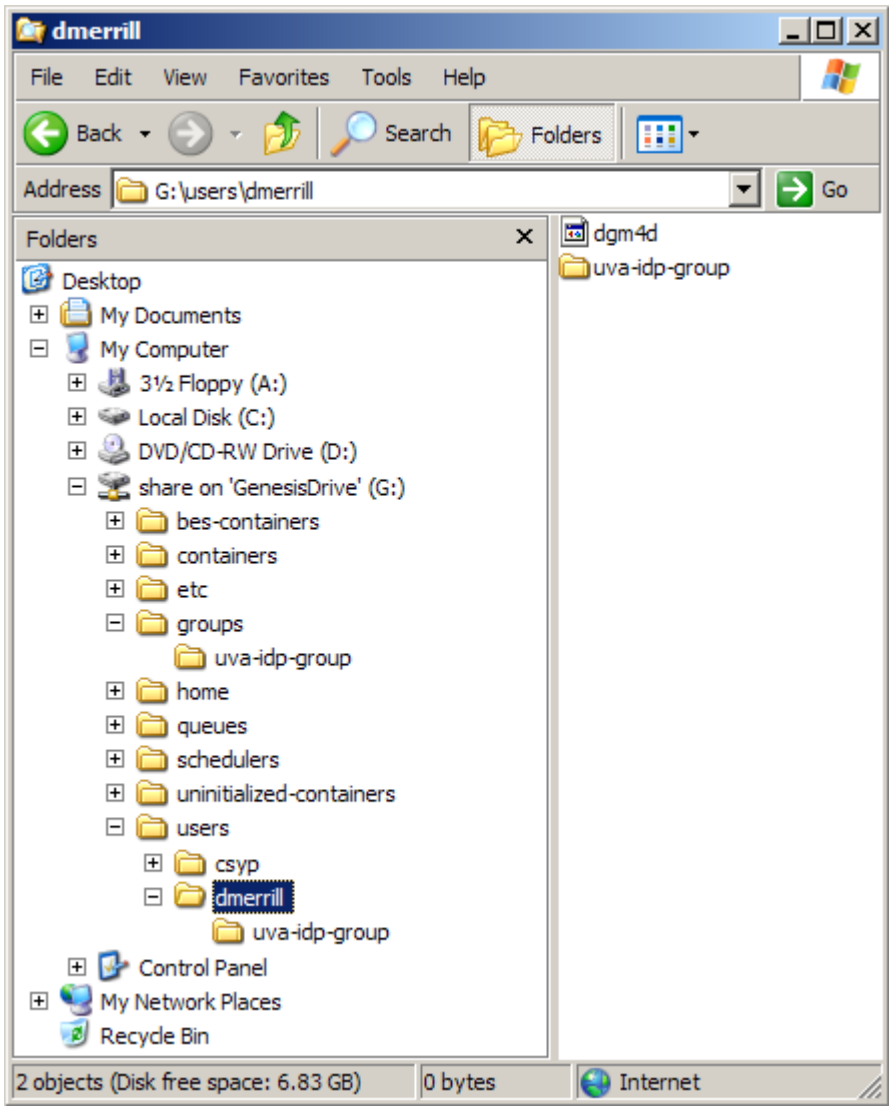

Figure 4: The "dmerrill" IDP show here contains links to the "VCGR" group and the "dgm4d" NIS entry, enabling single-sign-on acquisition of all three credentials from "dmerrill".

4 Access Control

The philosophy of site-autonomy empowers resource providers with the ability to specify security policies for their resources as they see fit. Each resource provider is responsible for ensuring that their resources are configured to allow only the desired access.

In Genesis II, access control decisions are made by pluggable authorization modules. This extensible design allows Genesis II installations to accommodate alternative authorization technologies. The default module uses access control lists (ACLs) to make authorization decisions at the granularity of logical resources (as opposed to doing so at the container or Web service granularity). When using this module, each Genesis II Grid resource maintains a set of three access-control lists: a read-access list, a write-access list, and an execute-access list. These lists can be populated with user-principal identities to allow varying degrees of access to different caller-entities bearing different security credentials.

For example, one can use the Genesis II security tools to add the *FacultyGroup* IDP resource to the read ACL of a ByteIO resource *Foo*. This has the effect of granting read-access of *Foo* to all users carrying the *FacultyGroup* credential. The security tools can add UsernameToken identities to these access-control lists, as well as identities from X.509 .cer certificate files that may be stored in the local filesystem or in the RNS Grid namespace.

As discussed in the previous section, Genesis II supports "group credentials" by allowing individuals to acquire multiple credentials, some of which can be used to logically indicate group roles or privileges. Genesis II also supports another grouping method that exploits the certificate-chain hierarchy conveyed within a user-principal's digital certificate. For example, one can configure the read-access list for file *Foo* to include the *UVa PKI Standard Assurance* identity. This has the effect of granting read-access to all users carrying security credentials whose certificates chain to the *UVa PKI Standard Assurance* certificate.

5 Secure Communication

Threat assessment is the process of identifying the specific security requirements for a given asset. In addition to establishing authorization requirements (i.e., who is allowed proper access), a threat assessment for a prospective resource to be exposed via the Grid may identify *confidentiality* and *integrity* requirements for the protection of communication over an untrusted network (e.g., the Internet).

Confidentiality is the assurance that only those principals authorized for information access are capable of doing so. Confidentiality guarantees in an insecure networked environment are usually derived through message encryption.

Integrity is the assurance that unauthorized changes made to communication messages can be detected by the recipient. Cryptographic digital signatures are generally used to enable the detection of message tampering.

Genesis II uses the SOAP protocol for messaging between communication endpoints. Fundamentally, the security of SOAP messages is affected by two aspects:

a) *Binding to a particular network transport protocol*. The SOAP protocol is extremely flexible. It can be enacted over virtually any other communication protocol, such as HTTP, SMTP, JMS [jms-1.1] message queues, etc. Genesis II uses HTTPS as the default transport, which uses anonymous-client SSL/TLS [tls-1.0] to provide security guarantees for properties such as container authentication, integrity, and confidentiality.

b) *Message-level credentialing and protection*. The SOAP protocol is sufficiently general in that it can support virtually any token credentialing system. In particular, Genesis II is compliant with the Web Services Security [wsSec-1.1] (WS-Security) family of specifications, a set of standards that define a general-purpose mechanism for associating security tokens with message content. WS-Security encompasses a set of profiles for encoding popular token types (e.g., X.509 [wsX509-1.1], Kerberos [wsKerb-1.1], SAML [wsSAML-1.1], and UsernameToken [wsUT-1.1] tokens). The WS-Security core specification also defines the application of XML-Encryption [xmlEnc02] and XML Digital Signature [xmlSig02] to provide end-to-end messaging integrity and confidentiality without reliance upon support from the underlying communication protocol.

In order to achieve real-world interoperability, Genesis II is also compliant with the WS-I Basic Security Profile [wsiBsp-1.0] (WS-I BSP). The WS-I BSP provides guidance on the use of WS-Security and its associated security token formats to resolve nuances and ambiguities between communicating implementations intending to leverage common security mechanisms.

Genesis II resources are intended to be configured with diverse integrity, confidentiality, and authorization requirements. These requirements are independent of the resource's service interface, yet still affect message format. As such, interoperability between Genesis II peers is dependent upon the ability to normatively describe and advertise individual secure communication requirements.

Genesis II adheres to the OGF Secure Addressing Profile [secAddr-1.0] (SecAddr) to distribute secure communication requirements within WS-Addressing endpoint references (EPRs). More specifically, these requirements are conveyed as WS-SecurityPolicy documents that have been embedded within EPRs. The WS-Addressing endpoint reference data structure is a useful construct because it provides the "invocation context" for a service endpoint—all of the necessary information that a client requires to establish meaningful communication. Additionally, the Genesis II is capable of signing EPR documents in order to provide guarantees of trust regarding the identity of the minter and the integrity of the EPR; useful properties given the OGSA models of storing and exchanging EPRs within intermediary services.

The EPR for a given Genesis II resource reflects the authorization token types required by that resource as well as any requirements for digital signature that would be necessary to authenticate those tokens. Genesis II security tools can also configure resources to have additional message-level confidentiality (and/or integrity requirements, if not already necessary for authentication).

6 References

ituOsi ITU. Security Architecture for Open Systems Interconnection for CCITT Applications. In *CCITT Recommendation X.800*, 1991

ituX509 ITU-T. Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks. In *ITU-T Recommendation X.509*, 2005

wsX509-1.1 OASIS. Web Services Security X.509 Certificate Token Profile 1.1. OASIS Standard Specification, 2006.

wsKerb-1.1 OASIS. Web Services Security Kerberos Token Profile 1.1. OASIS Standard Specification, 2006.

wsSAML-1.1 OASIS. Web Services Security SAML Token Profile 1.1. OASIS Standard Specification, 2006.

wsUT-1.1 OASIS. Web Services Security Username Token Profile 1.1. OASIS Standard Specification, 2006.

wsSec-1.1 OASIS. Web Services Security: SOAP Message Security 1.1. OASIS Standard Specification, 2006.

saml-2.0 OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005.

wsSecPol-1.2 OASIS. WS-SecurityPolicy 1.2. OASIS Standard Specification, 2007

wsTrust-1.3 OASIS. WS-Trust 1.3. OASIS Standard, 2007.

tls-1.0 T. Dierks and C. Allen. The TLS Protocol Version 1.0. IETF RFC 2246, 1999.

xmlEnc02 W3C. XML Encryption Syntax and Processing. W3C Recommendation, 2002.

xmlSig02 W3C. XML-Signature Syntax and Processing. W3C Recommendation, 2002.

wsiBsp-1.0 WS-I. Basic Security Profile Version 1.0. WS-I Final Material, 2007.

secComm-1.0 D. Merrill. Secure Communication Profile 1.0. Open Grid Forum, GFD.132, 2007.

secAddr-1.0 D. Merrill. Secure Addressing Profile 1.0. Open Grid Forum, GFD.131, 2007.

jms-1.1 M. Hapner et al. Java Message Service Specification 1.1. Sun Microsystems, Inc., 2002

x509Proxy S. Tueke et al. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. IETF RFC 3820, 2004.