

Grid Security Infrastructure (GSI) Roadmap

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/1id-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

Abstract

This document provides an overview or "roadmap" of the work done by the Grid Forum Grid Security Infrastructure (GSI) working group. It describes some of the terminology used in the working group's documents, the theory, motivations, and requirements behind GSI, and the various pieces that together comprise GSI. It identifies each document developed by the GSI working group, and describes the relationships among these documents and various IETF standards documents. It also provides advice to would-be GSI implementers about some of the issues discussed at length during GSI development, in hopes of making it easier to build implementations that will actually interoperate.

Table of Contents

| | |
|--|---|
| Grid Security Infrastructure (GSI) Roadmap | 1 |
| Status of this Memo | 1 |
| Abstract | 1 |
| Table of Contents | 1 |
| 1. Introduction | 2 |
| 2. Conventions used in this document | 3 |
| 3. Terminology..... | 3 |
| 3.1. Protocols, Services, APIs, SDKs | 3 |

| | | |
|--------|---|----|
| 3.1.1. | Protocol..... | 3 |
| 3.1.2. | Service | 4 |
| 3.1.3. | Software Development Kit (SDK) | 4 |
| 3.1.4. | Application Program Interface (API) | 5 |
| 4. | Security Terminology | 5 |
| 5. | GSI Theory | 6 |
| 5.1. | Grid Security Requirements | 6 |
| 5.2. | Existing Standards | 8 |
| 5.3. | GSI Solutions | 10 |
| 6. | Documents | 10 |
| 6.1. | Grid Security Protocols (GSP) | 11 |
| 6.2. | Grid Security APIs..... | 12 |
| 6.3. | Grid Security Services..... | 13 |
| 6.3.1. | SSL-K5 and PKINIT | 13 |
| 6.3.2. | K5Cert | 13 |
| 6.3.3. | MyProxy | 14 |
| 6.4. | Other Topics to Consider | 15 |
| 7. | Advice To Implementers | 15 |
| 8. | Acknowledgments | 15 |
| 9. | References | 15 |
| 10. | Security Considerations | 16 |
| 11. | Contact Information..... | 16 |

Introduction

The term "the Grid" was coined in the mid 1990s to denote a proposed distributed computing infrastructure for advanced science and engineering [9, 15]. The Grid Security Infrastructure (GSI) was subsequently developed, based on existing standards, to address the unique security requirements that arise in Grid environments, as described in [6, 8]. The formation of the Grid Forum as a body to promote and develop Grid technologies, and the broad acceptance that GSI has received within the Grid community, led to the formation of the Grid Forum GSI working group to foster the specification and development of GSI. This document is an informational Grid Forum draft that provides an overview or "roadmap" of the work done, and documents produced by, the GSI working group. It is intended to provide information; there are no requirements or specifications in this document.

Section 2 of this document defines terminology used in the working group's documents. Section 3 covers "GSI theory;" it explains what were the GSI working group's basic assumptions, motivations, and requirements, along with how these relate to existing standards and solutions. Section 4 provides an overview of the various documents that comprise GSI. It identifies which documents address which areas, and describes the relationships among the various documents. Section 5 contains "Advice to implementers." Its primary purpose is to capture some of the major issues discussed by the GSI working group, as a way of explaining WHY some of the requirements and specifications say what they say. This should cut down on the number of misinterpretations of the documents, and help developers build interoperable

implementations. Section 6 contains a list of contributors we wish to thank. Section 7 provides a list references. Section 8 discusses security considerations. Section 9 provides contact information for the editors.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [5].

Terminology

The next subsections define various terms used throughout this document.

Protocols, Services, APIs, SDKs

We first define four terms that we will use extensively throughout the GSI working group documents: protocol, service, SDK, and API.

1.1.1. Protocol

A *protocol* is a set of rules that end points in a telecommunication system use when exchanging information. For example:

- The Internet Protocol (IP) defines an unreliable packet transfer protocol.
- The Transmission Control Protocol (TCP) builds on IP to define a reliable data delivery protocol.
- The Transport Layer Security (TLS) Protocol [7] defines a protocol to provide privacy and data integrity between two communicating applications. It is layers on top of some reliable transport protocol such as TCP.
- The Lightweight Directory Access Protocol (LDAP) builds on TCP to define a query-response protocol for querying the state of a remote database.

An important property of protocols is that they admit to multiple implementations: two end points need only implement the same protocol to be able to communicate. Standard protocols are thus fundamental to achieving interoperability in a distributed computing environment.

A protocol definition also says little about the behavior of an entity that speaks the protocol. For example, the FTP protocol definition indicates the format of the messages

used to negotiate a file transfer, but does not indicate what a receiving entity should do with the file once received.

As the above examples indicate, a protocol is usually layered on top of some other more general protocol.

1.1.2. Service

A *service* is an entity that speaks a particular protocol and implements a particular behavior. (I.e., "service = protocol + behavior.") For example:

- An FTP server speaks the File Transfer Protocol, and supports remote read and write access to a collection of files. Different FTP server implementations may support different behaviors. For example, one may allow access to files on the server's disk, another may allow access to files on a mass storage system, and another may perform caching of files in memory to improve performance under certain conditions.
- An LDAP server speaks the LDAP protocol, and supports response to queries. One LDAP server implementation may respond to queries using a database of information, while another may respond to queries by dynamically making SNMP calls to generate the necessary information on the fly.

A service definition may permit a variety of implementations. For example, a service may or may not be persistent (i.e., always available); be able to detect and/or recover from certain errors; run with privileges; and/or have a distributed implementation for enhanced scalability. If variants are possible, then discovery mechanisms that allow a client to determine the properties of a particular instantiation of a service are important.

1.1.3. Software Development Kit (SDK)

We use the term *Software Development Kit (SDK)* to denote a set of code designed to be linked with, and invoked from within, an application program to provide specified functionality. Some SDKs provide access to services via a particular protocol. For example:

- The OpenLDAP release includes an LDAP client SDK, which contains a library of functions that can be used from a C or C++ application to perform queries to an LDAP service.
- JDNI is a Java SDK, which contains functions that can be used to perform queries to an LDAP service.

There may be multiple SDKs, for example from multiple vendors, which implement a particular protocol. Further, for client-server oriented protocols, there may be separate client SDKs for use by applications that want to access a services, and server SDKs for use by service implementers that want to implement particular, customized service behaviors.

An SDK need not speak any protocol. For example, an SDK that provides numerical functions may act entirely locally, and not need to speak to any services to performs its operations.

1.1.4. Application Program Interface (API)

An *Application Program Interface (API)* defines a standard interface (e.g., set of subroutine calls, or objects and method invocations in the case of an object-oriented API) for invoking a specified set of functionality. For example:

- The Generic Service Service (GSS) API [13] defines standard functions for verifying identify of communicating parties, encrypting messages, and so forth.
- The MPI API defines standard interfaces, in several languages, to functions used to transfer data among processes in a distributed or parallel computing system.

An API is, by definition, language specific, although multiple language bindings may be defined. The language may be a conventional programming language such as C or Java, or it may be a shell interface. In the latter case, the API refers particular definition of command line arguments to the program, the input and output of the program, and the exit status of the program.

An API will normally specify a standard behavior, but can admit to multiple implementations. In other words, there may be multiple SDKs that implement the same API.

It is important to understand the relationship between APIs and protocols. A protocol definition says nothing about the APIs that might be used to generate protocol messages. A single protocol may have many APIs; a single API may have multiple implementations that target different protocols. In brief, standard APIs enable *portability*; standard protocols enable *interoperability*. For example, both public key and Kerberos bindings have been defined for the GSS-API. Hence, a program that uses GSS-API calls for authentication operations can operate in either a public key *or* a Kerberos environment without change. On the other hand, if we want a program to operate in a public key *and* a Kerberos environment at the same time, then we need something new: a standard protocol that supports interoperability of these two environments.

Security Terminology

Security is a complex, multi-faceted problem. In order to understand Grid security requirements, one must understand some basic terms:

- *Identity (or Subject or Principal)*: This is the name of a particular person, service, or entity.

- *Authentication*: A procedure by which one party is able to determine the identity of another party.
- *Credentials*: A bundle of information containing identity and other related information, which is used during the authentication procedure.
- *X.509*: The standard credential format used in public key security infrastructures.
- *Impersonation*: When one entity to assumes the identity of another identity for the purpose of authentication.
- *Delegation*: When one entity grants ability to act on its behalf to another entity.
- *Authorization*: Once authentication has established identity, this is the procedure of determining what that subject is allowed to do.
- *Message integrity*: Protection of communication, which ensures that the contents of a message cannot be modified by an attacker.
- *Message confidentiality*: Protection of communication, which ensures that the contents of a message cannot be read by an attacker. This is sometimes called “encryption”, but this is confusing as encryption algorithms underlie many security operations aside from message confidentiality. This is sometimes also called “message privacy”, but privacy also tends to have another meaning as defined below.
- *Digital signature*: A cryptographic procedure that allows one entity to verify that a particular piece of data was produced by a particular subject. (SJT: This definition is not quite right.)
- *Non-repudiation*: A cryptographic procedure that allows one entity to prove that a piece of data was undeniably produced by a particular subject, perhaps at a particular time. (SJT: This definition is not quite right.)
- *Privacy*: This is protection from unauthorized information disclosure.

Other good security definitions can be found in RFC 2510, Section 1.2, Definitions of PKI Entities [3], including definitions for Subjects and End Entities, Certification Authority, Registration Authority.

GSI Theory

This section explains what were the GSI working group's basic assumptions, motivations, and requirements, along with how these relate to existing standards and solutions.

Grid Security Requirements

A Grid security solution should be based on existing standards wherever possible. Security is an extremely complex problem, with specific solutions incrementally developed over many years by many extremely talented people. Further, the community generally only trusts a particular security solution if it has stood the tests of time and repeated scrutiny.

However, Grid environments have a broad range of security requirements [6, 8]. Unfortunately, no single, existing, standard security solution addresses all of these requirements, though ideally a Grid security solution would extend existing standards.

Grid authentication requirements include:

- *Single sign on*

Users must be able to "log on" (authenticate) just once and then have access to any resource in the Grid that they are authorized to use, without further user intervention.

- *Delegation*

A user must be able to endow to a program the ability to run on that user's behalf, so that the program is able to access the resources on which the user is authorized. The program should (optionally) also be able to further delegate to another program.

- *Integration with various local security solutions*

Each site or resource provider may employ any of a variety of local security solutions, including Kerberos, Unix security, etc. The Grid security solution must be able to interoperate with these various local solutions. It cannot require wholesale replacement of local security solutions, but rather must allow mapping into the local environment.

- *User-based trust relationships*

In order for a user to use resources from multiple providers together, the security system must not require each of the resource providers to cooperate or interact with each other in configuring the security environment. In other words, if a user has the right to use sites A and B, the user should be able to use sites A and B together without requiring the security administrators from sites A and B to interact.

Grid requirements for communication protection include:

- *Flexible message protection*

An application must be able to dynamically configure a service protocol to use various levels of message protection, including none, just integrity, or integrity plus confidentiality. The choice may be motivated by factors such as sensitivity of the messages, performance requirements, the parties involved in the communication, and

the infrastructure over which the message is transiting.

- *Supports various reliable communication protocols*

While TCP is the dominant, and widely available, reliable communication protocol for the Internet, the security mechanisms must be usable with a wide assortment of other reliable communication protocols. For example, performance requirements may dictate the use of non-TCP protocols for use within specialized environments.

- *Supports independent data units (IDU)*

Some applications require "protection of a generic data unit (such as a file or message) in a way which is independent of the protection of any other data unit and independent of any concurrent contact with designated 'receivers' of the data unit" [1]. For example, streaming media, email, and unreliable UDP datagrams all require this form of protection.

Grid authorization requirements include:

- *Authorization by stakeholders*

Resource owners or stakeholders must be able to control which subjects can access the resource, and under what conditions.

- *Restricted delegation*

In order to minimize exposure from compromised or misused delegated credentials, it is desirable to have rich support for the restriction of the authorization rights that are delegated.

Existing Standards

A variety of security standards exist, but none address all of the requirements described in the previous section. This section briefly describes the relationship of some of these existing standards to the above requirements.

- *Kerberos* [12]

This is a widely accepted IETF standard for maintaining site security via authentication, message integrity, message confidentiality, based on shared secret cryptography. It provides many of the requirements for a Grid, such as single sign-on, delegation, and flexible message protection. Unfortunately, it does not address the following requirements:

- *Integration with various local security solutions*

Kerberos implementations tend to replace local security solutions, not integrate

with them. This means that every resource in a virtual organization must be running Kerberos, not only as the security for interoperating with other members of the virtual organization, but also as the native, local security solution. This is currently not palatable to many sites.

- *User based trust relationships*

Even if Kerberos were ubiquitously deployed, Kerberos still suffers from the fact that it requires site-based trust relationships, rather than user-based trust relationships. The fact that a user already has trust relationships with multiple sites (i.e. the user can login to each Kerberos realm) is not sufficient under Kerberos to allow that user to use resources at multiple sites as part of a single, secure, distributed operation. For this to work, the Kerberos security administrators of those two realms must set up inter-realm trust agreements. History has proven that this not feasible in practice except in the case of tightly controlled Grids such as in the military and other classified networks.

- *TLS* [7]

Previously known as SSL, this is the widely accepted IETF standard for Web authentication, message integrity, and message confidentiality, based on public key cryptography. Its strengths and weaknesses are somewhat the opposite of Kerberos, in that TLS addresses issues of user-based trust relationships, but does not address single sign-on and delegation.

- *PKIX* [4]

This is a set of IETF standards that describe protocols and syntax for managing X.509 credentials for public key security infrastructures. These may be used in conjunction with, for example, TLS.

- *CMS (Cryptographic Message Syntax)* [10]

This IETF standard defines a syntax "to digitally sign, digest, authenticate, or encrypt arbitrary messages" [10]. In other words, it defines a standard way to protect independent data units (IDUs).

- *GSS-API (Generic Security Service API)* [13, 16]

This is an IETF standard which defines an API for providing authentication, message integrity, and message confidentiality. It assumes two party, reliable, connection oriented communication, such as TCP/IP or any data communication protocol with these properties. It can be used with any of a variety of underlying security mechanisms. It supports, or is neutral to, all of the Grid requirements described above. Therefore it is an excellent API for Grid programming.

- *IDUP-GSS-API (Independent Data Unit Protection GSS-API)* [1]

This IETF standard extends the GSS-API to include support for protection of independent data units. This allows for protection of other forms of communication, such as unreliable, out-of-order, multicast, and/or connectionless.

- *SPKM (Simple Public Key GSS-API Mechanism)* [2]

This is an IETF standard GSS-API binding to public key protocols for authentication, message integrity, and message confidentiality. It does not support delegation. While commercial implementations of SPKM exist, to date it has not been widely adopted.

GSI Solutions

The GSI protocols, APIs, and services were assembled and/or developed to address the authentication and communication protection requirements described above, while exploiting existing standards to the greatest extent possible. GSI provides interdomain security protocols that bridge the gap between the different local security solutions at a Grid's constituent sites. The significant features of GSI as follows:

- Each entity (user, resource, service, etc.) is assigned a globally unique identity. We represent identity by a *certificate*, which specifies the name and additional information that can be used to identify the entity (e.g., a public key). In GSI, we represent certificates using the standard X.509 format. A *certificate authority*, or CA, is a trusted third party that is responsible for assigning name to an entity.
- Each entity is also provided with a means of proving that it possesses a specific identity. In GSI, identity checking is implemented by the authentication algorithm defined by the TLS protocol. The veracity of the entity's identity is only as good as the trust placed in the CA that issued the certificate in the first place. Thus the authentication algorithm must validate the identity of the CA as part of the authentication protocol.
- An entity may delegate a subset of its rights to a third party (such as a process created by a program) by creating a temporary identity called a *proxy*. A proxy certificate is a certificate signed by the user or a previous proxy for the user, thus creating a chain of signatures terminating with the CA that issued the initial certificate. By checking the certificate chain, processes started on separate sites by the same user can authenticate to one another without requiring that the user send his or her credentials to either site. A proxy may also have associated with it a specification of what operations the proxy credential can be used to perform, in which case it is termed a *restricted proxy*.

Documents

The Grid Security Infrastructure is defined by a particular combination of new and existing standard protocols, APIs, and services.

Grid Security Protocols (GSP)

The Grid Security Protocols (GSP) are defined by a particular combination of protocols and syntaxes:

- *Transport Layer Security Protocol (RFC 2246) [7]*

Provides for public key based authentication, message integrity, and message confidentiality between two parties over a reliable, connection oriented communication channel (e.g. TCP/IP).

- *X.509 Certificate and CRL Profile (RFC 2459) [11]*

GSP credentials conform to the X.509 certificate format. Standard certificate revocation list (CRL) formats are used for invalidating GSP credentials.

- *X.509 Proxy Certificates (Grid Forum draft)*

GSP defines a novel way of creating and checking user signed X.509 certificates to allow for single sign-on and delegation.

- *X.509 Proxy Delegation Protocol (Grid Forum draft)*

GSP defines a protocol that allows for remote creation of an X.509 proxy certificate over a secure, reliable, connection oriented communication channel. This defines how GSP supports remote delegation on top of TLS.

To ensure interoperability amongst Grid security implementations, any implementation that claims to implement GSP must implement the above standards.

In the future, GSP will be extended to include:

- *Cryptographic Message Syntax (RFC 2630) [10]*

Provides for public key based authentication, message integrity, message confidentiality, and digital signatures of independent data units (IDUs). This allows for authenticated and protected communication over unreliable, out-of-order, multicast, and/or connectionless mechanisms. Delegation can be performed over these mechanisms by using CMS to encrypt an X.509 proxy certificate.

- *Online Certificate Status Protocol (RFC 2560) [14]*

"In lieu of or as a supplement to checking against a periodic CRL, (OCSP enables applications) to obtain timely information regarding the revocation status of a certificate" [14].

- *X.509 Extension For Restricted Proxy Certificates (Grid Forum draft)*

In order to properly protect proxy credentials against compromise or misuse, GSP defines an X.509 extension within a proxy certificate which describes restrictions upon what the proxy certificate can be used for.

Grid Security APIs

It is recommended that a GSP compliant SDK implement the following standard APIs:

- *GSS-API (RFC 2743) [13, 16]*

This API allows an application to perform authentication, message integrity, and message confidentiality, and delegation when doing two party, reliable, connection oriented communication.

- *GSS-API Extensions for the Grid (Grid Forum draft)*

Experience in using GSS-API for numerous Grid applications has shown that small extensions to GSS-API are required. These extensions allow for better management of multiple credentials by an applications, and for more flexible delegation.

- *GSI Shell API (Grid Forum draft)*

This defines shell interfaces to GSI operations such as proxy creation (i.e. login), proxy destruction (i.e. logout), proxy inquiry, etc.

In the future, as GSP is extended as described above, it is recommended that a GSP compliant SDK also implement the following API:

- *IDUP-GSS-API (RFC 2479) [1]*

This API allows an application to perform authentication, message integrity, message confidentiality, and delegation

Currently, there are no standards for you the above APIs relate to the GSP protocols. The GSI working group will develop the following standard bindings:

- *The Grid GSS-API Mechanism (Grid Forum draft)*

This document defines how the API functions defined in GSS-API and GSS-API Extensions for the Grid map to the GSP protocols.

- *The Grid IDUP-GSS-API Mechanism (Grid Forum draft)*

This document defines how the API functions defined in IDUP-GSS-API map to the GSP protocols.

Grid Security Services

Various services are required for ease of use and interoperability with local security environments. Several such services are describes in this section. These services will be defined by GSI working group documents that define their protocols, and possibly client APIs.

1.1.5. SSL-K5 and PKINIT

These services allow a client to use GSP proxy credential to obtain a local Kerberos Ticket Granting Ticket (TGT). This is required for GSI to interoperate with a local site that is running Kerberos – when a user authenticates with a service via GSP, that site service can use the delegated GSP proxy credential to obtain a local TGT that allows the service access to local resources, including local AFS and DCE DFS file systems. SSL-K5 is a simple version of this service, based on the SSL protocol, which can be used with existing Kerberos Domain Controllers (KDCs). PKINIT is an IETF draft standard, which will replace SSL-K5 as it becomes more widely available in KDCs.

1.1.6. K5Cert

This is a Kerberos service and client (i.e. it uses the Kerberos authentication protocol) that allows an authenticated Kerberos user to generate a GSP proxy credential. This service is useful for sites that want to give their users easy access to Grid resources via GSP based protocols. Once the user has logged into the local Kerberos realm, it is easy (and can even be automated) for the user to obtain a GSP proxy credential for Grid use.

There are three variants of this service:

- *Online CA*

The K5cert service has its own Certificate Authority (CA) certificate and keys, and can create certificates on-the-fly that are signed by this CA. So when the K5cert service receives an authenticated request for a proxy certificate, it generates a (probably short term) X.509 certificate that it returns to the client.

An advantage of this approach to a user is simplicity -- the user never needs to worry about obtaining or protecting a normal, long-term X.509 certificate from a CA. Instead, K5cert generates proxy certificates for the user as needed.

A disadvantage of this approach is that it introduces another CA. This means that if the user wants to access Grid services at another site, that other site must trust this CA. In addition, if the K5cert service is compromised, an attacker could create Grid proxy credentials for any user at the site.

- *User long-term certificate repository*

In order to remove the disadvantage of the online CA approach, a user could obtain a normal, long-term X.509 certificate from some CA. But instead of managing and protecting that credential, the user turns the certificate and private key over to K5cert. The K5cert service would manage and protect this certificate and private key in its own certificate repository. So when the K5cert service receives an authenticated request for a proxy certificate, it retrieves the user's certificate and private key from the repository, and generates the proxy certificate that it returns to the client.

An advantage to this approach is user simplicity, while not introducing another CA. It also allows a site to protect the user's private key better than an uneducated or careless user might protect their own key.

A disadvantage of this approach is that an attacker who compromises the K5cert service would gain access to the private keys in the repository. This would, in turn, require users to obtain new long-term credentials from their CA.

- *User proxy repository*

This is a variant of the certificate repository approach. Instead of the repository holding the user's normal, long-term certificate and key, it would instead hold proxy certificates that are delegated to the service from users.

An advantage of this approach is the expert users maintain control over their own long-term private key. If the K5cert service is compromised, the user's private key is not compromised. The user need only delegate a new proxy to the K5cert repository after it is secured.

A disadvantage of this approach is that it puts more burden on the user to properly manage and protect their own certificate and private key.

1.1.7. MyProxy

This is similar to the K5cert user proxy repository service, except that proxies are associated with a name and password. Instead of a client using Kerberos to authenticate with the MyProxy service, the client and server instead use TLS in Ephemeral Diffie-Hellman mode to establish a confidential (but un-authenticated) channel between them. The client then supplies the server with a name and password. The service compares the name and password with its repository, and if a match is found then it delegates a user proxy back to the client.

The MyProxy service has proven useful for Web portals to Grid resources. A user can use the MyProxy client to place a proxy into the service's repository. Sometime later, the client can then login to a Web portal from anywhere using a name and password. The portal can then use the MyProxy client to retrieve a proxy for the user from the MyProxy service.

Other Topics to Consider

Future work on GSI should likely include:

- Authorization API, protocols, and languages.
- Standards for managing the mapping of Grid user identities to local identities. For example, this might include a standard "Gridmap file" syntax, and a service protocol for management of this mapping.
- Firewalls: How GSI relates to firewalls should be further explored.

Advice To Implementers

You're on your own. :-)

Acknowledgments

We are grateful to numerous colleagues for discussions on the topics covered in this paper, in particular (in alphabetical order, with apologies to anybody we've missed): Joe Bester, Randy Butler, Carl Kesselman, Doug Engert, Ian Foster, Marty Humphrey, Cliff Neuman, Gene Tsudik, Von Welch.

The organization and other basic text of this memo were taken from the IETF PKIX Roadmap draft.

This work was supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38; by the Defense Advanced Research Projects Agency under contract N66001-96-C-8523; by the National Science Foundation; and by the NASA Information Power Grid project.

References

- [1] Adams, C., "Independent Data Unit Protection Generic Security Service Application Program Interface (IDUP-GSS_API," RFC 2479, December 1998.
- [2] Adams, C., "The Simple Public-Key GSS-API Mechanism (SPKM)," RFC 2025, October 1996.
- [3] Adams, C. and S. Farrell, "Internet X.509 Public Key Infrastructure, Certificate Management Protocols," RFC 2510, March 1999.
- [4] Arsenault, A. and S. Turner, "Internet X.509 Public Key Infrastructure, PKIX Roadmap," Internet Draft, March 10 2000.

- [5] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997.
- [6] Butler, R., D. Engert, I. Foster, C. Kesselman, and S. Tuecke, "A National-Scale Authentication Infrastructure," *IEEE Computer*, vol. 33, pp. 60-66, 2000.
- [7] Dierks, T. and C. Allen, "The TLS Protocol, Version 1.0," RFC 2246, January 1999.
- [8] Foster, I., C. Kesselman, G. Tsudik, and S. Tuecke, "A Security Architecture for Computational Grids," presented at Proceedings of the 5th ACM Conference on Computer and Communications Security, 1998.
- [9] Foster, I., C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of Supercomputer Applications*, 2001.
- [10] Housley, R., "Cryptographic Message Syntax," RFC 2630, June 1999.
- [11] Housley, R., W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," RFC 2459, January 1999.
- [12] Kohl, J. and C. Neuman, "The Kerberos Network Authentication Service (V5)," RFC 1510, September 1993.
- [13] Linn, J., "Generic Security Service Application Program Interface, Version 2, Update 1," RFC 2743, January 2000.
- [14] Myers, M., R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol," RFC 2560, June 1999.
- [15] Stevens, R. and e. al., "From the I-WAY to the National Technology Grid," *Communications of the ACM*, vol. 40, pp. 50-61, 1997.
- [16] Wray, J., "Generic Security Service API Version 2, C-bindings," RFC 2744, January 2000.

Security Considerations

This document contains an overview or "roadmap" of the security protocols and service interfaces defined in other documents (see section 4), which together define the Grid Security Infrastructure. Refer to these other documents for security considerations.

Contact Information

Steven Tuecke

Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439
Phone: 630-252-8711
Email: tuecke@mcs.anl.gov