Bachelor Thesis

Konzeption einer Anbindung eines Authentifizierungs-Frameworks (Shibboleth) an das Autorisierungssystem einer Grid-Ressource

Frank Kautz
Hochschule Darmstadt

DLR

Hochschule Darmstadt

- Fachbereich Informatik -

Konzeption einer Anbindung eines Authentifizierungs-Frameworks (Shibboleth) an das Autorisierungssystem einer Grid-Ressource

Abschlussarbeit zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

vorgelegt von

Frank Kautz

Referent: Prof. Dr. Peter Wollenweber

Korreferent: Prof. Dr. Frank Bühler

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, den

Abstrakt

In Grid-Systemen werden Ressourcen einer großen Zahl von Benutzern bereitgestellt. Diese Ressourcen werden von unterschiedlichen Organisationen bereitgestellt und die Benutzer kommen ebenfalls aus unterschiedlichen Organisationen. Dies wirft die Frage nach Sicherheitsrichtlinien auf, die den Zugriff auf die Ressourcen reglementieren. Zu diesem Zweck werden in Grid-Systemen virtuelle Organisationen gebildet, in denen Sicherheitsrichtlinien definiert werden. Die Authentifizierung der Benutzer wird mit X.509-Zertifikaten realisiert.

Im Rahmen dieser Bachelorarbeit soll ein Konzept entwickelt werden, dass die Durchsetzung der Sicherheitsrichtlinien einer virtuellen Organisation betrifft. Das Shibboleth-Framework soll in die Autorisierungskonzepte von Grid-Systemen eingebunden werden. Für die Autorisierung stellt das Shibboleth-Framework die Attribute eines Benutzers bereit. Grid-Systeme stellen Ressourcen bereit, dies können Rechenkapazität, Datenbanken oder Sensoren sein. Es gibt aber auch Applikationen oder Frameworks, die als Grid-Ressource bereitgestellt werden. Solche Grid-Ressourcen können auch eigene Autorisierungssysteme besitzen. Diese müssen ebenfalls die Sicherheitsrichtlinien der virtuellen Organisation durchsetzen. An diese Autorisierungssysteme sollen die Benutzerattribute weitergeleitet werden, damit diese bei der Autorisierung berücksichtigt werden können. Für die Integration der Benutzerattribute in das Autorisierungssystem einer Grid-Ressource wird exemplarisch ein rollenbasiertes Autorisierungssystem angenommen.

Damit die Aufgabenstellung umgesetzt werden kann, muss analysiert werden, welche Anforderungen an die Authentifizierung und Autorisierung in Grid-Systemen gestellt werden. Wie diese Anforderungen von den Grid-Systemen umgesetzt werden und welche Möglichkeiten es gibt diese Implementierung zuerweitern. In diesem Zusammenhang wird auf die Grid-Middleware Globus Toolkit eingegangen. In der weiteren Analyse müssen die Konzepte des Shibboleth-Frameworks dargestellt werden und die zugrunde liegenden Technologien aufgezeigt werden. Abschließend muss gezeigt werden, wie die Konzepte der Grid-Systeme und die Konzepte des Shibboleth-Frameworks miteinander verbunden werden können, damit eine Zusammenarbeit der beiden Systeme realisiert werden kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Motivation und Ziele

Motivation

Gemeinsam mit Partnern aus der Schiffbauindustrie und der Informationstechnik wird im Projekt SESIS ein integriertes schiffbauliches Entwurfs- und Simulationssystem entwickelt. Das Basis-System des Entwurfs- und Simulationssystems bildet RCE (Reconfigurable Computing Environment).

Bei RCE handelt es sich auf technischer Ebene um ein Java-Framework, das auf Plug-Ins basiert und den Zugriff auf verteilte Dienste und Ressourcen ermöglicht. Das RCE-System stellt eine Webservice-Schnittstelle bereit, diese kann von Grid-Systemen verwendet werden.

Ein Grid ist ein System, das verteilte Ressourcen verwaltet, diese können Datenbanken, Anwendungen, Rechenkapazitäten usw. sein. Das Konzept eines Grid ist nicht abhängig von einer speziellen Hardware oder einem speziellen Betriebssystem. Alle Ressourcen eines Grids sind über ein Netzwerk miteinander verbunden und mittels einer Grid-Middleware gekoppelt. Die Grid-Middleware bildet die universelle Schnittstelle zwischen den Ressourcen.

Die Ressourcen eines Grid können zu einer Organisation gehören oder auch organisationsübergreifend vorhanden sein. Diese Ressourcen werden zu einer virtuellen Organisation (VO) zusammen geschlossen. Jede VO ist für die Authentifizierung und Autorisierung von Benutzern zuständig, die auf die Ressourcen der jeweiligen VO zugreifen. In dieser Arbeit soll diese Aufgabe durch das Authentifizierungs-Framework Shibboleth abgewickelt werden. Durch den Einsatz von Shibboleth soll eine rollenbasierte Autorisierung in den Grid-Ressourcen realisiert werden. Dies ermöglicht eine feinere Rechtevergabe in der Grid-Ressource und führt zu einer besseren Datensicherheit.

Ziele

In dieser Bachelor-Arbeit soll ein Konzept zur Anbindung des Authentifizierungs-Frameworks Shibboleth an das RCE-System entwickelt werden. Dabei soll explizit auf die Grid-Middleware Globus eingegangen werden. Die benötigte Infrastruktur für die Anbindung des Authentifizierungs-Frameworks Shibboleth soll dargelegt werden. Bei der Erstellung des Konzeptes ist zu berücksichtigen, dass die Authentifizierung von X.509-Long Lived Certificates auf X.509-Short Lived Certificates umgestellt werden muss.

2 Grundlagen

In diesem Kapitel werden die grundlegenden Begriffe und Technologien erläutert, die für die Thematik dieser Bachelorarbeit notwendig sind. Zuerst wird das Grid Computing vorgestellt, da es von zentraler Bedeutung in dieser Ausarbeitung ist. Grundlegend werden die Begriffe Authentifizierung und Autorisierung geklärt, es werden auch einige wichtige Konzepte aus beiden Themengebieten dargelegt. In Grid-Systemen spielen X.509-Zertifikate eine entscheidende Rolle. Damit die Hintergründe von X.509-Zertifikaten erkennbar werden, wird auf das Design und die Arbeitsweise einer Public Key Infrastruktur (PKI) detailliert eingegangen.

2.1 Grid Computing

Grid-Systeme haben ihren Ursprung im wissenschaftlichen Bereich, dort sollten die Grid-System die Zusammenarbeit von Wissenschaftlern erleichtern. Das Technologische Vorbild des Grid Computing ist die Technologie des „power grid“ (Stromnetz), deshalb kann man zwischen den beiden Technologien einige Parallelen erkennen. Wenn jemand ein elektrisches Gerät mit Strom versorgen will, steckt er den Stromstecker des Gerätes in die Steckdose und das Gerät funktioniert. Der Benutzer muss nicht wissen, an welchem Ort sich das Kraftwerk befindet und wie der Strom erzeugt wird. Er muss sich nur entscheiden von wem er den Strom beziehen will. In einem Grid-System ist das vergleichbar. Ein Benutzer meldet sich an dem Grid-System an und greift auf eine Grid-Ressource zu. Der Benutzer muss nur entscheiden von welchem Ressourcen-Anbieter er eine Ressource beziehen möchte. An welchem geographischen Ort sich die Ressource befindet ist unerheblich für den Benutzer. Wie die Daten zwischen den Grid-Ressourcen ausgetauscht werden, muss der Benutzer ebenfalls nicht wissen, da dies von der Grid-Middleware realisiert wird. In den Anfängen der Grid-Technologie definierten Ian Foster und Carl Kesslemann in dem Buch „The Grid - Blueprint for a New Computing Infrastructure“ Computational Grid-Systeme wie folgt:

> „A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.“ (siehe [Foste-1998a] Seite 18)

Die Computational Grid-Systeme sind eine Art von Grid-Systemen, diese stellen Rechenleistung zur Verfügung. Eine allgemeinere Definition von Grid-Systemen beschrieb Ian Foster in der Veröffentlichung „The Anatomy of the Grid“ [Foste-2001]. Diese Definition stellt das zugrundeliegende Problem da, dass Grid-Systeme lösen und wird wie folgt beschrieben:

> „coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations.“ (siehe [Foste-2001] Seite 2)

Ein Grid-System stellt Ressourcen bereit, diese Ressourcen können Rechenleistung, Daten, Sensoren oder Dienste sein. Diese Ressourcen sollen für die gemeinsame Lösung von Problemen größeren Personengruppen zur Verfügung gestellt werden.

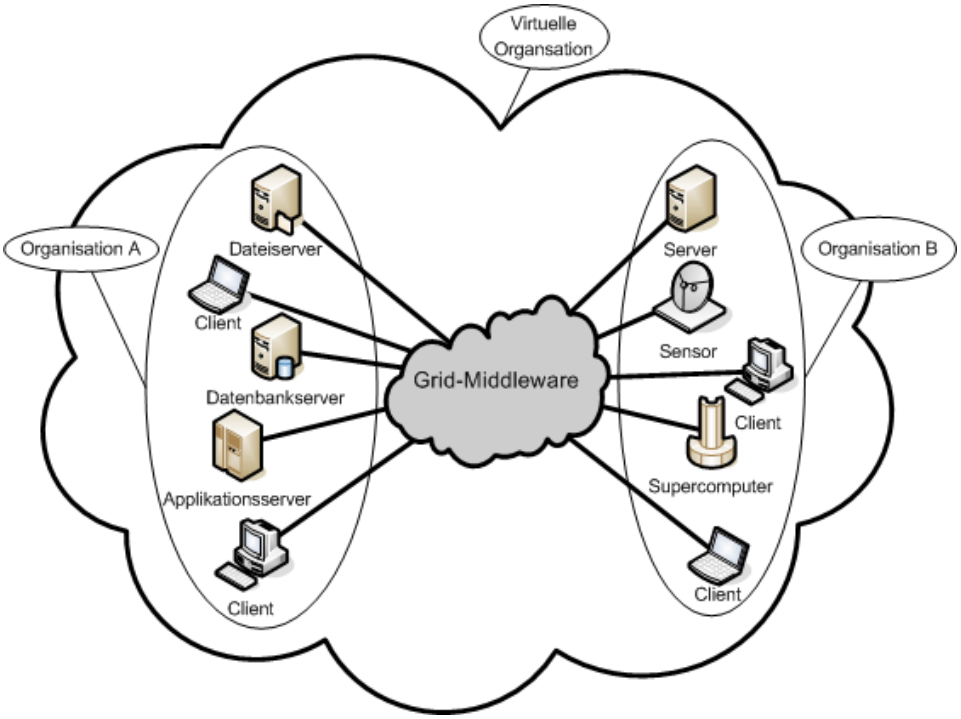

Abbildung 1: Aufbau einer virtuellen Organisation.

Die Ressourcen und die Benutzer eines Grid-Systems können zu unterschiedlichen Organisationen/Unternehmen gehören. Die Benutzer des Grid-Systems und die Grid-Ressourcen bilden zusammen eine virtuelle Organisation (VO). In der Abbildung 1 wird dies verdeutlicht. In einer virtuellen Organisation müssen Richtlinien (Policy) festgelegt werden, die einige sicherheitsrelevante Fragen klären. Wer darf Ressourcen bereitstellen? Welche Ressourcen werden bereitgestellt? Wer darf unter welchen Bedingungen auf eine Ressource zugreifen? Dies sind sehr wichtige Aspekte eines Grid-Systems. Virtuelle Organisationen können sich gegenseitig Ressourcen bereitstellen und damit ein größeres Grid-System bilden. Ein großer Vorteil von Grid-Systemen besteht darin, das diese keine besondere Hardware benötigen. Es kann jede Standard-Computerhardware für Grid-Ressourcen verwendet werden. Für die Kommunikation zwischen den einzelnen Ressourcen eines Grid-Systems werden auch keine speziellen Protokolle verwendet, es werden die in der Netzwerkkommunikation standardmäßig verwendeten Protokolle eingesetzt, wie z. B. das Simple Object Access Protocol (SOAP), TCP/IP. Auf den Grid-Ressourcen des Grid-Systems wird eine Grid-Middleware installiert. Die Grid-Middleware realisiert die Kommunikation zwischen

den Grid-Ressourcen und verwaltet die Grid-Ressourcen. Die Sicherheitsrichtlinien werden von der Grid-Middleware umgesetzt. Es werden Funktionen für das Überwachen (monitoring) der Ressourcen und das Auffinden (discovery) der anderen Grid-Ressourcen bereitgestellt. Die Grid-Middleware soll die komplexe Struktur eines Grid-Systems vor dem Benutzer verbergen. Der Benutzer eines Grid-Systems muss nicht wissen, wo die gewünschte Ressource steht. Der Benutzer verbindet sich zum Grid-System, authentifiziert sich und fordert eine gewünschte Ressource, wie z. B. Rechenleistung oder Daten an. Dass der Benutzer die Rechenleistung oder die gewünschten Daten bekommt, gewährleistet die Grid-Middleware. (vgl. [Jacob-2005] Seite 3 - 6, 10, 11; [Foste-2001] Seite 1 - 6; [Foste-1998a] Seite 18)

Das Open Grid Forum spezifizierte einen Standard für serviceorientierte Grid-Systeme, den Standard Open Grid Services Architecture (OGSA). Dieser Standard beschreibt Grid-Systeme auf der Basis der Konzepte und der Technologien von Webservices. Es wird eine Referenzarchitektur und die Anforderungen an Grid-Systeme spezifiziert. Die Aufgaben einer Grid-Middleware werden festgehalten. Die Grid-Middleware Globus Toolkit wurde ab der Version 3.0 auf der Basis der Standards Open Grid Services Architecture entwickelt, da einige Mitglieder der Globus Alliance an der Entwicklung des Standards beteiligt waren. Der Standard Web Service Resource Framework (WSRF) wurde von der Globus Allianz und IBM entwickelt. Veröffentlicht wird dieser Standard von OASIS. Der Standard WSRF spezifiziert einige Schnittstellen und Operationen, die einen Webservice zustandsbehaftet werden lassen. Ein zustandsbehafteter Webservice kann Daten zwischen zwei Aufrufen speichern, damit kann ein solcher Webservice komplexere Funktionen erfüllen. (vgl. [Jacob-2005] Seite 46, 47; [Foste-2006])

Im Laufe der Zeit wurden eine Reihe von Grid-Middleware-Systemen entwickelt, die Bekanntesten sind die folgenden Drei:

- **Globus Toolkit:** Das Globus Projekt wurde 1995 von dem U.S. Argonne National Laboratory, dem University of Southern California's Information Sciences Institute (ISI) und der University of Chicago (UofC) gegründet. Im September 2003 wurde aus dem Globus Projekt die Globus Allianz. Das Globus Toolkit war die erste Grid-Middleware und gilt heute als „de facto Standard" (siehe [Foste-2004] Seite 45) für Grid-Systeme. Das Globus Toolkit wurde so entwickelt, dass es auf Standard Hardware und mit Standard Protokollen arbeitet. Zusätzliche Informationen über das Globus Toolkit kann man der Dokumentation auf der Webseite der Globus Allianz entnehmen [GlobA-2008a].

- **UNICORE:** (Uniform Interface to Computing Resources) Das Projekt für die Entwicklung von UNICORE wurde vom Bundesministerium für Bildung und Forschung (BMBF) finanziert. Dieses Projekt startete 1997, es beteiligten sich eini-

ge deutsche Forschungszentren, wie das Forschungszentrum Jülich und einige Unternehmen an der Entwicklung. UNICORE wurde entwickelt um den Benutzern der deutschen Supercomputer-Zentren einen nahtlosen, sicheren und intuitiven Zugriff auf ihre heterogenen Ressourcen zu ermöglichen. Weitere Details über UNICORE enthält die Dokumentation auf der Webseite des UNICORE Forums [Unico-2008].

- **gLite:** Die Grid-Middleware gLite entstand im Rahmen des Projekt Enabling Grids for E-sciencE (EGEE), das im April 2004 begann. An der Entwicklung von gLite arbeiteten eine Reihe von europäischen Forschungseinrichtungen, wie z. B. „The European Organization for Nuclear Research (CERN)“. Das Projekt EGEE wurde von der Europäischen Union finanziert und hat das Ziel eine Service-Grid-Infrastruktur zu entwickeln, die den Wissenschaftlern 24-Stunden am Tag zur Verfügung steht. Weitere Informationen über gLite kann man in der Dokumentation auf der gLite Webseite nachlesen [gLite-2008].

2.2 Authentifizierung

Bei der Authentifizierung wird die Identität einer Person oder eines Computer-Systems überprüft. Die zu identifizierende Person oder Objekt gibt ihre Identität an, diese Identität muss mittels einer geeigneten Methode von der Person oder dem Objekt nachgewiesen werden. Die angegebene Identität und der dazugehörige Nachweis wird auch als Credentials bezeichnet. Zum Beispiel wird der Benutzername zusammen mit dem Passwort als Credential bezeichnet. Zum Nachweis der Identität haben sich drei Methoden durchgesetzt (vgl. [Ecker-2008] Seite 431):

- Authentifizierung durch Wissen
- Authentifizierung durch einen persönlichen Besitz
- Authentifizierung durch ein biometrisches Merkmal

Unter der **Authentifizierung durch Wissen** versteht man die Authentifizierung durch einen Benutzernamen und des dazu gehörigen Passworts, dies ist das meist verbreitetste Authentifizierungs-Verfahren. Bei diesem Verfahren muss das System sicherstellen, das die gespeicherten Passwörter nicht von Unbefugten ausgelesen werden können. Die meisten Systeme speichern deshalb nur einen kryptografischen Hashwert des Passwortes zusammen mit dem Benutzernamen ab. Dies hat den Vorteil, dass das Passwort nicht ohne größeren Aufwand wiederhergestellt werden kann. Das Challenge-Response-Verfahren ist ein sehr wichtiges Verfahren für die Authentifizierung durch Wissen, da bei diesem Verfahren die Korrektheit des Wissens (Passwort) überprüft werden kann, ohne das Wissen zu übertragen. Dies hat den Vorteil,

dass ein Angreifer das benötigt Wissen bei der Übertragung nicht abfangen kann und sich deshalb nicht mit der vorgetäuschten Identität erfolgreich authentifizieren kann. Bei diesem Verfahren müssen die Beteiligten denselben Verschlüsselungsalgorithmus verwenden und einen gemeinsamen Schlüssel haben, dieser kann zum Beispiel aus dem Passwort abgeleitet werden. Der Client sendet seine Identifikation an den Server. Darauf sendet er Server dem Client eine Zufallszahl, die sogenannte Challenge. Die Challenge beantwortet der Client mit der verschlüsselten Zufallszahl, dies ist der Response. Der Response wird vom Client mit dem festgelegten Verschlüsselungsalgorithmus, der Zufallszahl und dem gemeinsamen Schlüssel erzeugt. Der Server kann den erhaltenen Response mit dem selbst erzeugten Response vergleichen. Stimmen beide Response überein, hat sich der Client erfolgreich authentifiziert. [Ecker-2008] Seite 451 - 455)

Bei der **Authentifizierung durch einen persönlichen Besitz** wird ein Gegenstand als Nachweis der Identität verwendet. In der Praxis wird meistens eine Smartcard verwendet. Für die Authentifizierung wird ein gemeinsamer privater Schlüssel (Pre-Shared-Secret) benötigt, der auf der Smartcard abgespeichert ist und auch dem Authentifizierungsserver bekannt ist. Die Authentifizierung mittels einer Smartcard wird in drei Schritte unterteilt. Im ersten Schritt muss sich der Besitzer an der Smartcard mit seiner PIN authentifizieren. Meistens gibt der Benutzer seine PIN auf einem Tastenfeld am Kartenlesegerät ein, dass die PIN direkt an die Smartcard weiterleitet und die Smartcard überprüft, ob die richtige PIN eingegeben wurde. Nach dem der Benutzer sich erfolgreich an der Smartcard authentifiziert hat, beginnt der zweite Schritt, bei diesem authentifiziert sich die Smartcard am Zielsystem. Dazu wird das Challenge-Response-Verfahren eingesetzt, das Zielsystem übermittelt eine Zufallszahl (Challenge) an die Smartcard. Die Smartcard antwortet auf die Challenge mit der verschlüsselten Zufallszahl, die mit dem gemeinsamen privaten Schlüssel verschlüsselt wurde. Beim dritten Schritt authentifiziert sich das Zielsystem an der Smartcard. Dabei wird ebenfalls das Challenge-Response-Verfahren verwendet, nur dieses Mal stellt die Smartcard die Challenge an das Zielsystem. (vgl. [Ecker-2008] Seite 467 - 472)

Die **Authentifizierung durch ein biometrisches Merkmal** umfasst die Authentifizierung einer Person durch Körpermerkmale. Diese Methode wird in zwei Kategorien unterteilt, zum einen die physiologischen Merkmale und zum anderen die verhaltenstypischen Merkmale. Zu den physiologischen Merkmalen gehören die Fingerabdrücke, Iris, Retina oder auch das Gesicht. Für diese Merkmale werden Scanner benötigt, wie den Fingerabdruckscanner oder ein Irisscanner. Bei den verhaltenstypischen Merkmalen gibt es das spezifische Tippverhalten, die Stimme oder die Dynamik einer handschriftlichen Unterschrift. (vgl. [Ecker-2008] Seite 481 - 484)

Eine besondere Form des Authentifizierens stellt das **Single SignOn (SSO)** da. Beim Single SignOn meldet der Benutzer sich einmal an einem Authentifizierungs-

system an und erhält dadurch Zugriff auf alle Dienste, die das System verwaltet bzw. schützt und der Benutzer die benötigten Berechtigungen hat. Der Benutzer muss bei einem solchen System nur einmal mit dem Authentifizierungssystem interagieren, dies erleichtert das Arbeiten in verteilten Systemen erheblich. (vgl. [Ecker-2008] Seite 499)

2.3 Autorisierung

Die Autorisierung ist die Gewährung eines Zugriffs auf eine geschützte Ressource. Bevor die Autorisierung stattfinden kann, muss sich ein Benutzer erfolgreich am System authentifizieren. Für eine zuverlässige Autorisierung ist eine Zugriffskontrolle unumgänglich. Bei der Zugriffskontrolle wird geprüft ob ein Benutzer, der auf eine Ressource zugreifen möchte, die benötigten Berechtigungen hat. Den unterschiedlichen Zugriffskontrollmechanismen liegen zwei Kategorien von Sicherheitsmodellen zugrunde. Die Informationsflussmodelle und die Zugriffskontrollmodelle. Bei Informationsflussmodellen werden die Informationskanäle zwischen den Objekten (Prozesse, Dateien, usw.) definiert. Bei den Zugriffskontrollmodellen werden die Zugriffe auf die Objekte (Prozesse, Dateien, usw.) definiert. In der Praxis haben sich die Zugriffskontrollmodelle durchgesetzt, diese werden in drei Kategorien unterteilt (vgl. [Ecker-2008] Seite 232 - 233):

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role Based Access Control (RBAC)

Die Grundidee der **Discretionary Access Control** (benutzerbestimmbare Zugriffskontrolle) ist das Eigentümerprinzip. Der Eigentümer eines Objekts verwaltet die Zugriffsrechte für das Objekt. Für jedes Objekt müssen die Zugriffsrechte explizit gesetzt werden, es gibt keine Möglichkeit Zugriffsrechte systemweit zuvergeben.

Die **Mandatory Access Control** (systembestimmte oder regelbasierte Zugriffskontrolle) übersetzt. Bei diesem Modell werden Zugriffsrechte systemweit vergeben. Die systembestimmten Zugriffsrechte werden als wichtiger als die benutzerbestimmten Zugriffsrechte angesehen. Allerdings ist es möglich durch benutzerbestimmte Zugriffsrechte, die systembestimmten Zugriffsrechte weiter einzuschränken. Damit können sich folgenden beiden Szenarien ergeben. Wenn der Benutzer A explizit dem Benutzer B den Zugriff auf das Objekt O gestattet aber die systembestimmten Zugriffsrechte diesen Zugriff verweigern, erhält Benutzer B keinen Zugriff auf das Objekt O. Wenn ein Benutzer A einem Benutzer B den Zugriff auf das Objekt O verweigert,

obwohl dieser durch die systembestimmten Zugriffsrechte auf Objekt O zugreifen darf, kann Benutzer B nicht auf Objekt O zugreifen.

Bei der **Role Based Access Control** (rollenbasierte Zugriffskontrolle) wird auf eine Aufgaben spezifische Rechtevergabe gesetzt. Es werden Zugriffsrechte und Rollen definiert. Den Rollen werden die für die jeweilige Aufgabe benötigten Rechte zugeteilt. Die Benutzer erhalten über Rollenmitgliedschaften die Rechte, die sie zum Ausführen einer Aufgabe benötigen.

2.4 Security Assertion Markup Language

Die Security Assertion Markup Language (SAML) beschreibt ein XML-basierendes Protokoll, dass für die Authentifizierung und die Autorisierung in verteilten Systemen eingesetzt werden kann. Entwickelt wurde die Security Assertion Markup Language von dem Security Services Technical Committee der Organization for the Advancement of Structured Information Standards (OASIS). Im November des Jahres 2002 wurde die SAML in der Version 1.0 ein Standard der OASIS, aktuell ist die Version 2.0. Die wichtigsten Punkte des SAML-Standards werden in den folgenden Punkten erläutert.

In der Spezifikation **SAML-Core** wird der Aufbau und die Bedeutung der SAML-Assertions definiert. Der Aufbau der SAML-Nachrichten des Protokolls wird ebenfalls beschreiben. Die SAML-Assertions sind Pakete, die die Informationen über Benutzer enthalten. Die Benutzerinformationen, wie z. B. Attribute, sind in SAML-Statements enthalten. Die SAML-Assertions werden zum Transport in Kommunikationsprotokolle eingebunden, wie z. B. das Hypertext Transfer Protocol (HTTP) oder das Simple Object Access Protocol (SOAP). Die SAML-Assertions können auch in andere Objekte, wie z. B. X.509-Zertifikate, eingebunden werden.

Die **SAML-Profile** beschreiben unterschiedliche Kommunikationsszenarien, wie Single SignOn oder den Attributaustausch. Diese Profile beschreiben die Verwendung der SAML-Assertions und definieren das Request-Response-Verhalten des Protokolls. In den Profilen werden Rollen beschrieben, die die beteiligten Systeme während der Kommunikation einnehmen. Die Aufgabe der Rolle wird spezifiziert und welche SAML-Nachrichten für die Erfüllung dieser Aufgaben verwendet werden. Es wird auch festgelegt, welches Kommunikationsprotokoll verwendet wird und wie die Assertions in dieses eingebunden bzw. wieder extrahiert werden.

Mit der Hilfe der **SAML-Metadaten** kann man die Systeme beschreiben, die mittels der Security Assertion Markup Language kommunizieren. Damit die Systeme miteinander kommunizieren können, müssen diese einige Informationen austauschen und Vereinbarungen treffen. Die Informationen und die Vereinbarungen werden in einer Metadaten-Datei festgehalten, die auf der Spezifikation der SAML-Metadaten basiert.

Innerhalb der SAML-Kommunikation müssen für die Systeme eindeutige IDs vergeben werden und deren Rolle muss festgehalten werden. Die Rolle wird dem entsprechenden SAML-Profile entnommen, dass die Kommunikationsbeziehung beschreibt. Die X.509 Zertifikate der Systeme müssen in der Metadaten-Datei festgehalten werden, damit eine eindeutige Identifizierung der Systeme gewährleistet werden kann und auch Systeme mit gefälschten Identitäten erkannt werden können.

Einige Frameworks, die dem SAML-Standard implementieren definieren Entwürfe (Drafts), die den SAML-Standard um benötigte SAML-Profile oder die SAML-Metadaten erweitern. Diese Entwürfe werden von den Framework Herstellern bei OASIS eingereicht und die Frameworks implementieren diese auch, aber es ist nicht sichergestellt, dass die Entwürfe zu einem OASIS-Standard werden.

2.5 Public Key Infrastruktur

Im Rahmen von asymmetrischen kryptografischen Verfahren und digitalen Signaturen werden digitale Schlüsselpaare benötigt. Diese bestehen aus einem privaten und einem öffentlichen Schlüssel. Damit die asymmetrischen kryptografischen Verfahren verwendet werden können, müssen die öffentlichen Schlüssel ausgetauscht werden, dazu werden diese in digitalen Zertifikaten eingebettet. In der Praxis haben sich X.509-Zertifikate durchgesetzt. Ein Beispiel für die Verwendung von digitalen Schlüsselpaaren ist die Verschlüsselung einer Kommunikation. Der Sender verschlüsselt eine Nachricht mit dem öffentlichen Schlüssel des Empfängers. Dem Empfänger wird die verschlüsselte Nachricht zu gesendet. Dieser kann die Nachricht mit seinem privaten Schlüssel entschlüsseln. Für die Erzeugung und Verwaltung von digitalen Zertifikaten wird die Public Key Infrastruktur eingesetzt. In Grid-Systemen werden X.509-konforme Proxy-Zertifikate zur Authentifizierung der Benutzer eingesetzt.

2.5.1 X.509-Zertifikate

Ein digitales Zertifikat ordnet einen öffentlichen Schlüssel einer natürlichen oder einer juristischen Person zu. Es erfolgt nur eine Bestätigung dieser Zuordnung, es werden keine Aussagen zur Vertrauenswürdigkeit einer Person gemacht. Die Feststellung der Vertrauenwürdigkeit einer Person kann nicht durch PKI-Systeme erfolgen, diese muss der Kommunikationspartner selbst festlegen. In der Praxis haben sich die digitalen Zertifikate nach dem Standard X.509 durchgesetzt, diese werden in dem RFC 3280 [Housl-2002] spezifiziert. Wenn ein X.509-Zertifikat einer bestimmten Person/Benutzer zugeordnet ist, wird dieses auch „End Entity Certificate“ (EEC) genannt. Ein X.509-Zertifikat hat folgenden Aufbau:

- **Version** (Versionsnummer): Nach welcher Zertifikatsversion dieses X.509-Zertifikat aufgebaut ist. Diese ist notwendig, damit der Standard an neue Anforderungen angepasst werden kann, aber auch ältere Zertifikate noch verwendet werden können.
- **Serial Number** (Seriennummer): Jedes X.509-Zertifikat hat eine eindeutige Seriennummer, dafür muss die ausstellende Certification Authority sorgen. Damit werden die X.509-Zertifikate eindeutig identifiziert und in der Certificate Revocation List wird nur diese Seriennummer eingetragen, wenn das X.509-Zertifikat ungültig wird.
- **Signature Algorithm** (Signaturinformationen): In diesem Feld wird der Signatur Algorithmus festgehalten, der zum Signieren des X.509-Zertifikates verwendet wurde.
- **Issuer** (Zertifikataussteller): Der Name des Zertifikatausstellers, dieser ist in der Form eines Distinguished Name (DN) festgehalten.
- **Validity** (Gültigkeitsdauer): Der Zeitraum in dem das X.509-Zertifikat gültig ist. Es gibt ein Startdatum, ab wann das X.509-Zertifikat gültig ist und ein Datum, bis wann das X.509-Zertifikat gültig ist. In der Regel sind die X.509-Zertifikate zwei Jahre gültig. Dieses Feld zeigt, dass es wichtig ist, dass alle Computer-Systeme zeitlich synchronisieren sind, die mit X.509-Zertifikaten arbeiten. Da es sonst passieren kann, dass ein X.509-Zertifikat als ungültig erkannt wird, nur weil der Gültigkeitszeitraum auf dem Ziel-System noch nicht begonnen hat, obwohl auf dem System des Absenders das X.509-Zertifikat schon gültig ist.
- **Subject** (Benutzername): Der Name des Zertifikatinhabers, dieser ist in der Form eines Distinguished Name (DN) festgehalten.
- **Subject Public Key Info** (Schlüsselinformationen): Der Algorithmus, der zur Erzeugung des Schlüsselpaares des Zertifikatinhabers verwendet wurde und die Schlüsselstärke, z. B. 1024 Bit. Es wird auch der öffentliche Schlüssel des Benutzers gespeichert.
- **ID** (eindeutiger Identifikator): In der Version 2 und 3 können optional noch eindeutige IDs für den Zertifikatinhaber und den Zertifikatersteller eingefügt werden.
- **Extensions** (Erweiterungen): In der Version 2 und 3 gibt es die Erweiterungen, diese können dafür verwendet werden noch zusätzliche Informationen in den

X.509-Zertifikaten zuspeichern. Es gibt zwei Arten von Erweiterungen, die unkritischen und die kritischen Erweiterungen. Die unkritischen Erweiterungen können von Programmen ignoriert werden, wenn das Programm diese Erweiterung nicht verarbeiten bzw. interpretieren kann. Wenn ein Programm eine kritische Erweiterung einliest, diese aber nicht verarbeiten bzw. interpretieren kann, muss diese X.509-Zertifikat als ungültig betrachtet werden. Zu den bekanntesten Erweiterungen gehören die Erweiterungen „key usage“ und „certificate policies“. Mit der Erweiterung „key usage“ kann festgelegt werden, für welchen Zweck der Schlüssel eingesetzt werden darf, z. B. Signieren, Verschlüsseln. Unter welchen Bedingungen das X.509-Zertifikat erzeugt wurde und für welchen Zweck dieses eingesetzt werden kann, wird in der Erweiterung „certificate policies“ festgehalten. Alle Erweiterungen sind optional in einem X.509-Zertifikat.

- **Signature Algorithm** (Signatur): Zuletzt folgt die Signatur des X.509-Zertifikats. In der Signatur ist ein Hashwert des Zertifikats verschlüsselt gespeichert, diesen benötigt der Empfänger bei der Validierung des X.509-Zertifikats.

Im Anhang A auf Seite 65 befindet sich ein Beispiel eines X.509-Zertifikats, dieses Beispiel enthält auch eine unkritische Erweiterung mit einer SAML-Assertion. Die digitalen Zertifikate werden von einer Public Key Infrastruktur erzeugt und verwaltet. (vgl. [Ecker-2008] Seite 379 - 381,) In Grid-Systemen werden **Proxy-Zertifikate** verwendet, diese sind Spezielle X.509-Zertifikate. Die Proxy-Zertifikate werden zur Authentifizierung der Benutzer und zur Delegierung der Benutzer-Credentials verwendet. Mit einem Proxy-Zertifikat kann ein Benutzer seine Identität auf einen Dienst übertragen, damit dieser in seinem Namen Aufgaben erledigen kann. In dem RFC 3820 [Tueck-2004] werden die Proxy-Zertifikate spezifiziert. Es gibt einige Eigenschaften, die ein X.509-Zertifikat zu einem Proxy-Zertifikat werden lassen. Im folgenden Text werden diese Eigenschaften erläutert. Der Issuer (Zertifikataussteller) ist ein DN aus einem „End Entity Certificate“ des Benutzers oder ein DN aus einem anderen Proxy-Zertifikat des Benutzers. Mit dem privaten Schlüssel des Zertifikats, das im Issuer-Feld eingetragen ist, wird das Proxy-Zertifikat signiert. Der Subject (Benutzername) ist der DN aus dem Issuer-Feld mit einem zusätzlichen „Common Name“ (CN), der den Subjekt eindeutig macht. Dieser Zusatz darf in keinem weiteren Proxy-Zertifikat eingesetzt werden. Für Proxy-Zertifikate ist eine kritische Erweiterung spezifiziert worden. Die Erweiterung ProxyCertInfo legt fest, dass dies Zertifikat ein Proxy-Zertifikat ist und ob Einschränkungen für dieses Proxy-Zertifikat festgelegt wurden. Das Feld „pCPathLenConstraint“ schränkt die Pfadtiefe der Proxy-Zertifikate ein, wie oft dieses Proxy-Zertifikat zur Signierung von anderen Proxy-Zertifikaten eingesetzt werden darf. Die Erweiterung „proxyPolicy“ definiert eine Richtlinie, wie dieses Proxy-Zertifikat eingesetzt werden darf. Diese Erweiterung enthält einen Richtlinienausdruck und

in welcher Sprache dieser Ausdruck verfasst wurde. Ein weiterer wichtiger Punkt ist, dass die Proxy-Zertifikate nur eine relativ kurze Gültigkeit haben, von nur wenigen Stunden. (vgl. [Tueck-2004])

2.5.2 Aufbau einer Public Key Infrastruktur

Eine Public Key Infrastruktur (PKI) erzeugt und verwaltet digitale Zertifikate (X.509-Zertifikate). Bei der Verwendung von asymmetrischen Kryptographischenverfahren haben sich PKIs durchgesetzt. Der wichtigste Bestandteil einer Public Key Infrastruktur ist die **Certification Authority** (CA), diese erstellt die X.509-Zertifikate. Eine Certification Authority kann auch einen **Zeitstempeldienst** unterhalten. Dieser wird benötigt, wenn Daten mit einem Zeitpunkt vertrauenswürdig verknüpft werden müssen. Die **Registration Authority** (RA) bürgt für die Verbindung zwischen einem öffentlichen Schlüssel und einer Identität. Die Certification Authority und Registration Authority arbeiten nach den Regeln des **Certification Practice Statement** (CPS), dies ist eine Richtlinie für die Erstellung der Zertifikate. Die Richtlinie beschreibt die Art der Authentifizierung der Zertifikatsinhaber und auch wie die CA geschützt wird. Welche Richtlinien bei der Erstellung eines Zertifikats beachtet werden und wie das Zertifikat verwaltet wird, ist in dem Certification Practice Statement festgehalten. Ein Trust Center besteht aus einer Certification Authority und Registration Authority. Alle X.509-Zertifikate die vor Ende ihrer Gültigkeit ungültig geworden sind, z. B. wenn der dazugehörige private Schlüssel des Benutzers bekannt geworden ist, müssen in einer **Certificate Revocation List** (CRL) aufgeführt werden. In der Certificate Revocation List wird die ID der ungültigen X.509-Zertifikate gespeichert. Die Certificate Revocation List wird in einem öffentlich-zugänglichen Verzeichnis bereitgestellt, damit jeder die Möglichkeit hat ungültige X.509-Zertifikate nicht nur an dem abgelaufenen Gültigkeitsdatum oder der falschen Signatur zuerkennen, sondern auch überprüfen kann, ob das Zertifikat zurückgezogen worden ist. Die Implementierung des Online Certificate Status Protocol (OCSP) vereinfacht die Überprüfung der Certificate Revocation List. Wie die Certificate Revocation List werden auch alle gültigen Zertifikate in einem Verzeichnis bereitgestellt, damit alle Benutzer einer Public Key Infrastruktur auf die öffentlichen Schlüssel der anderen Benutzer zugreifen können. (vgl. [Ecker-2008] Seite 381 - 389)

2.5.3 Erstellen von Zertifikaten

Bevor ein X.509-Zertifikat erstellt werden kann, muss sich eine Identität authentifizieren. Eine natürliche Person kann sich durch die Vorlage ihres Personalausweises authentifizieren. Eine juristische Person muss durch einen berechtigten Vertreter, der

juristischen Person authentifiziert werden. Die Daten der Identität werden in der Registration Authority gespeichert. Im nächsten Schritt muss der Antragsteller oder die Zertifizierungsstelle ein Schlüsselpaar generieren. Das Schlüsselpaar besteht aus einem öffentlichen Schlüssel und einem privaten Schlüssel. Der öffentliche Schlüssel wird den Kommunikationspartnern bereitgestellt. Der private Schlüssel darf unter keinen Umständen bekannt werden, sonst können sich andere Personen für die Person ausgeben, der der Schlüssel zugeordnet ist. Aus diesem Grund sollte das Schlüsselpaar in einer speziell gesicherten Umgebung im Trust Center erzeugt werden, dort sollte sichergestellt werden das der private Schlüssel nicht unbemerkt eingesehen oder kopiert werden kann. Im nächsten Schritt wird das X.509-Zertifikat mit den Daten der Identität und den Daten des öffentlichen Schlüssels der Identität durch das Trust Center erzeugt. Jetzt fehlt auf dem X.509-Zertifikat noch eine digitale Signatur des Trust Centers. Für das signieren der X.509-Zertifikate besitzt die Certification Authority einen privaten Schlüssel. Dieser wird in einem asymmetrischen Signaturverfahren eingesetzt um das X.509-Zertifikat zusignieren. Mit der Signatur bestätigt die Certification Authority die Echtheit des X.509-Zertifikats. Die Signatur stellt sicher das Manipulationen an dem X.509-Zertifikat erkannt werden und dieses X.509-Zertifikat verworfen wird. Der private Schlüssel der CA muss strikt vor unbefugtem Zugriff geschützt werden, da sonst die Integrität der Certification Authority verloren geht. Jeder der den privaten Schlüssel, der CA erhält, könnte im Namen der CA X.509-Zertifikate erstellen, bei denen die Identität des Zertifikatsinhabers gefälscht ist. Im letzten Schritt wird das X.509-Zertifikat in das Zertifikatsverzeichnis übermittelt und dem Benutzer werden alle Daten auf einem sicheren Weg übergeben. Am besten werden der öffentliche Schlüssel, der private Schlüssel und das Zertifikat auf einer Smartcard dem Benutzer persönlich übergeben. Auf einer Smartcard ist der private Schlüssel verhältnismäßig gut vor unberechtigtem Zugriff geschützt. Wenn man den privaten Schlüssel auf einer Festplatte speichert, sollte dieser verschlüsselt und durch ein Passwort geschützt abgelegt werden. (vgl. [Ecker-2008] Seite 381 - 385)

2.5.4 Validierung von Zertifikaten

Für die Validierung von X.509-Zertifikat benötigt man den öffentlichen Schlüssel des Zertifikaterstellers, den öffentlichen Schlüssel der Certification Authority. Bei der Validierung wird ein Hashwert vom X.509-Zertifikat erzeugt und der auf dem X.509-Zertifikat gespeichert Hashwert mit dem öffentlichen Schlüssel, der ausstellenden CA entschlüsselt. Wenn beide Hashwerte übereinstimmen und das X.509-Zertifikat noch nicht abgelaufen ist, wird das X.509-Zertifikat positiv validiert. Es gibt keine weltweite Vernetzung aller Certification Authoritys durch Vertrauensbeziehungen. Meistens gibt es eine Gruppe von mehreren CAs, die sich gegenseitig Vertrauen und ein „Web of

Trust“ erzeugen. Eine weltweite Vernetzung aller CAs würde gegen das Konzept von PKI-Systemen verstoßen. Da bei einer solchen Vernetzung alle CAs sich gegenseitig vertrauen würden, dies aber nicht im Sinne des Benutzers wäre. Ein „Web of Trust“ (Vertrauensnetz) besitzt eine Root-CA. Die Root-CA vergibt die X.509-Zertifikate der CAs der nächsten hierarschichen Ebene, diese CAs können wiederum weiter Zertifikate für CAs ausstellen. Alle CAs, die eine gemeinsame Root-CA haben, gehören zu einem „Web of Trust“. Zwischen solchen „Web of Trusts“ können mittels Cross-Zertifizierung eine Vertrauensbeziehung aufgebaut werden. Ohne solche Vertrauensbeziehungen ist eine gegenseitige Validierung von X.509-Zertifikaten nicht möglich. Bei der Validierung von digitalen Zertifikaten muss man zwei Szenarien unterscheiden. Wenn zwei Identitäten miteinander kommunizieren wollen, die ihre X.509-Zertifikate von derselben CA beziehen und wenn die beiden Identitäten ihre X.509-Zertifikate von unterschiedlichen CAs beziehen. Im ersten Szenario kennen beide Identitäten den öffentlichen Schlüssel ihrer Gemeinsamen CA und können damit die X.509-Zertifikate überprüfen. Wenn die Kommunikationspartner ihre X.509-Zertifikate von unterschiedlichen Certification Authority beziehen ist es komplizierter die X.509-Zertifikate zu validieren.

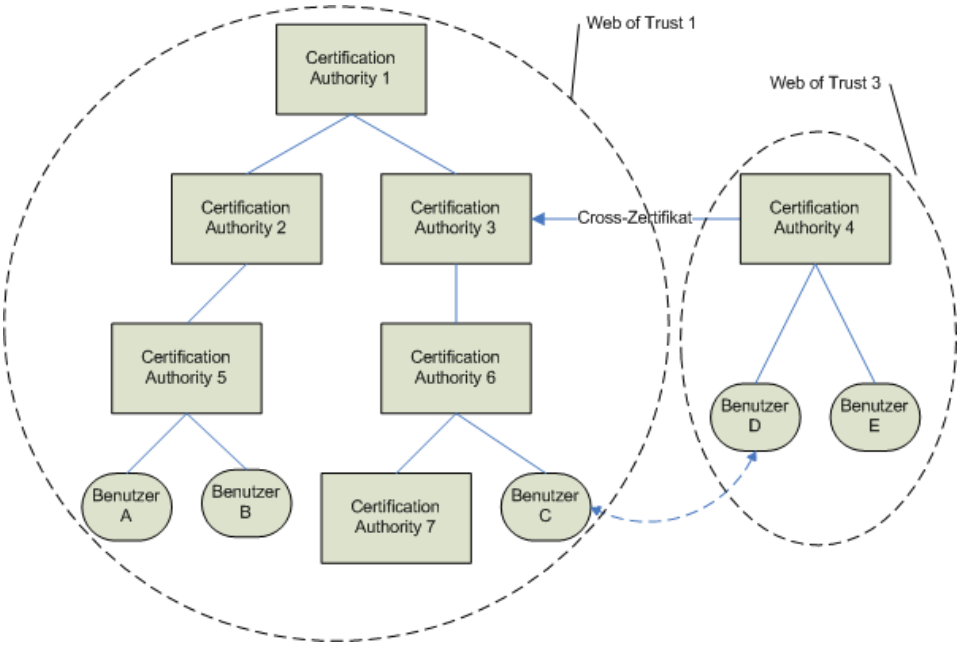

Abbildung 2: Validierung von digitalen Zertifikaten ([Ecker-2008] Seite 390).

In der Abbildung 2 ist ein solches Szenario dargestellt. In diesem Fall möchte Benutzer C, der sein X.509-Zertifikat von der CA 6 bezieht, mit Benutzer D kommunizieren, der sein X.509-Zertifikat von der CA 4 bezieht. Diese beiden Certification Authority liegen in unterschiedlichen „Web of Trust“. Damit eine Validierung der X.509-Zertifikate erfolgen kann, muss eine Cross-Zertifizierung zwischen den beiden „Web of Trust“ vorhanden sein, sonst schlägt die Validierung fehl und das Zertifikat wird abgelehnt.

Benutzer C schickt Benutzer D sein X.509-Zertifikat, damit dieser das X.509-Zertifikat validieren kann, benötigt er alle X.509-Zertifikate, die zwischen beiden liegen. In diesem Fall sind das die X.509-Zertifikate der CA 3, CA 4 und CA 6. Der Validierungsprozess beginnt mit der Überprüfung des Zertifikats der CA 4. Der Benutzer D kann das Zertifikat der CA 4 überprüfen, da er den öffentlichen Schlüssel dieser CA kennt, da es seine eigene Root-CA ist. Mit diesem öffentlichen Schlüssel kann man das X.509-Zertifikat der CA 3 validieren, da dieses Zertifikat mit dem Schlüssel der CA 4 cross-zertifiziert wurde. Bei der Cross-Zertifizierung wurde auch das X.509-Zertifikat der CA 1 benötigt, da diese das X.509-Zertifikat CA 3 erstellt hat. Mit dem öffentlichen Schlüssel der CA 3 kann das Zertifikat der CA 6 überprüft werden. Nun kann mit dem öffentlichen Schlüssel der CA 6, das übermittelte X.509-Zertifikat des Benutzers C validieren. Wie sich zeigt, ist die Validierung von X.509-Zertifikaten ist ein rekursiver Prozess. (vgl. [Ecker-2008] Seite 389 - 393)

3 Analyse

Bevor mit der Konzeption begonnen werden kann, müssen die Anforderungen an das Konzept erarbeitet werden. Neben den Anforderungen, die durch die Aufgabenstellung spezifiziert wurden, müssen auch die Anforderungen der beteiligten Komponenten beachtet werden. Es müssen die Anforderungen berücksichtigt werden, die Grid-Systeme an die Authentifizierung und Autorisierung stellen. Alle Prozesse in Grid-Systemen, bei denen eine Authentifizierung oder Autorisierung stattfindet, müssen identifiziert werden.

3.1 Anforderungen an Authentifizierung und Autorisierung in Grid-Systemen

Grid-Systeme stellen Ressourcen organisationsübergreifend bereit, diese müssen vor unberechtigtem Zugriff geschützt werden. Deshalb ist es notwendig einige Sicherheitsrichtlinien für Grid-Systeme aufzustellen, diese müssen dem Konzept von Grid-Systemen gerecht werden. Dabei ist vor allem der Anspruch der hohen Flexibilität von Grid-Systemen und die Verwendung von offen Standard Protokollen zubeachten. Nataraj Nagaratnam, et al. spezifizierten in der Veröffentlichung „The Security Architecture for Open Grid Services“ (vgl. [Nagar-2002] Seite 9 - 10) einige Sicherheitsanforderungen für Grid-Systeme. Folgende Punkte stellen die Anforderungen an Authentifizierung und Autorisierung da:

- **Authentifizierung:** Es soll eine Schnittstelle definiert werden, die von mehreren unterschiedliche Authentifizierungsmethoden verwendet werden kann. Die ver-

wendeten Methoden müssen bei jeder Authentifizierungsoperation anwendbar sein. Dabei kann es sich um Standardtechnologien oder auch um unternehmensspezifische Technologien handeln.

- **Delegierung:** Es werden Verfahrensweisen benötigt, die es ermöglichen Zugriffsrechte von einem Servicebenutzer auf einen Service zu delegieren, dieses sollte mittels Richtlinien (Policy) festgelegt werden. Wenn Zugriffsrechte von einer Identität auf eine andere Identität delegiert werden muss beachtet werden, dass diese Delegierung nur für einen Grid-Job und für einen begrenzten Zeitraum erfolgt. Dies ist notwendig um das Risiko eines Missbrauchs der Identität bzw. der Zugriffsrechte zu verhindern.

- **Single SignOn (SSO):** Nach dem sich ein Benutzer erfolgreich authentifiziert hat, muss sich dieser innerhalb eines bestimmten Zeitraums nicht ein weiteres Mal authentifizieren, wenn dieser auf eine weitere OGSA-Ressource zugreifen möchte. Dabei muss berücksichtigt werden, dass eine Serviceanfrage zwischen verschiedene Sicherheitsdomänen erfolgen kann. Deshalb sollten Föderationen zwischen den Sicherheitsdomänen gebildet werden, diese werden auch virtuellen Organisationen (VO) genannt.

- **Lebensdauer und Erneuerung von Credentials:** In einigen Fällen kann es sein, dass ein Grid-Job, der von einem Benutzer gestartet wurde, mehr Zeit in Anspruch nimmt, als die Lebensdauer der übergebenen Credentials zulässt. Deshalb muss es möglich sein den Benutzer vor dem Ablauf der Credentials zu informieren oder es muss die Möglichkeit gegeben die Credentials zu erneuern, um den Grid-Job fertigstellen zu können.

- **Autorisierung:** Es müssen Policies (Richtlinien) für die Zugriffskontrolle auf die einzelnen OGSA-Services erstellt werden. Der Benutzer soll die Möglichkeit haben selber festzulegen, welchen Serviceanbietern er vertraut. Verschiedene Zugriffskontrollmodelle sollen unterstützt werden.

- **Vertraulichkeit:** Die Vertraulichkeit der zugrundeliegenden Kommunikation ist durch geeignete Mechanismen sicherzustellen, damit die Vertraulichkeit von Nachrichten und Dokumenten in der OGSA-Infrastruktur gewährleistet werden kann. Die Vertraulichkeit muss bei Punkt-zu-Punkt-Verbindungen und bei Store-and-Forward-Mechanismen gewährleistet werden.

- **Nachrichten Integrität:** Der Empfänger muss die Möglichkeit haben manipulierte Nachrichten oder Dokumente zuerkennen. In den Richtlinien wird festgehalten, welche Mechanismen zur Überprüfung verwendet werden, dies hängt maßgeblich von der Qualität des angebotenen Services ab (QoS).

Die Punkte Vertraulichkeit und Nachrichten Integrität sind essenziell notwendig, damit die anderen Punkte vertrauenswürdig realisiert werden können. Die Vertraulichkeit der Kommunikation wird durch Verschlüsselungsmechanismen gewährleistet. Meistens werden SSL/TLS-Tunnel (Secure Socket Layer / Transport Layer Security) zur Verschlüsselung der Kommunikation eingesetzt. Die Nachrichten Integrität kann durch Signierung der zu transportierenden Daten realisiert werden. Bei der Authentifizierung haben sich in Grid-Systemen die X.509-Zertifikate durchgesetzt. In den weiteren Punkten gibt es unterschiedliche Ansätze in der Umsetzung dieser Anforderungen.

3.2 Authentifizierung und Autorisierung im Globus Toolkit

Im Globus Toolkit stellt die Grid Security Infrastructure (GSI) die Sicherheitsfunktionen bereit, die in Grid-Systemen benötigt werden. Dies sind Authentifizierung, Autorisierung, Delegierung und das Schützen der zuübertragenden Nachrichten. Die Grid Security Infrastructure setzt für die Erfüllung ihrer Aufgaben eine Reihe von Standards ein. Welche Standards für welche Aufgabe verwendet werden, ist in der Abbildung 3 dargestellt.

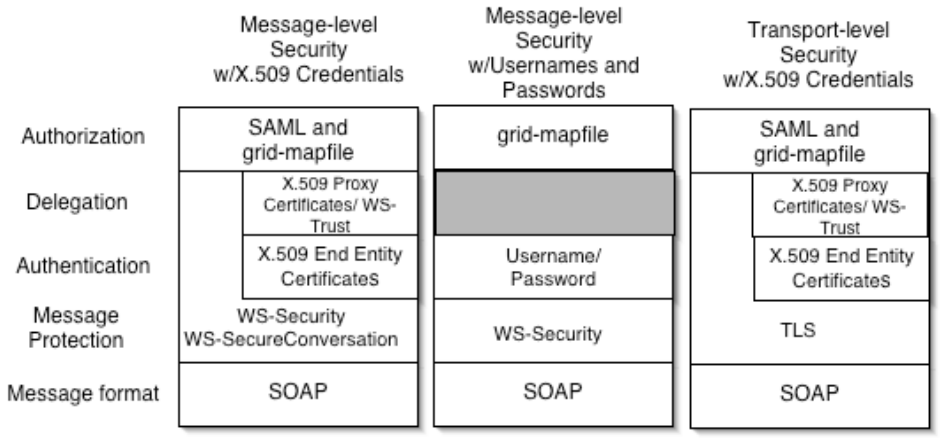

Abbildung 3: Funktionen der Grid Security Infrastructure ([Welch-2005] Seite 2).

Im Kapitel 3.1 wurden Anforderungen an die Authentifizierung und Autorisierung in Grid-Systemen festgehalten. Im folgenden Teil kann man sehen, dass die Grid Security Infrastructure des Globus Toolkit diese auch umsetzt. Die Anforderungen werden auf die Funktionen und die verwendeten Standards des GSI abgebildet.

Die **Vertraulichkeit** wird mit der Verwendung der „Transport-Level Security“ oder der „Message-Level Security“ sichergestellt. Bei der „Transport-Level Security“ werden die SOAP-Nachrichten mittels eines TLS-Tunnels übertragen, da dieser verschlüsselt ist, sind die Nachrichten vor unberechtigtem Zugriff geschützt. Die „Message-Level Security“ gewährleistet die Vertraulichkeit durch die Verschlüsselung der

einzelnen SOAP-Nachrichten, dies ist genauer in den Standards zur Sicherheit von Webservice spezifiziert.

Die **Nachrichten Integrität** kann bei der Verwendung der „Message-Level Security“ gewährleistet werden. Die Message-Level Security basiert auf den Standards für Webservicesicherheit. Diese Standards spezifizieren die Möglichkeit zur Signierung von SOAP-Nachrichten, damit ist es möglich manipulierte Nachrichten zuerkennen.

Die **Authentifizierung** wird im Globus Toolkit durch X.509-Zertifikate oder auch durch X.509 konforme Proxy-Zertifikate umgesetzt. Seit dem Globus Toolkit in der Version 4.0 gibt es die Möglichkeit die Authentifizierung mittels Benutzernamen und Passwort durchzuführen.

Die **Delegierung** wird mit der Hilfe von Proxy-Zertifikaten und dem Delegation Service des Globus Toolkits umgesetzt. Für die Delegierung wird ein Proxy-Zertifikat erstellt, dass den öffentlichen Schlüssel des Delegation Service enthält und mit dem privaten Schlüssel eines Proxy-Zertifikats des Benutzers signiert wird. Das auf diesem Weg erzeugt Proxy-Zertifikat erhält der Delegation Service und dieser stellt es den Grid-Ressourcen zur Verfügung, die im Sicherheitskontext des Benutzers Aufgaben delegieren müssen.

Bei der Erzeugung eines Proxy-Zertifikaten wird das **Single SignOn** durchgeführt. Der Benutzer erzeugt einen privaten Schlüssel und einen Certificate Request für ein Proxy-Zertifikat. Dieses signiert der Benutzer mit dem privaten Schlüssel seines End Entity Certificate, da dieser Verschlüsselte auf einem Datenträger liegt, muss der Benutzer das Passwort zum Entschlüsseln des privaten Schlüssels eingeben. Das Proxy-Zertifikat wird zur Authentifizierung eingesetzt, dieses Proxy-Zertifikat ist aus Sicherheitsgründen eine Short Lived Certificate. Solange das Proxy-Zertifikat gültig ist, muss der Benutzer kein Passwort eingeben, da dieser sich durch das Proxy-Zertifikat authentifiziert.

Die Problematik der **Lebensdauer und Erneuerung von Credentials** kann im Globus Toolkit durch die Verwendung eines myProxy-Servers und eines Jobmanagers automatisiert realisiert werden. Der myProxy-Server ist eine Certification Authority und ein Verzeichnisdienst, der die privaten Schlüssel und die X.509-Zertifikate eines Benutzers speichern kann. Auf dem myProxy-Server wird der private Schlüssel und das End Entity Certificate eines Benutzers auf dem myProxy-Server gespeichert. Der Benutzer kann am myProxy-Server ein Proxy-Zertifikat anfordern und dieses in einem Grid-System zur Authentifizierung einsetzen oder um seine Credentials an einen Grid-Service zu delegieren. Wenn das delegierte Proxy-Zertifikat abläuft, aber ein gültiges Proxy-Zertifikat für die Abarbeitung eines Grid-Jobs benötigt wird kann ein Jobmanager ein neues Proxy-Zertifikat vom myProxy-Server anfordern und den Grid-Job abschließen. Für eine manuelle Erneuerung von delegierten Proxy-Zertifikat bietet der Delegation Service eine geeignete Schnittstelle an.

Die **Autorisierung** wird im Globus Toolkit mit der Hilfe des Grid-Mapfile durchgeführt. Das Grid-Mapfile ist eine Textdatei in der eine Liste von Benutzername gespeichert ist, die berechtigt sind auf einen Grid-Service zuzugreifen. Die Grid Security Infrastructure kann die Autorisierung auch mittels AuthorizationDecisions der SAML-Spezifikation durchführen, diesen werden allerdings vom Shibboleth-Framework nicht unterstützt. Diese werden verwendet um die Autorisierung mit der Hilfe des Community Authorization Service (CAS) durchzuführen. Für die Umsetzung weitere Autorisierungsmechanismen wurde im Globus Toolkit das Authorization Framework entwickelt. Dieses ermöglicht die Realisierung von unternehmensspezifischen Autorisierungsmechanismen. Das Authorization Framework implementiert den SAML-Standard und basiert auf dem Autorisierungsmodell der eXtensible Access Control Markup Language (XACML). In der Abbildung 4 ist der Aufbau des Authorization Frameworks abgebildet.

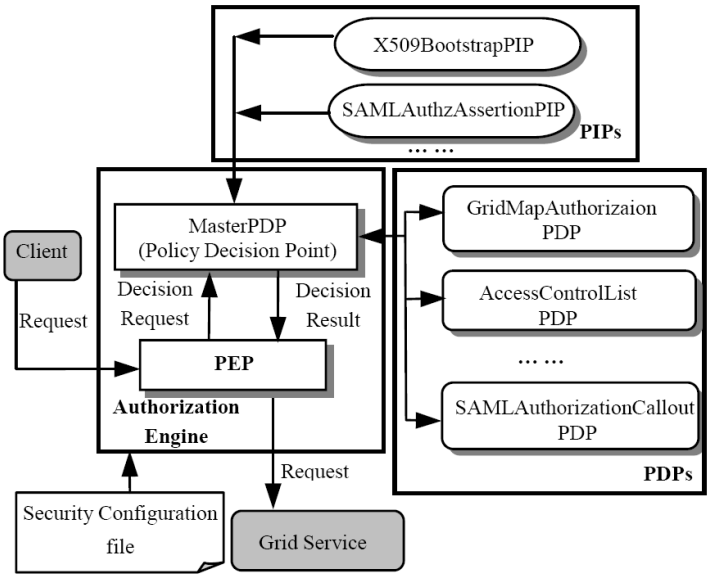

Abbildung 4: Aufbau des Authorization Framework. ([LangB-2006] Seite 3).

Das Authorization Frameworks hat drei Grundbestandteile mit folgenden Aufgaben:

- Der **PEP (Policy Enforcement Point)** fängt die Zugriffsversuche (Request) von Benutzern ab und übergibt den Request an einen Master-PDP (Policy Decision Point). Nach dem der Master-PDP die Autorisierungsentscheidung getroffen hat setzt der PEP diese Entscheidung um.
- Ein **PDP (Policy Decision Point)** setzt die Sicherheitsrichtlinien oder einen Teil der Sicherheitsrichtlinien durch. Ein PDP kontaktiert andere PDPs oder PIPs (Policy Information Point) um die Informationen zuerhalten, die dieser für eine Entscheidung im Rahmen der Zugriffskontrolle benötigt.
- Der **PIP (Policy Information Point)** holt die Informationen, die ein PDP für Umsetzung der Sicherheitsrichtlinie benötigt, wie z. B. die Benutzerattribute.

Die Hauptkomponente ist die Authorization Engine, diese besteht aus dem PEP und dem Master-PDP. Der PEP fängt alle Benutzeranfragen ab und übergibt die Anfrage an den Master-PDP. Der Master-PDP muss alle Informationen sammeln, die er für die Umsetzung der Sicherheitsrichtlinien benötigt. Zu diesem Zweck fragt der Master-PDP alle Policy Information Points des Authorization Frameworks ab. Es werden auch einige spezielle PIPs abgefragt, die sogenannten Bootstrap-PIPs. Diese Sammeln Informationen über die Anfrage, wie z. B. den Subject (Benutzerobjekt), die gewünschte Grid-Ressource und die gewünschte Aktion, die auf der Grid-Ressource ausgeführt werden soll. Die gesammelten Informationen übergibt der Master-PDP den benötigten PDPs des Authorization Frameworks. Jeder PDP setzt eine Sicherheitsrichtlinie um. Anhand der Informationen die der Master-PDP, dem Policy Decision Point übergibt, fällt dieser eine Autorisierungsentscheidung, die der umgesetzten Sicherheitsrichtlinie entspricht. Diese Entscheidung teilt der PDP dem Master-PDP mit. Der Master-PDP sammelt die Autorisierungsentscheidung aller PDPs und fällt anhand dieser eine endgültige Autorisierungsentscheidung. Diese Entscheidung wird dem Policy Enforcement Point übermittelt, der die Autorisierungsentscheidung durchsetzt. Für eigene Autorisierungsmechanismen, die im Authorization Framework umgesetzt werden sollen, muss der Entwickler eigene Policy Information Points und eigene Policy Decision Point entwickeln. Diese müssen in das Authorization Framework eingebunden werden und mittels Konfigurationsdateien eingerichtet werden. (vgl. [Welch-2005] Seite 1 - 6; [LangB-2006] Seite 2 - 3)

3.3 Anwendungsfälle

Die Szenarien in Grid-Systemen, bei denen die Authentifizierung und Autorisierung eine Rolle spielt, werden in der Abbildung 5 als Use-Case-Diagramm dargestellt. Bei dieser Analyse beschreiben die Use-Cases die Abläufe in einem Grid-System ohne Anbindung an das Shibboleth-Framework. Eine Beschreibung der einzelnen Use-Cases folgt im Anschluss des Diagramms in den Tabellen 1 bis 5. Bei den Abläufen wird die Grid-Middleware Globus Toolkit 4.0 zugrunde gelegt.

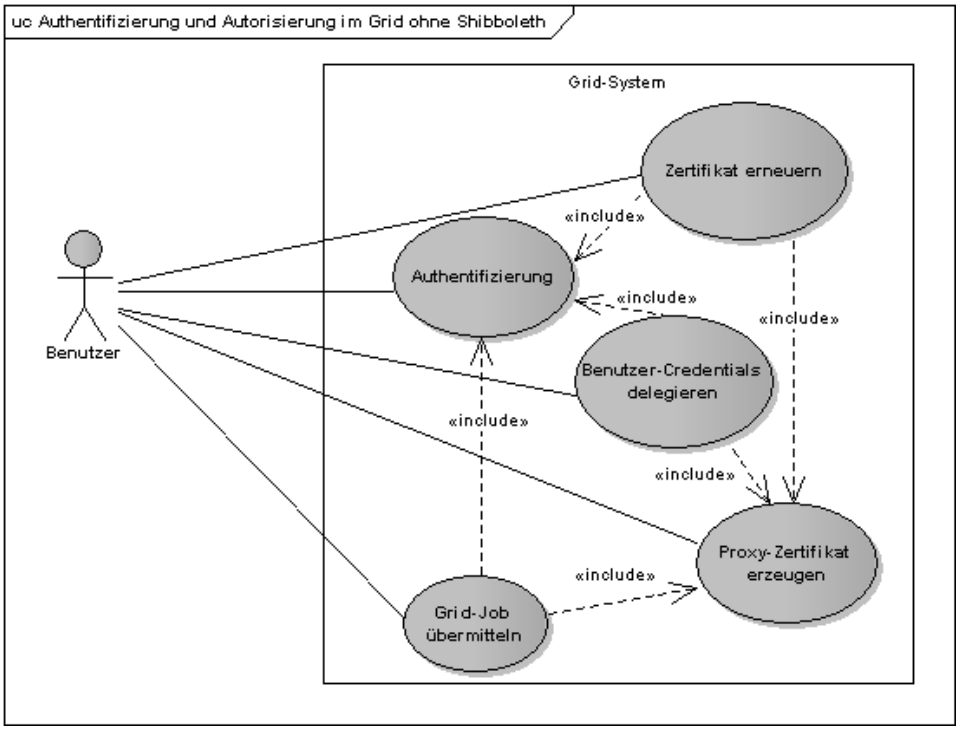

Abbildung 5: Use-Case-Diagramm mit den Anwendungsfällen

Aus Gründen der Übersichtlichkeit wurden im Use-Case-Diagramm die Actors „Grid Security Infrastructure“, „Delegation Service“, „Delagation Service Factory“ und „Grid-Service“ nicht abgebildet. In den folgenden Beschreibungen werden, diese Actors berücksichtigt.

Mit dem Kommandozeilenbefehls `grid-proxy-init -rfc` kann der Benutzer die Erzeugung eines Proxy-Zertifikats automatisiert durchführen, dieser Befehl wird vom Globus Toolkit bereitgestellt (vgl. [GlobA-2008e]).

Use Case Name	**Proxy-Zertifikat erzeugen**
Primary Actor	Benutzer
Further Actors	keine
Stakeholders and their Interests	**Benutzer:** Erzeugen eines Proxy-Zertifikats.
Success Guarantees	Ein Proxy-Zertifikat wurde für den Benutzer erzeugt.
Minimal Guarantees	Der Benutzer bekommt eine Fehlermeldung angezeigt.
Trigger	Der Benutzer benötigt ein Proxy-Zertifikat für die Authentifizierung an einem Grid-System.

Basic Course	1. Der Benutzer erzeugt einen privaten und den dazugehörigen öffentlichen Schlüssel. 2. Der Benutzer erzeugt mit dem neuen öffentlichen Schlüssel einen „Certificate Request“ für ein Proxy-Zertifikat. 3. Als Antwort auf den „Certificate Request“ wird ein Proxy-Zertifikat erzeugt. 4. Der Benutzer stellt einen privaten Schlüssel bereit, der zu einem End Entity Certificate gehört. 5. Der Benutzer gibt das Passwort für die Entschlüsselung des privaten Schlüssels ein. 6. Das Proxy-Zertifikat wird mit dem privaten Schlüssel des End Entity Certificate des Benutzers signiert.
Alternative Course	4a. Der Benutzer stellt einen privaten Schlüssel bereit, der zu einem Proxy-Zertifikat gehört. 4a1. Das neue Proxy-Zertifikat wird mit dem privaten Schlüssel eines anderen Proxy-Zertifikats des Benutzers signiert.

Tabelle 1: Beschreibung des Use-Cases: Proxy-Zertifikat erzeugen

Die Authentifizierung erfolgt mit einem Proxy-Zertifikat während des Aufbaus des TLS-Tunnels. Bei dem Aufbau des TLS-Tunnels wird eine gegenseitige Authentifizierung (Mutual Authentication) durchgeführt (vgl. [GlobA-2008e]).

Use Case Name	**Authentifizierung**
Primary Actor	Benutzer
Further Actors	Grid Security Infrastructure
Stakeholders and their Interests	**Ressourcen Besitzer:** Eindeutige und richtige Identifizierung des Benutzers. **Benutzer:** Erfolgreiche Authentifizierung am Grid-System.
Success Guarantees	Der Benutzer wurde vom Grid-System identifiziert und es kann mit der Autorisierung begonnen werden.
Minimal Guarantees	Eine Fehlermeldung wird angezeigt und ein Eintrag in der Log-Datei wird erzeugt.
Trigger	Der Benutzer hat ein Proxy-Zertifikat erzeugt und möchte eine Grid-Ressource verwenden.

Basic Course	1. Der Benutzer baut eine Verbindung zum Grid-System auf, diese soll mit einem TLS-Tunnel gesichert werden. 2. Der Benutzer übermittelt sein Proxy-Zertifikat an die Grid Security Infrastructure. 3. Die Grid Security Infrastructure validiert das Proxy-Zertifikat. 4. Das Proxy-Zertifikat wurde von der Grid Security Infrastructure akzeptiert. 5. Die Grid Security Infrastructure erzeugt eine zufällige Nachricht und sendet diese an den Benutzer mit der Bitte, diese zu verschlüsseln. 6. Der Benutzer verschlüsselt die Nachricht mit dem privaten Schlüssel, der zu seinem Proxy-Zertifikat gehört. 7. Der Benutzer sendet die verschlüsselte Nachricht an die Grid Security Infrastructure zurück. 8. Die Grid Security Infrastructure entschlüsselt die übermittelte Nachricht mit dem öffentlichen Schlüssel des Benutzers aus dem Proxy-Zertifikat. 9. Die entschlüsselte Nachricht wird mit der übermittelten zufälligen Nachricht verglichen. Wenn beide übereinstimmen, konnte der Benutzer sich erfolgreich authentifizieren. 10. Jetzt muss sich die Grid Security Infrastructure mit derselben Prozedur beim Benutzer authentifizieren, damit der TLS-Tunnel aufgebaut werden kann.
Alternative Course	4a. Das Proxy-Zertifikat konnte von der Grid Security Infrastructure nicht positiv validiert werden. 4a1. Der Aufbau des TLS-Tunnels wird abgebrochen und die Authentifizierung ist fehlgeschlagen.

Tabelle 2: Beschreibung des Use-Cases: Authentifizierung

Ein Grid-Job ist eine Aufgabe, die von einer Computing Resource eines Grid-Systems abgearbeitet wird. Der GRAM-Service (Globus Resource Allocation and Management) verwaltet die Grid-Jobs und teilt diese den Grid-Ressourcen zu. (vgl. [GlobA-2008f]).

Use Case Name	**Grid-Job übermitteln (submit)**
Primary Actor	Benutzer
Further Actors	GRAM-Service
Stakeholders and their Interests	**Ressourcen Besitzer:** Schutz der Grid-Ressource vor unberechtigten Zugriffen. **Benutzer:** Der Grid-Job wird erfolgreich abgearbeitet.
Success Guarantees	Der Job wurde erfolgreich bearbeitet und das Ergebnis wird für den Benutzer bereitgestellt.
Minimal Guarantees	Eine Fehlermeldung wird angezeigt und ein Eintrag in der Log-Datei wird erzeugt.
Trigger	Der Benutzer möchte einen Grid-Job an eine Grid-Ressource übermitteln.
Basic Course	1. Include Use-Case: Proxy-Zertifikat erzeugen 2. Include Use-Case: Authentifizierung 3. Der Benutzer erzeugt einen Grid-Job auf dem GRAM-Service. 4. Der GRAM-Service sendet den Grid-Job an einen Scheduler. 5. Der Scheduler startet den Grid-Job auf der Grid-Ressource. 6. Der Scheduler informiert den GRAM-Service, wenn der Grid-Job abgearbeitet ist. 7. Der GRAM-Service informiert den Benutzer, dass der Grid-Job erfolgreich abgearbeitet wurde.
Alternative Course	-

Tabelle 3: Beschreibung des Use-Cases: Grid-Job übermitteln

Der Use-Case „Benutzer-Credentials delegieren“ spielt dann eine Rolle, wenn eine Grid-Ressource weitere Grid-Ressourcen benötigt um eine Grid-Job abzuarbeiten (vgl. [GlobA-2008d]).

Use Case Name	**Benutzer-Credentials delegieren**
Primary Actor	Benutzer
Further Actors	Delegation Service Factory
Stakeholders and their Interests	**Ressourcen Besitzer:** Schutz der Ressource vor unberechtigten Zugriffen. **Benutzer:** Der Benutzer möchte seine Credentials erfolgreich auf eine Grid-Ressource delegieren.

Success Guarantees	Die Benutzer-Credentials wurden an die Grid-Ressource delegiert und diese kann im Benutzerkontext Grid-Jobs delegieren.
Minimal Guarantees	Eine Fehlermeldung wird angezeigt und ein Eintrag in der Log-Datei wird erzeugt.
Trigger	Der Benutzer möchte einer Grid-Ressource die Möglichkeit einräumen in seinem Benutzerkontext Grid-Jobs zu delegieren.
Basic Course	1. Include Use-Case: Proxy-Zertifikat erzeugen 2. Include Use-Case: Authentifizierung 3. Der Benutzer kontaktiert den Delegation Factory Service, um das X.509-Zertifikat des Delegation Factory Service zuerhalten. 4. Der Benutzer erzeugt ein Proxy-Zertifikat, dass den öffentlichen Schlüssel aus dem X.509-Zertifikat des Delegation Factory Service enthält und signiert dieses mit seinem privaten Schlüssel. 5. Der Benutzer übermittelt das Proxy-Zertifikat an den Delegation Factory Service. 6. Die Delegation Factory Service stellt das Proxy-Zertifikat in einer Delegated Credential Resource der Grid-Ressource bereit. 7. Der Delegation Factory Service übermittelt dem Benutzer eine Endpoint Reference auf die Delegated Credential Resource, diese benötigt der Benutzer für die Erneuerung der delegierten Credentials.
Alternative Course	-

Tabelle 4: Beschreibung des Use-Cases: Teilaufgabe delegieren

In dem Use-Case „Zertifikat erneuern“ wird erläutert, wie ein delegiertes Proxy-Zertifikat erneuert werden kann (vgl. [GlobA-2008d]).

Use Case Name	**Zertifikat erneuern**
Primary Actor	Benutzer
Further Actors	Delegation Service, Delegation Service Factory

Stakeholders and their Interests	**Ressourcen Besitzer:** Schutz der Ressource vor unberechtigten Zugriffen. **Benutzer:** Eine Grid-Ressource soll weiterhin im Benutzerkontext Grid-Jobs delegieren.
Success Guarantees	Die Grid-Ressource erhält gültige Benutzer-Credentials, die dieser zum Delegieren von Grid-Jobs verwenden kann.
Minimal Guarantees	Eine Fehlermeldung wird angezeigt und ein Eintrag in der Log-Datei wird erzeugt.
Trigger	Eine Grid-Ressource stellt fest, dass ein benötigtes Proxy-Zertifikat abgelaufen ist.
Basic Course	1. Include Use-Case: Proxy-Zertifikat erzeugen 2. Include Use-Case: Authentifizierung 3. Der Benutzer kontaktiert den Delegation Factory Service, um das X.509-Zertifikat des Delegation Factory Service zuerhalten. 4. Der Benutzer erzeugt ein Proxy-Zertifikat, dass den öffentlichen Schlüssel aus dem X.509-Zertifikat des Delegation Factory Service enthält und signiert dieses mit seinem privaten Schlüssel. 5. Der Benutzer übermittelt das Proxy-Zertifikat an den Delegation Service. 6. Der Delegation Service erneuert das Proxy-Zertifikat in der Delegated Credential Resource. 7. Die Delegated Credential Resource übermittelt das neue Proxy-Zertifikat an die Grid-Ressourcen.
Alternative Course	-

Tabelle 5: Beschreibung des Use-Cases: Zertifikat erneuern

3.4 Gegenüberstellung einiger Authentifizierungs-Frameworks

In diesem Kapitel werden die drei Authentifizierungs-Frameworks Shibboleth, Community Authorization Server und Virtual Organization Membership Service diskutiert. Es werden die Vor- und Nachteile erarbeitet und es wird die Entscheidung für eines der Frameworks begründet.

3.4.1 Community Authorization Server

Der Community Authorization Server (CAS) stellt Benutzern Proxy-Zertifikate aus. In ein Proxy-Zertifikate wird vom CAS eine SAML-AuthorizationAssertions eingebunden, diese stellt die VO-Richtlinie bezüglich des Benutzers da. Diese Assertion wird auch „CAS Policy Assertion“ genannt, diese wird als unkritische Erweiterung in das Proxy-Zertifikat eingebunden. In der Richtlinie ist spezifiziert, welche Rechte der Benutzer in dieser VO hat. In einer virtuellen Organisation wird nur ein Community Authorization Server eingerichtet, damit dieser seine Arbeit aufnehmen kann, benötigt dieser eine eigene Datenbank in der alle Benutzer und die Richtlinien gespeichert sind, dies erhöht den administrativen Aufwand und zentralisiert die Benutzerverwaltung. Der Community Authorization Server wurde für Grid-System entwickelt und findet auch nur in solchen Systemen Verbreitung. (vgl. [Pearl-2003])

3.4.2 Shibboleth-Framework

Das Shibboleth-Framework ist ein Authentifizierungs- und Autorisierungs-Framework. Es stellt die Attribute eines Benutzer für die Autorisierung bereit. Die Benutzer und deren Attribute können aus dem LDAP-Verzeichnis oder der SQL-Datenbank entnommen werden, die schon zur Benutzerauthentifizierung in einer Organisation eingesetzt wird. Es entsteht kein zusätzlicher Aufwand für das erstellen einer weiteren Datenbank. Für jede Organisation, die an der virtuellen Organisation beteiligt ist, wird ein Identity-Provider benötigt, der die Authentifizierung übernimmt und auch die Attribute bereitstellt, dieser wird von jeder Organisation selbst administriert. Der Austausch der Attribute findet mittels SOAP-Nachrichten statt. Es gibt auch die Möglichkeit die Attribute in das Zertifikat des Benutzers einzubinden. Für die Anbindung der Grid-Middleware Globus Toolkit gibt es eine Erweiterung des GridShib-Projektes. Diese Erweiterung stellt die Attribute in der Grid-Middleware bereit und ermöglicht die Autorisierung mittels dieser Attribute. Allerdings muss bei dieser Erweiterung im Identity Provider eine Mapping-Datei erstellt werden, die für jeden Benutzer eine Beziehung zwischen dem Benutzernamen im Identity Provider und dem DN aus dem Proxy-Zertifikat des Benutzers herstellt. Dies erhöht den administrativen Aufwand, da die Benutzeradministrierung dezentralisiert ist, wird der zusätzliche Aufwand auf die einzelnen Organisationen verteilt. Das Shibboleth-Framework wurde für den Einsatz mit Web-Systemen entwickelt und findet deshalb eine sehr große Verbreitung. (vgl. [Inet2-2008])

3.4.3 Virtual Organization Membership Service

Der Virtual Organization Membership Service (VOMS) stellt den Benutzern Proxy-Zertifikat aus. Diese enthalten die Eigenschaften des Benutzers, die in auf dem VOMS-

System definiert wurden. Zu den Eigenschaften gehören Gruppenzugehörigkeiten oder auch Benutzerrollen, diese Eigenschaften können zur Autorisierung in dem Grid-System verwendet werden. Da der Benutzer seine Eigenschaften in seinem Proxy-Zertifikat mitbekommt, benötigt der VOMS-Server keine Verbindung zur Grid-Middleware. Das X.509-Zertifikat des VOMS-Servers muss als vertrauenswürdig von der Grid-Middleware eingestuft werden, da das Proxy-Zertifikat des Benutzers und damit auch dessen Eigenschaften vom VOMS-Server signiert wurden. Eine virtuelle Organisation wird durch einen VOMS-Server verwaltet, dies führt zu einer zentralen Verwaltung der Grid-Benutzer durch eine Organisation der VO. Für den VOMS-Server muss eine extra Datenbank angelegt werden, in der alle Benutzer mit ihren Eigenschaften angelegt werden müssen. Das neu anlegen der Benutzer erhöht den Aufwand für die Administrierung erheblich. Der Virtual Organization Membership Service wurde für Grid-System entwickelt und wurde auf die Bedürfnisse von Grid-Systemen abgestimmt, deshalb findet es nur in Grid-Systemen Verbreitung. (vgl. [DataG-2003])

3.4.4 Beurteilung

Grundsätzlich sind die Autorisierungsinformationen, die alle drei Systeme bereitstellen für die Verwendung in einem Autorisierungssystem einer Grid-Ressource geeignet. Die größte Flexibilität in der Zustellungsart der Autorisierungsinformationen bietet das Shibboleth-Framework, da es mehrere Wege für das Abholen der Attribute bereitstellt. Ein weiterer Vorteil ist das die Autorisierungsinformationen aus einer bestehen Datenbank bezogen werden können und nicht erst eine neue Datenbank erstellt werden muss. Dies reduziert den administrativen Aufwand und verringert die Redundanz von Informationen. Weniger Redundanz der Benutzerdaten vereinfacht das Umsetzen von Sicherheitsrichtlinien, da der Benutzer nur einmal deaktiviert/gelöscht werden muss, wenn ein Mitarbeiter eine Organisation verlässt. Bei redundanten Benutzerdaten kann es passieren, dass der Benutzer einmal nicht deaktiviert wird und deshalb weiterhin Zugriff auf Daten erhält, den er nicht mehr haben dürfte. Ein weiterer Vorteil des Shibboleth-Frameworks ist, dass jede Organisation der virtuellen Organisation für die Verwaltung ihrer Benutzer zuständig ist und damit der administrative Aufwand auf alle Organisationen aufgeteilt wird. Vor allem die letzten beiden Punkte gaben den Ausschlag für das Shibboleth-Framework, deshalb wird in dem Konzept das Shibboleth-Framework verwendet.

3.5 Resümee der Analyse

In Grid-Systemen stellen die Proxy-Zertifikate der Benutzer eine sehr wichtige Komponente da. Mit diesen wird ein großer Teil der Anforderungen an Grid-Systeme um-

gesetzt bzw. die Umsetzung unterstützt. Die Anforderungen Single SignOn, Delegierung der Benutzer-Credentials und Authentifizierung werden mit den Proxy-Zertifikaten unterstützt. Zur Umsetzung der Vertraulichkeit von Nachrichten in Grid-Systemen tragen die Proxy-Zertifikate ebenfalls bei, da der öffentliche Schlüssel des Benutzers mit dem Proxy-Zertifikat verteilt wird. Die Proxy-Zertifikate übernehmen eine zentrale Rolle in Grid-Systemen, deshalb sollten diese bei der Lösung der Problematik dieser Arbeit Beachtung finden. Es zeigte sich das Grid-Systeme, das Konzept des „Web of Trust“ umsetzen. Dies zeigt sich sehr deutlich daran, dass ein Benutzer von Grid-Systemen die Möglichkeit hat seine Credentials auf Grid-Ressourcen zuübertragen und damit der Grid-Ressource ermöglicht im Namen des Benutzers zuagieren. Der „Web of Trust“ Gedanke wird in Grid-Systemen durch die Verwendung der X.509-Zertifikate unterstützt, da PKI-Systeme ebenfalls dem Prinzip des „Web of Trust“ unterliegen. Bei der Authentifizierung durch X.509-Zertifikate vertraut das Authentifizierungssystem darauf, dass der Zertifikatersteller die Identität überprüft hat, die dem X.509-Zertifikat zugeordnet ist. Dieses Vertrauen zum Zertifikatersteller muss explizit hergestellt werden, in dem das X.509-Zertifikat des Zertifikaterstellers als vertrauenswürdig eingestuft wird. Ein Problem sind die indirekten Vertrauensbeziehungen, die bei einem „Web of Trust“ entstehen. Wenn eine Person A und eine Person B sich gegenseitig vertrauen und Person B auch noch Person C vertraut, dann hat Person A auch ein indirektes Vertrauensverhältnis zu Person C. Eine solche Vertrauenskette kann beliebig lang sein und führt dazu, dass eine Person über ein indirektes Vertrauensverhältnis völlig unbekannten Personen vertraut. Wenn eine Person, der direkt oder indirekt vertraut wird, ein Zertifikat mit einer falschen Identität erstellt und diese Identität zu einem Benutzer gehört, der Zugriff auf eine Ressource hat, kann mit diesem Zertifikat unberechtigterweise auf die Ressource zugegriffen werden. Dieser Problematik wird in Grid-Systemen durch die Verwendung von Proxy-Zertifikaten entgegengewirkt. Zum Einen kann mit der ProxyPolicy eines Proxy-Zertifikat, die Verwendung des Zertifikats eingeschränkt werden. Der zweite Punkt ist, dass Proxy-Zertifikate nur eine kurze Gültigkeit von nur wenigen Stunden haben. Durch die kurze Gültigkeit kann der Schaden eingeschränkt werden, da alle Zertifikate, die mit dem privaten Schlüssel des Proxy-Zertifikats signiert wurden, ungültig werden, sobald das Proxy-Zertifikat abgelaufen ist. Da für die Validierung des Zertifikats, das Zertifikat des Zertifikaterstellers gültig sein muss. Mit der Hilfe des Shibboleth-Frameworks kann versucht werden, die Risiken, die ein vertrauensbasiertes System hat, zu verringern.

4 Das Shibboleth-Framework

Die Authentifizierungs- und Autorisierungsinfrastruktur (AAI) von Shibboleth werden in diesem Kapitel erläutert. Das Zusammenspiel der einzelnen Komponenten wird in Sequenzdiagrammen vermittelt. Die Komponenten des GridShib-Projekts werden vorgestellt, diese realisieren eine Anbindung des Shibboleth-Frameworks an die Grid-Middleware Globus Toolkit.

4.1 Grundlagen von Shibboleth

Im Rahmen des Shibboleth Projekts wurde unter der Führung des Middleware Architecture Committee for Education (MACE) der „Internet2 Middleware Initiative“ das Shibboleth-Framework entwickelt. Das Ziel des Shibboleth Projektes ist es eine Open-Source-Software zuentwickeln, die es ermöglicht Web-Ressourcen, die einer Zugriffskontrolle unterliegen, organisationsübergreifende bereitzustellen. Die erste Version des Shibboleth-Framework erschien im Juni 2003. Das Shibboleth-Framework implementiert den Standard der Security Assertion Markup Language (SAML). In der Version 1.3 des Shibboleth-Frameworks sind die SAML-Profile und die SAML-Assertions der SAML-Spezifikation in der Version 1.1 entnommen. Die Metadatendatei des Shibboleth-Frameworks wird nach der SAML-Spezifikation in der Version 2.0 erstellt. Im Shibboleth-Framework ist der Identity Provider (IdP) für die Authentifizierung der Benutzer zuständig, der IdP stellt auch Informationen bereit die für die Autorisierung benötigt werden, das sind die Attribute des Benutzers. Der IdP enthält einen Single SignOn-Service, das hat den Vorteil das die Benutzer sich nur einmal anmelden müssen und dann Zugriff auf alle Webservices bekommen. Das zweite wichtige Element des Shibboleth-Frameworks ist der Service Provider (SP), der die Webservices bereitstellt und die Autorisierung der Benutzer durchführt, dazu benötigt dieser die Authentifizierungs- und Autorisierungsinformationen des Benutzers. Die Systeme des Shibboleth-Frameworks bilden eine Authentifizierungs- und Autorisierungsinfrastruktur (AAI). An einer Authentifizierungs- und Autorisierungsinfrastruktur (AAI) des Shibboleth-Frameworks können mehrere Organisationen beteiligt sein. Die beteiligten Organisationen bilden eine virtuelle Organisation (VO), für die virtuelle Organisation werden Regeln definiert, die bei der Authentifizierung und Autorisierung beachtet werden müssen. Jede Organisation ist für die Authentifizierung seiner Benutzer zuständig. Die Autorisierung der Benutzer wird von der Organisation realisiert, die den Webservice bereitstellt. Für die Autorisierung werden die Attribute des Benutzers verwendet, diese muss die Organisation des Benutzers bereitstellen. (vgl. [Inet2-2008]; [Scavo-2005] Seite 5, 6)

Eine Beschreibung der Authentifizierungs- und Autorisierungsinfrastruktur (AAI)

des Shibboleth-Frameworks erfolgt in einer Metadaten-Datei. In der Metadaten-Datei wird die Aufgabe und die eindeutige ID von jedem System der Shibboleth AAI notiert. Es wird auch das X.509-Zertifikat von jedem System in der Metadaten-Datei festgehalten, dies ist für Authentifizierung der einzelnen Systeme notwendig. Alle Systeme der Shibboleth AAI müssen auf die Metadaten-Datei Zugriff haben, damit diese alle Systeme dieser AAI identifizieren werden können und eine Kommunikation zwischen den Systemen stattfinden kann. Wenn ein System eine SAML-Nachricht mit einer ID eines Systems bekommt, die nicht in der Metadaten-Datei aufgelistet ist, wird diese Nachricht verworfen. Damit Manipulationen an der Metadaten-Datei festgestellt werden können, sollte die gesamte Metadaten-Datei signiert werden. Ein Beispiel für eine Metadaten-Datei ist im Anhang B auf der Seite 67 abgebildet. (vgl. [Scavo-2005] Seite 26 - 30)

Die Grundlage des Kommunikationsablaufs im Shibboleth-Framework bilden die „Single SignOn“ und „Attribute Exchange“ Profile, die in der Security Assertion Markup Language in der Version 1.1 definiert wurden. Diese SAML-Profile mussten auf die Bedürfnisse des Shibboleth-Frameworks angepasst werden. In den SAML-Profilen muss der Benutzer zuerst eine Verbindung zu dem Identity Provider aufbauen, damit sich der Benutzer authentifizieren kann und die benötigten Autorisierungsinformationen bekommt, z. B. die Attribute. Mit diesen Informationen kann der Benutzer den Service Provider kontaktieren und Zugriff auf den gewünschten Webservice erhalten. Im Shibboleth-Frameworks soll der Benutzer direkt den Service Provider kontaktieren können. Der Service Provider veranlasst den weiteren Ablauf, der nötig ist um den Benutzer zuauthentifizieren und zuautorisieren. Dazu setzt das Shibboleth-Framework vier SAML-Profile um:

- Das **Authentication Request Profile** ist die Grundlage für alle weiteren SAML-Profile, die das Shibboleth-Framework implementiert. In diesem SAML-Profile wird eine Authentifizierungsanfrage in der Form einer URL spezifiziert. Diese erzeugt der Service Provider des Shibboleth-Frameworks und leitet damit den Benutzer zum Identity Provider, der den Authentifizierungsvorgang beginnt.

- Das **Browser/POST Profile** besteht aus dem „Authentication Request Profile“ des Shibboleth-Frameworks und dem „SAML 1.1 Browser/POST Profile“. Die Authentifizierungsanfrage wird durch das „Authentication Request Profile“ spezifiziert. Der Benutzer erhält die Authentifizierungsinformationen als „HTML Form“. Die Daten des „HTML Forms“ werden zum Service Provider als „HTML POST-Request“ übermittelt, dieser Ablauf ist in dem „SAML 1.1 Browser/POST Profile“ spezifiziert.

- Das **Browser/Artifact Profile** verwendet das „Authentication Request Profile“

des Shibboleth-Frameworks und das „SAML 1.1 Browser/Artifact Profile“. Das „Authentication Request Profile“ beschreibt den Ablauf der Authentifizierungsanfrage. Der Benutzer bekommt vom Identity Provider nur eine Artifact übermittelt, dies stellt einen Zeiger auf die Authentifizierungsinformationen da. Mit der Hilfe des Artifacts kann der Service Provider die Authentifizierungsinformationen vom Identity Provider mittels eines Back-Channel-Exchange abholen. Da bei dieser Variante die Authentifizierungsinformationen direkt zwischen dem Identity Provider und dem Service Provider übermittelt werden, besteht nicht die Gefahr, dass der Benutzer die Daten manipulieren kann. Im „SAML 1.1 Browser/Artifact Profile“ ist dieser Ablauf definiert.

- Das **Browser/POST Profile with Attribute Exchange** ergänzt das „Browser/-POST Profile“ des Shibboleth-Frameworks um einen Attributaustausch. Der Attributaustausch wird mittels eines Back-Channel-Exchange durchgeführt. Bei einem Back-Channel-Exchange findet der Austausch der Daten direkt zwischen dem Service Provider und dem Identity Provider statt, der Webbrowser des Benutzers ist nicht in diese Kommunikation eingebunden.

- Das **Browser/Artifact Profile with Attribute Exchange** erweitert das „Browser/Artifact Profile“ des Shibboleth-Frameworks um einen weiteren Back-Channel-Exchange, in diesem werden die Attribute des Benutzers übermittelt.

Im Shibboleth-Framework kann der Benutzer den Service Provider zuerst kontaktieren, daraus ergibt sich das „Identity Provider Discovery Problem“. Der Service Provider kann nicht wissen, welcher Identity Provider der Shibboleth AAI, der Identity Provider des Benutzers ist. Für die Lösung dieses Problem wurde der Discovery Service entwickelt, dort kann der Benutzer seinen Identity Provider auswählen, bevor dieser dorthin weitergeleitet wird. Von allen SAML-Profilen des Shibboleth-Frameworks, außer dem „Authentication Request Profile“, gibt es noch eine abgewandelte Variante, die den Discovery Service im Kommunikationsablauf berücksichtigt. (vgl. [Scavo-2005] Seite 13 - 24)

Die Kommunikation zwischen dem Webrowser und dem Shibboleth-AAI, sowie die Kommunikation innerhalb der Shibboleth-AAI, findet mittels SSL/TLS-Verbindungen statt. Damit ist sicher gestellt das die übertragenen Daten verschlüsselt werden und die Benutzerdaten nicht zeitnah aus der Übertragung gewonnen werden können. Die Etablierung der SSL/TLS-Verbindungen ist die Aufgabe der Webserver, die die Komponenten des Shibboleth-Frameworks bereitstellen. Alle Komponenten der Shibboleth-AAI müssen sich gegenseitig authentifizieren, bevor diese miteinander kommunizieren. Die Authentifizierung erfolgt mittels X.509-Zertifikaten. Ein weiterer Schutzmechanismus ist, dass alle SAML-Nachrichten vom Ersteller signiert werden, damit

kann einer Manipulation der SAML-Nachrichten erkannt werden und diese SAML-Nachricht wird nicht akzeptiert. In den SAML-Nachrichten sind SAML-Assertions eingebettet, diese enthalten Informationen über den Benutzer. Die wichtigsten SAML-Nachrichten, die das Shibboleth-Framework verwendet, sind die Folgenden:

- Das **AuthenticationAssertions** beinhaltet die Informationen über eine Authentifizierung eines Benutzers. Ein AuthenticationAssertions ist im Anhang B auf der Seite 67 abgebildet.
- Eine **AttributeQuery** wird dafür verwendet um die Attribute eines Benutzers anzufordern. Im Anhang B auf der Seite 68 ist eine AttributeQuery dargestellt.
- Ein **AttributeResponse** ist die Antwort auf eine AttributeQuery und es beinhaltet die Attribute eines Benutzers. Die Attribute sind in eine AttributeAssertion eingebettet, dass von dem AttributeResponse übertragen wird. Ein Beispiel eines AttributeResponse ist im Anhang B auf der Seite 69 vorhanden.

4.2 Die Architektur von Shibboleth

Die Authentifizierungs- und Autorisierungsinfrastruktur (AAI) des Shibboleth-Frameworks besteht aus drei Komponenten dem Identity Provider, dem Service Provider und dem Discovery Service. Die AAI des Shibboleth-Frameworks wird von einem LDAP-Server ergänzt. Im LDAP-Verzeichnis sind die Benutzer und deren Attribute gespeichert. Alle Kommunikationsverbindungen sind aus Sicherheitsgründen durch einen SSL/TLS-Tunnel geschützt, dies betrifft die Kommunikation zwischen dem Webbrowser und den Komponenten der Shibboleth AAI, genauso wie die Kommunikation zwischen den Komponenten der Shibboleth AAI und auch die Verbindung zum LDAP-Server. Den Aufbau der Authentifizierungs- und Autorisierungsinfrastruktur (AAI) kann man in der Abbildung 6 erkennen.

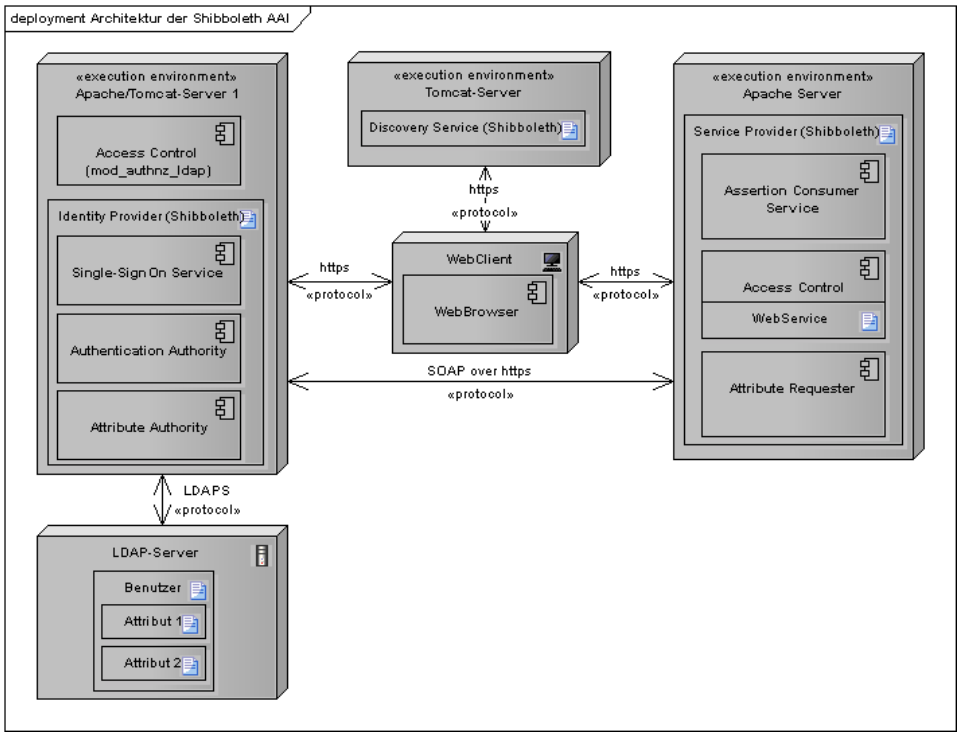

Abbildung 6: Architektur der Authentifizierungs- und Autorisierungsinfrastruktur des Shibboleth-Frameworks. (vgl. [Scavo-2005] Seite 21)

Die Abbildung 6 stellt eine vereinfachte Sicht auf eine Shibboleth-AAI da. In der Realität besteht eine Shibboleth-AAI aus mehreren Service Providern und aus mehreren Identity Provider mit der dazugehörigen Benutzerdatenbank. Die Anzahl der Identity Provider hängt von der Anzahl der beteiligten Organisationen ab, da jede Organisation ihre Benutzer selber verwaltet und diese in einer eigenen Benutzerdatenbank speichert. Auf diese Benutzerdatenbank greift der Identity Provider zu. Die Anzahl der verwendetet Service Provider hängt von der Menge der bereitgestellten Webservices ab und von welchen Organisationen diese bereitgestellt werden. Ein Discovery Service pro Authentifizierungs- und Autorisierungsinfrastruktur ist ausreichend.

4.2.1 Identity Provider

Jeder Benutzer, der die Authentifizierungs- und Autorisierungsinfrastruktur von Shibboleth benutzt, benötigt ein Benutzerkonto auf dem Identity Provider (IdP). Der IdP bezieht seine Benutzer von einem LDAP-Verzeichnis oder einer anderen Datenbank, die sich mit Java Database Connectivity (JDBC) ansprechen lässt. Der Identity Provider (IdP) authentifiziert die Benutzer und stellt auch die Attribute des Benutzers bereit. Die Installation erfolgt auf einem Tomcat-Server (Version 5.5.x), dem ein Apache-Server (Version 2.2.x) mit SSL-Unterstützung vorgeschaltet ist. Der IdP besteht aus vier Komponenten (vgl. [Scavo-2005] Seite 5 - 6; [Canto-2007]):

- Die **Authentication Authority** identifiziert die Benutzer und erzeugt die AuthenticationStatements. Die benötigten Informationen bekommt die Authentication Authority von der Zugriffskontrolle, die den SSO-Service vor unberechtigten Zugriffen schützt. Wenn die Zugriffskontrolle eines Apache-Servers den SSO-Service schützt, wird die Umgebungsvariable `$REMOTE_USER` zur Identifikation des Benutzers verwendet.

- Der **Single SignOn-Service** muss von einer Zugriffskontrolle geschützt werden. Dafür kann die Zugriffskontrolle des Apache-Servers oder des Tomcat-Servers eingesetzt werden. Der SSO-Service beginnt den Authentifizierungsprozess im IdP. Der „Inter-Site Transfer Service“ des SSO-Service übernimmt die Kommunikation mit der Authentication Authority und erzeugt die AuthenticationAssertions, die die AuthenticationStatements von der Authentication Authority enthalten.

- Der **Artifact Resolution Service** wird für alle Browser/Artifact-Profile benötigt. Bei diesen Profilen sendet der Identity Provider nur eine Referenz (Artifact) auf die SAML-Assertions an den Service Provider. Diese Referenz sendet der Service Provider an den Artifact Resolution Service, darauf übermittelt der Artifact Resolution Service die SAML-Assertion direkt an den Service Provider. Die SAML-Assertion wird nicht über den Browser des Benutzers übertragen, dieses Verfahren wird Back-Channel-Exchange genannt.

- Die **Attribute Authority** bearbeitet die AttributeRequests. Die Signatur eines AttributeRequests wird von der Attribute Authority validiert. Wenn die Signatur korrekt ist, erzeugt die Attribute Authority ein AttributeAssertion, dass alle Attribute des Benutzers enthält. Die AttributeAssertion wird dem SAML-Profil entsprechend weiter behandelt. Die Weitergabe der Attribute kann durch eine Attribute Release Policy eingeschränkt werden. In dieser kann festgelegt werden welche Attribute an welchen Service Provider weitergegeben werden, da nicht jeder Service Provider zur Autorisierung dieselben Attribute benötigt. Aus Sicherheitsgründen sollte jeder Service Provider nur die Attribute erhalten, die dieser für die Autorisierung benötigt.

In der Version 2.0 wurde der Identity Provider komplett überarbeitet. Eine wichtige Neuerung ist, dass der IdP die Authentifizierung selber durchführen kann und nicht mehr auf die Zugriffskontrolle des Apache-Webservers angewiesen ist. Dafür wurde der Java Authentication and Authorization Service (JAAS) im IdP implementiert. Damit hat der Identity Provider die volle Kontrolle über den Authentifizierungsvorgang.

4.2.2 Service Provider

Der Service Provider (SP) stellt die Webservices bereit und schützt diese vor unberechtigten Zugriffen. Die Zugriffskontrolle erfolgt anhand der Attribute des Benutzers, die der Service Provider vom Identity Provider erhält. Der SP kann auch so konfiguriert werden, dass ein AuthenticationStatement über den Benutzer zur Autorisierung ausreicht und damit kein Attributaustausch mit dem IdP erfolgen muss. Der SP wird auf einem Apache Webserver (Version 2.0.x) installiert. Alle Anfragen werden vom Apache-Server an den Shibboleth Service Provider durchgereicht, dieser führt die Autorisierung durch. Der SP besteht aus zwei Komponenten (vgl. [Scavo-2005] Seite 6):

- Der **Assertion Consumer Service** verarbeitet die AuthenticationAssertions, die dieser vom SSO-Service oder dem Artifact Resolution Service des Identity Providers erhält. Wenn nötig initiiert der Assertion Consumer Service den Attributaustausch, der vom Attribute Requester des SP und von der Attribute Authority des IdP durchgeführt wird. Danach wird ein Sicherheitskontext zwischen dem Client und dem Service Provider etabliert und der Client wird auf den gewünschten Webservice weitergeleitet.
- Der **Attribute Requester** kontaktiert die Attribute Authority des Identity Providers um die Attribute des Benutzers zuerhalten. Zwischen dem Identity Provider und dem Service Provider muss ein Sicherheitskontext etabliert sein, beide Kommunikationspartner müssen sich gegenseitig authentifizieren. Die Authentifizierung findet mittels X.509-Zertifikaten statt. Ohne den Sicherheitskontext findet kein der Austausch der Attribute statt. Die Übertragung wird über einen Back-Channel-Exchange realisiert, dies ist eine direkt Verbindung zwischen dem Identity Provider und dem Service Provider, der Browser des Benutzers wird in diese Kommunikation nicht eingebunden.

4.2.3 Discovery Service

Der Discovery Service oder auch „Where are you from?-Service“ (WAYF) ermöglicht dem Benutzer seinen Heimat-IdP auszuwählen. Bei Systemen, an denen mehr als eine Organisation beteiligt ist, gibt es meistens mehrere IdPs. Deshalb ist es notwendig einen Discovery Service zuverwenden, damit der Service Provider den richtigen Identity Provider kontaktieren kann, um die korrekten Benutzer-Attribute zu erhalten. Die Liste der Identity Provider liest der Discovery Service aus der Shibboleth-Metadaten-Datei aus (vgl. [Scavo-2005] Seite 6). Der Discovery Service erhält einen AuthnRequest, diesen leitet der Discovery Service an den Heimat-IdP des Benutzers weiter.

Dafür muss der Benutzer auf der Weboberfläche des Discovery Service seinen Heimat-IdP auswählen. Auf diesen Identity Provider wird der Benutzer per redirect weitergeleitet und muss sich dort authentifizieren.

4.3 Die Kommunikationsprozesse in der AAI

In diesem Abschnitt wird der Kommunikationsverlauf in der Shibboleth AAI erläutert. Das Shibboleth-Framework unterstützt verschieden Profile. Die Profile unterscheiden sich in der Art der Übermittlung der Authentifizierung-Informationen und ob die Attribute des Benutzers für eine Autorisierung benötigt werden. In den Abbildungen 7 und 8 wird der Kommunikationsverlauf des „Browser/Post-Profiles mit Attributaustausch“ dargestellt.

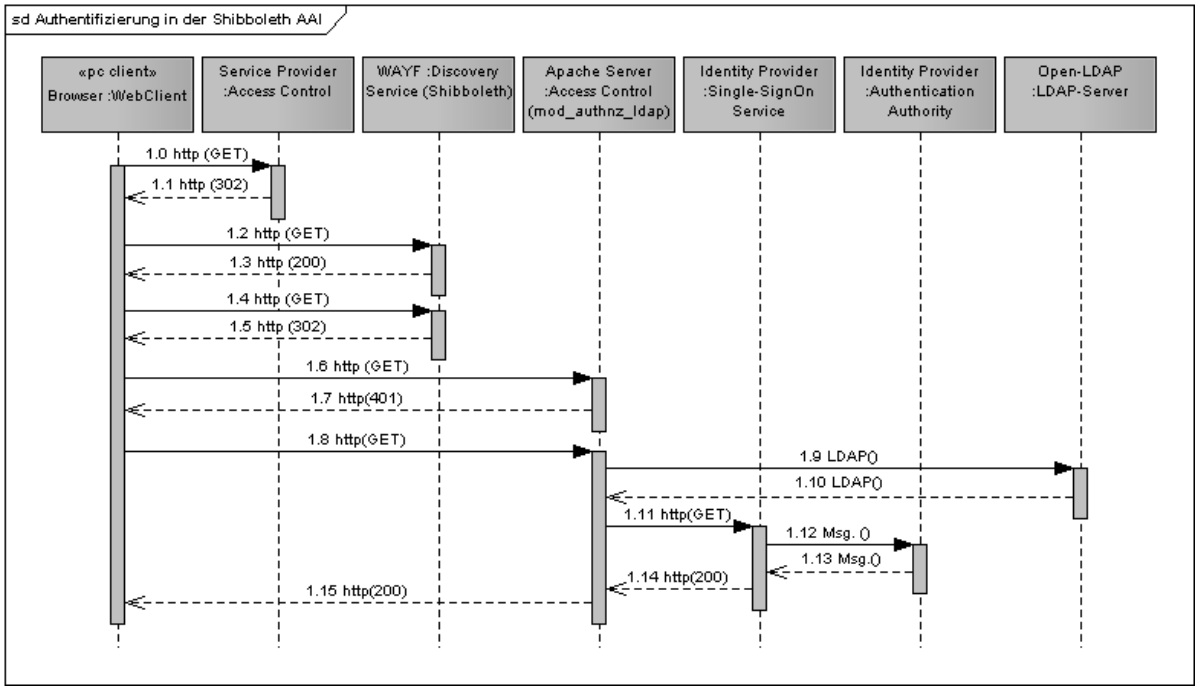

Abbildung 7: Ablauf der Authentifizierung in der Shibboleth AAI. (vgl. [Scavo-2005] Seite 21)

Wenn der Benutzer auf einen Webservice zugreifen möchte, der durch eine Authentifizierungs- und Autorisierungsinfrastruktur (AAI) des Shibboleth-Frameworks geschützt ist, muss dieser sich zuerst an der Shibboleth-AAI authentifizieren. In der Abbildung 7 wird der Authentifizierungsvorgang in der Shibboleth AAI unter der Verwendung des „Browser/Post-Profiles mit Attributaustausch“ dargestellt. Dieses Shibboleth-Profil beginnt damit, dass der Benutzer mit einem Webbrowser den gewünschten Webservice aufruft (1.0), dies erfolgt mit einem „GET Request“. Da zwischen dem Benutzer und dem Service Provider kein Sicherheitskontext etabliert ist, sprich der Benutzer nicht authentifiziert und nicht autorisiert wurde, bekommt der Benutzer einen

redirect als Antwort (1.1). Der redirect enthält die URL des Discovery Service, dieser URL sind auch einpaar Variablen angehängt, die im späteren Verlauf ausgewertet werden. Im folgenden Beispiel kann man eine solche URL sehen.

```
1 https :// wayf.example.org/?
2   target=https :// sp.example.org/myresource&
3     shire=https :// sp.example.org/shibboleth/SSO/POST&
4       providerId=https :// sp.example.org/shibboleth
```

Eine URL eines redirect zum Dicovery Service. (vgl. [Scavo-2005] Seite 14)

In der Zeile 1 des Beispiels ist die URL des Discovery Services zusehen und in den Zeilen 2 bis 4 folgen drei Variablen, die an den Discovery Service übergeben werden. In der Variable target (Zeile 2) ist die URL des ursprünglich angewählten Webservices gespeichert. Die Variable shire (Zeile 3) enthält die URL des Assertion Consumer Service des Service Providers. In der Variable providerId (Zeile 4) ist die eindeutige ID des Service Providers gespeichert. Dieser redirect veranlasst den Webbrowser des Benutzers den Discovery Service mittels eines „GET Request“ aufzurufen (1.2). Vom Discovery Service bekommt der Benutzer ein „HTML Form“, dass in Webbrowser des Benutzers eine Auswahlliste mit den bekannten Identity Providern darstellt (1.3). Der Benutzer wählt seinen Identity Provider in der Liste aus und sendet die Daten mit einem „GET Request“ an den Discovery Service (1.4). Der Discovery Service erstellt ein Cookie mit den Daten des ausgewählten Identity Providers. Desweiteren wird ein redirect erzeugt, der den Benutzer zu dem Single SignOn-Service des ausgewählten Identity Providers weiterleitet (1.5). Allerdings wird dieser redirect von der Zugriffskontrolle des Apache Servers abgefangen (1.6), dieser überprüft ob der Benutzer am Identity Provider angemeldet ist. Da dies nicht der Fall ist, bekommt der Benutzer den „http-Statuscode 401 Unauthorized“ zurück geliefert (1.7). Dies veranlasst den Webbrowser ein Eingabefenster zuöffnen, in dem der Benutzer seinen Benutzernamen und Passwort eingeben muss. Darauf hin versucht der Browser erneut den Single SignOn-Service des Identity Provider zukontaktieren, allerdings ist dieses Mal in der http-Nachricht das „Authorization Header Field“ mit den Benutzer Credentials ausfüllt (1.8). Dieses Mal verwendet die Zugriffskontrolle des Apache-Servers die Daten aus dem „Authorization Header Field“ um den Benutzer zu authentifizieren. Dazu kontaktiert der Apache-Server einen LDAP-Server, in dem der Benutzer vorhanden ist (1.9). Dort werden die Benutzer-Credentials überprüft und das Ergebnis wird dem Apache Server mitgeteilt (1.10). Wenn die Benutzer-Credentials korrekt sind, wird der „GET Request“ an den Single SignOn-Service des Identity Providers weitergeleitet (1.11). Damit der Authentifizierungsvorgang im Identity Provider abgeschlossen werden kann, wird die Authentication Authority kontaktiert (1.12). Die Authentication Authority wertet die Apache-Umgebungsvariable `$REMOTE_USER` aus, dort ist der Benutzername des vom Apache-Server authentifizierten Benutzers gespeichert. Für den

Benutzer wird ein AuthenticationStatement erzeugt, dieses wird von der Authentication Authority an den Single SignOn-Service geschickt (1.13). Der Single SignOn-Service erzeugt ein „HTML Form“ in das ein SAML-Response eingefügt ist. Der SAML-Response enthält ein AuthenticationAssertion, dass das AuthenticationStatement des Benutzers enthält. Ein Beispiel für eine AuthenticationAssertion kann man im Anhang B auf der Seite 67 sehen. Im folgenden Beispiel ist ein SAML-Response abgebildet.

```
<samlp:Response
  xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
  MajorVersion="1" MinorVersion="1"
  IssueInstant="2004-12-05T09:22:02Z"
  Recipient="https://sp.example.org/shibboleth/SSO/POST"
  ResponseID="c7055387-af61-4fce-8b98-e2927324b306">
      <!-- Singnatur des Response Erstellers eingefügen-->
  <samlp:Status><samlp:StatusCode Value="samlp:Success"/></samlp:Status>
      <!-- SAML AuthenticationAssertions eingefügen -->
</samlp:Response>
```

Ein SAML-Response des Single SignOn-Service eines IdPs. (vgl. [Scavo-2005] Seite 15)

Diese „HTML Form“ wird an den Benutzer geschickt (1.14)(1.15). Damit ist der erste Teil des „Browser/Post-Profiles mit Attributaustausch“ durchgeführt worden, die Authentifizierung. (vgl. [Scavo-2005] Seite 21 - 23)

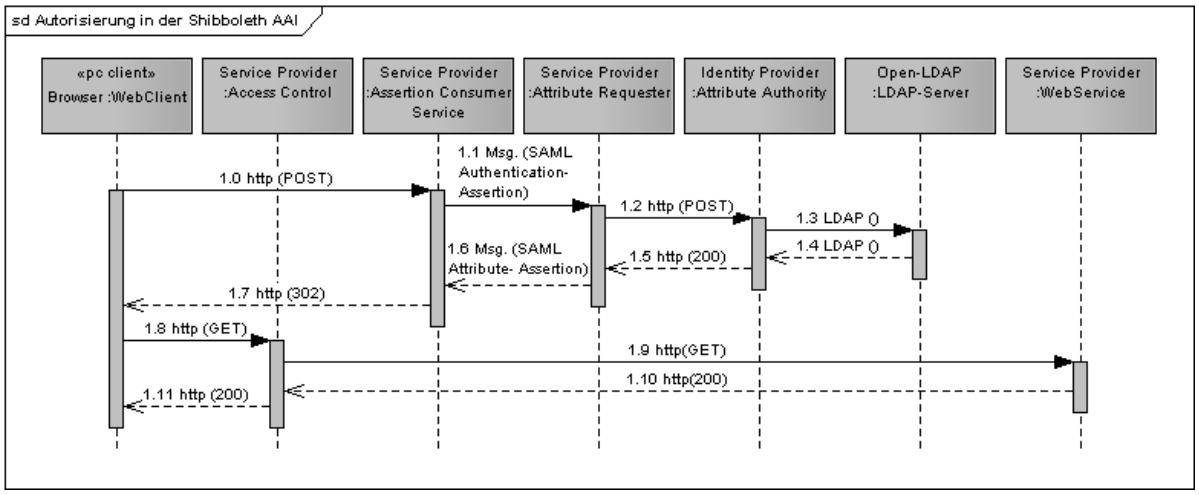

Abbildung 8: Ablauf der Autorisierung in der Shibboleth AAI. (vgl. [Scavo-2005] Seite 21)

In der Abbildung 8 wird der zweite Teil des „Browser/Post-Profiles mit Attributaustausch“ dargestellt, die Autorisierung. Am Ende der Authentifizierung hat der Benutzer ein AuthenticationStatement erhalten, dieses bekam er in einem „HTML Form“ übermittelt. Wenn der Benutzer das „HTML Form“ abschickt, wird der Inhalt des „HTML Form“ in einem „POST request“ an den Assertion Consumer Service des Service

Providers geschickt (1.0). Der Assertion Consumer Service überprüft die Signatur des SAML-Response um die Korrektheit des SAML-Response zu überprüfen. Wenn der SAML-Response korrekt ist, wird ein Sicherheitskontext zwischen dem Benutzer und dem Service Provider etabliert. Damit die Attribute des Benutzers abgeholt werden können, übergibt der Assertion Consumer Service die Kontrolle an den Attribute Requester (1.1). Der Attribute Requester sendet einen „POST request“ an die Attribute Authority des Identity Providers (1.2). Der „POST request“ transportiert eine SOAP-Nachricht. Der SOAP-Envelope enthält einen SAML-Request, dieser beinhaltet eine AttributeQuery. Ein Beispiel für eine solche AttributeQuery ist im Anhang B auf der Seite 68 abgebildet. Die Attribute Authority holt sich die Attribute des Benutzers aus einem LDAP-Verzeichnis (1.3)(1.4). Die Attribute werden in ein AttributeStatement zusammengefasst, dieses wird in eine AttributeAssertion eingebettet. Zum Versenden wird die AttributeAssertion in einen SAML-Response eingepackt, diese wird ebenfalls als SOAP-Nachricht an den Attribute Requester geschickt (1.5). Ein Beispiel für einen AttributeResponse ist im Anhang B auf der Seite 69 dargestellt. Die AttributeAssertion wird an den Assertion Consumer Service weitergeleitet (1.6). Dieser entnimmt die Attribute aus dem AttributeAssertion und übergibt diese in den Sicherheitskontext. Desweitern erzeugt der Assertion Consumer Service einen redirekt zu dem gewünschten Webservice (1.7). Wenn der Benutzer auf den Webservice zugreift (1.8), überprüft die Zugriffskontrolle den Sicherheitskontext des Benutzers. Es wird überprüft, ob ein Sicherheitskontext zwischen dem Benutzer und dem Service Provider existiert. Wenn dies der Fall ist, wird festgestellt ob der Benutzer die benötigten Attribute besitzt, die für den Zugriff auf den Webservice benötigt werden. Wenn dies auch zutrifft, erhält der Benutzer zugriff auf den Webservice (1.9) - (1.11). (vgl. [Scavo-2005] Seite 21 - 23)

4.4 Die GridShib-Erweiterung

Das GridShib-Projekt hat zum Ziel das Shibboleth-Framework an Grid-Systeme anzubinden, die auf der Grid-Middleware Globus Toolkit basieren. Damit wird eine Attributbasierte Autorisierung in Grid-Systemen umgesetzt. Damit entfällt die Notwendigkeit die Gridmap-Datei zuverwalten, die sonst zur Autorisierung verwendet wird. Dieses Projekt wurde von der NSF Middleware Initiative finanziert und hat momentan des Status eines Globus Incubator Projekts. Eine Globus Incubator Projekt, könnte man als „Globus Projekt auf Probe“ bezeichnen, da es bei einem erfolgreich Abschluss zu einem Globus Projekt wird. Im GridShib-Projekt werden vier Komponenten entwickelt:

- GridShib for Globus Toolkit (Version 0.6.0)
- GridShib for Shibboleth (Version 0.5.1)

- GridShib Certificate Authority (Version 0.5.1)
- GridShib SAML Tools (Version 0.3.2)

4.4.1 GridShib for Globus Toolkit

GridShib for Globus Toolkit erweitert das Globus Toolkit 4.0 um Funktionen, die eine Kommunikation mit einem Identity Provider (IdP) des Shibboleth-Framework ermöglichen. Diese Erweiterung fragt die Attribute eines Benutzers an der Attribute Authority (AA) eines Identity Provider ab, die für die Autorisierung des Benutzers herangezogen werden. Es ist möglich die Attribute mittels push oder pull abzuholen. Das Authorization Framework des Globus Toolkit 4.0 wird um Policy Information Points (PIP) und Policy Decision Point (PDP) erweitert. Der PIP sammelt die Informationen, die für die Zugriffskontrolle benötigt werden. In diesem Fall sind das die Attribute des Benutzers. Die Attribute stellt der PIP dem PDP und anderen PIPs zur Verfügung. Der PDP holt die benötigten Informationen für die Zugriffskontrolle von den PIPs ab und gewährt oder verweigert anhand dieser, in Verbindung mit den definierten Sicherheitsrichtlinien, den Zugriff auf die angeforderte Grid-Ressource. Mit dieser Erweiterung kann im GSI auf attributbasierte Autorisierung umgestellt werden, allerdings muss auf dem Identity Provider die Erweiterung GridShib for Shibboleth installiert sein, damit der IdP auf die Anfragen dieser Erweiterung antworten kann. (vgl. [Barto-2006] Seite 5)

4.4.2 GridShib for Shibboleth

GridShib for Shibboleth erweitert den Identity Provider des Shibboleth-Frameworks um Funktionen, die es einem Grid-System ermöglichen die Attribute eines Benutzers abzuholen. Diese Erweiterung kann nur auf einem Identity Provider in der Version 1.3 installiert werden. Die Hauptaufgabe dieser Erweiterung ist die Abbildung des Distinguished Name (DN), aus einer AttributeQuery, auf den principal name des Benutzers im IdP. In Grid-Systemen findet die Authentifizierung der Benutzer mittels X.509-Zertifikaten statt. Das Grid-System entnimmt dem X.509-Zertifikat den distinguished name des Benutzers, dieser wird in der AttributeQuery eingebaut und dient zur Identifizierung des Benutzers, deshalb ist das namemapping notwendig. Das Mapping wird vom Administrator in einer Datei oder in einer Datenbank abgelegt. Für jeden Benutzer muss ein Eintrag erstellt werden, dies hat zur Folge, dass der Administrationsaufwand erheblich ist und deshalb ungeeignet für eine große Anzahl von Benutzern ist. Eine andere Variante für das Mapping ist die Certificat Registry dieser Erweiterung. Bei der Certificat Registry können die Benutzer ihre End Entity Certificate hochladen. Da die EECs einem Benutzer des IdP zugeordnet werden, kann der IdP den übermittelten Distinguished Name mit den EECs in der Certificat Registry abgleichen und

einen Benutzer identifizieren. (vgl. [Barto-2006] Seite 5)

4.4.3 GridShib SAML Tools

Die GridShib SAMLTools stellen Funktionen für die Verwendung der Security Assertion Markup Language bereit, es werden auch Funktionen zur Einbindung der SAML-Assertions in X.509-Zertifikate mitgeliefert. Dafür werden in den GridShib SAMLTools sechs Komponenten bereitgestellt (vgl. [GlobA-2008b]).

- **SAML Security Info Tool:** Dieses Konsolenprogramm durchsucht Globus-Credentials auf SAML-Assertions und gibt diese auf der Konsole aus.
- **X.509 Binding Tool:** Zum Einfügen von SAML-Assertions in die unkritische Erweiterung von X.509-Proxy-Zertifkaten kann dieses Konsolenprogramm eingesetzt werden.
- **SAML Assertion Issuer Tool:** Zur Erzeugung von SAML-Assertions kann dieses Konsolenprogramm verwendet werden. Es können zwei Arten von SAML-Assertions erzeugt werden, ein AuthenticationStatement oder ein AttributeStatement. Dieses Programm hat auch die Möglichkeit die SAML-Assertions in die unkritische Erweiterung eines X.509-Proxy-Zertifkat einzubinden. In Zukunft soll dieses Konsolenprogramm einen vollfunktionsfähigen Attribute Resolver enthalten, dieser soll mit einem Attribute Resolver des Identity Providers vergleichbar sein.
- **SAML Query Client:** Dieses Konsolenprogramm kann die Attribute eines Benutzer von einer SAML Attribute Authority abfragen. Der Response der Attribute Authority wird validiert und anschließend ausgegeben. Optional kann das SAML-Assertion in ein X.509-Proxy-Zertifikat eingefügt werden. In der aktuellen Version kann der SAML Query Client nur einen begrenzten Umfang von SAML-Querys absetzen, dieses soll in den nächsten Versionen verbessert werden.
- **GridShib Common:** Diese Java-API ist für Entwickler von Webportalen gedacht, diese ermöglicht es SAML in die Authentifizierungs- und Autorisierungs-Infrastruktur des Webportals einzubinden. In dieser API ist das GridShib Security Framework enthalten. Diese ermöglicht das erzeugen und verarbeiten von SAML-Assertions, die in ein X.509-Zertifikat eingebunden sind.
- **Globus SAML Library:** Diese Library ist eine Erweiterung der OpenSAML 1.1. Library. Die Globus SAML Library enthält die Spezifikationen „Metadata Profile for the OASIS Security Assertion Markup Language (SAML) V1.x“ (siehe [OASIS-2007a]) und „Metadata Extension for SAML V2.0 and V1.x

Query Requesters“ (siehe [OASIS-2007b]). Es wird auch ein weiteres SAML-Profil unterstützt „Subject-based Profiles for SAML V1.1 Assertions“ (siehe [OASIS-2007c]), da dieses für die Realisierung der GridShib-Erweiterung benötigt wird.

4.4.4 GridShib Certificate Authority

Die GridShib Certificate Authority (CA) ist eine Certificate Authority, die vom Benutzer über eine Webseite angesprochen wird. Der Benutzer authentifiziert sich gegenüber der CA und kann mittels eines Java Web Start-Programms ein Short-Lived Certificate (SLC) herunterladen. Das Java Web Start-Programm sorgt dafür, dass das Zertifikat über eine sichere Verbindung heruntergeladen wird und deshalb nicht abgefangen werden kann. Die GridShib Certificate Authority kann die Short-Lived Certificates auf zwei Arten erzeugen. Zum Einen kann das Programm OpenSSL direkt von der GridShib CA zum Erzeugen der Zertifikate eingesetzt werden oder es kann ein myProxy-Server als Backend-System verwendet werden. Bei dieser Variante wird der myProxy-Server als CA installiert. Dieser erzeugt die Zertifikate und leitet diese an die GridShib CA weiter. In beiden Varianten kann die GridShib CA mit der Hilfe der SAML-Tools in die unkritische Erweiterung des Zertifikates eine SAML-Assertion einbinden. (vgl. [GlobA-2008c])

4.5 Die Interoperabilität zwischen dem Shibboleth-Framework und Grid-Systemen

Das Shibboleth-Framework und Grid-System setzen sehr unterschiedliche Konzepte im Bereiche der Authentifizierung und Autorisierung um. Die Grid-Systeme setzen sehr stark auf Vertrauen (Web of Trust), da diese X.509-Zertifikate zur Authentifizierung einsetzen. Weitere Details zu „Web of Trust“ wurden schon im Kapitel 3.5 erörtert. Das Shibboleth-Framework setzt für die Authentifizierung keine X.509-Zertifikate ein, sondern verwendet Benutzernamen und Passwörter. Bei einer Authentifizierung durch Wissen (Passwort) kann der Benutzer seine Identität nachweisen, in dem er das Wissen (Passwort) an das Authentifizierungssystem übermittelt. Im Shibboleth-Framework wird das Konzept des „Web of Trust“ nur für die Kommunikation zwischen den Systemen einer Shibboleth-AAI eingesetzt. Das „Web of Trust“ einer Shibboleth-AAI wird durch die Metadaten-Datei eingegrenzt. Nur System, die mit ihrem X.509-Zertifikat in dieser Datei aufgeführt sind, gehören zu diesem „Web of Trust“. Nur AuthenticationRequests können ohne Zugehörigkeit zum „Web of Trust“ an die Identity Provider gestellt werden, allerdings muss sich an dieser Stelle der Benutzer mit seinem Benutzernamen und seinem Passwort authentifizieren. Das GridShib-Projekt

versucht die unterschiedlichen Konzepte der Authentifizierung von Benutzern miteinander zu verknüpfen, die die Grid-Systeme und das Shibboleth-Framework einsetzen. Dies ist nötig damit das Shibboleth-Framework und Grid-Systeme zusammenarbeiten können. In der aktuellen Version der GridShib-Erweiterungen muss für die Interoperabilität von Grid-Systemen und dem Shibboleth-Framework, ein erheblicher administrativer Aufwand investiert werden. Da es nötig ist für jeden Benutzer den DN seines X.509-Zertifikats auf den Benutzernamen im LDAP-Verzeichnis abzubilden. Die Erweiterungen des GridShib-Projektes können erheblich zur Lösung der Aufgabenstellung dieser Arbeit beitragen, da diese die Problematik der Interoperabilität von Grid-Systemen und dem Shibboleth-Framework lösen.

5 Konzeption der Anbindung

In diesem Kapitel wird die Konzeption für die Anbindung von Shibboleth an eine Grid-Ressource behandelt. Die Architektur des Systems wird vorgestellt. Es werden die Anpassungen der Grid-Komponenten und die Konfiguration der AAI beleuchtet.

5.1 Erste Grundideen

In diesem Absatz werden die ersten Ideen für dieses Konzept erläutert. Diese Ideen schildern unterschiedlicher Vorgehensweisen um die Attribute des Benutzers vom Identity Provider in die Grid-Ressource zu bekommen.

1. Mit dem Grid-Client authentifiziert sich der Benutzer am Identity Provider und holt die Attribute ab. Das AuthenticationStatement und das AttributeAssertion werden vom Grid-Client in ein Proxy-Zertifikat eingebettet. Die Grid-Ressource extrahiert die Attribute des Benutzers aus dem AttributeAssertion, dass in dem Proxy-Zertifikat des Benutzers enthalten ist. Der Vorteil dieses Ansatzes ist, dass die Grid Security Infrastructure des Globus Toolkit nicht verändert werden muss. Diese Idee hat den Nachteil, dass der Grid-Client die gesamte Kommunikation mit dem Identity-Provider abwickelt und die Informationen für die Autorisierung sammelt. Durch diesen Umstand ist es einfacher die Autorisierungsinformationen zu manipulieren, deshalb wurde diese Variante verworfen.

2. Es werden die Erweiterungen des GridShib-Projektes eingesetzt um die Attribute des Benutzers in die Grid Security Infrastructure des Globus Toolkit zuholen. Die Grid-Ressource holt sich die Attribute des Benutzers aus der Grid Security Infrastructure des Globus Toolkit. Der Vorteil dieses Konzeptes ist, dass die selben Informationen, die Benutzerattribute, zur Autorisierung des Benutzers in der

Grid-Middleware und in der Grid-Ressource verwendet werden. Ein Nachteil ist, dass man von der Implementierung des GridShib-Projektes abhängig ist und deshalb nicht immer die neuste Version des Shibboleth-Frameworks oder des Globus Toolkits einsetzten kann. Da die Erweiterungen des GridShib-Projektes für die neuen Versionen angepasst werden muss.

In diesem Konzept wird die zweite Idee weiter ausgearbeitet und detailliert vorgestellt. Zuerst wird die Architektur vorgestellt und die Anpassungen erläutert, die an den einzelnen Komponenten vorgenommen werden müssen.

5.2 Die Architektur

In diesem Abschnitt wird die Architektur erläutert, die in diesem Konzept erarbeitet wurde. Es wird auf die Komponenten der Architektur eingegangen und welche Aufgaben diese in dem Konzept erfüllen. Für die Realisierung des Konzepts werden folgende Software Komponenten benötigt:

- Certificate Authority
- Shibboleth Identity Provider 1.3
- LDAP-Server
- Globus Toolkit 4.x
- GridShib for Shibboleth
- Gridshib for Globus Toolkit
- GridShib SAML Tools

Der Benutzer erhält sein End Entity Certificate (EEC) von einer Certificate Authority. Auf welchem Weg der Benutzer das EEC erhält, wird hier nicht näher spezifiziert, da es dafür unterschiedliche Wege gibt. Zum einen kann man das Zertifikat auf einer Smartcard erhalten oder auch über eine sichere Verbindung beim Anbieter herunterladen. Für die Authentifizierung im Grid-System benötigt der Benutzer ein RFC 3820 konformes Proxy-Zertifikat, dieses wird mit dem End Entity Certificate des Benutzers durch den Grid-Client erzeugt. Der Grid-Client beinhaltet einen Shibboleth-Client, der mit der Hilfe der GridShib SAML Tools die Authentifizierung des Benutzers am Identity Provider durchführt. Die GridShib SAML Tools werden auch für die Einbindung der AuthenticationAssertions in das Proxy-Zertifikat benötigt. Für jede Organisation muss ein Shibboleth Identity Provider in der Version 1.3 eingerichtet werden, diese müssen mit dem LDAP-Server der Organisation verbunden werden, damit

die Authentifizierung gegen diesen LDAP-Server durchgeführt werden kann und der IdP die Attribute des Benutzers an den Grid-Service-Provider weitergeben kann. Der SSO-Service des Identity Providers wird durch die Zugriffskontrolle eines Apache-Servers (2.2.x) geschützt. Die Zugriffskontrolle wird für die Authentifizierung der Benutzer mit dem LDAP-Server der Organisation verbunden, dafür wird das Apache-Modul „mod_authnz_ldap“ verwendet. Das Grid-System basiert auf der Grid-Middleware Globus-Toolkit 4.0, es wird die Java-Implementierung eingesetzt, der „Java Web Service Core“. Damit das Shibboleth-Framework und das Globus Toolkit 4.0 zusammenarbeiten, werden die Erweiterungen des GridShib-Projektes eingesetzt. Die Erweiterung „GridShib for Shibboleth“ wird auf dem Identity Provider 1.3 des Shibboleth-Frameworks installiert. Auf dem „Java Web Service Core“ des Globus Toolkit 4.0 wird die Erweiterung „GridShib for Globus Toolkit“ installiert. Mit diesen beiden Komponenten wird der Austausch der Attribute zwischen dem Identity Provider und dem Authorization Framework des Globus Toolkit möglich. Eine erste Autorisierung erfolgt im Authorization Framework anhand der Attribute des Benutzers. Die Grid-Ressource hat eine eigene Zugriffskontrolle, für diese werden die Attribute des Benutzers benötigt. Deshalb müssen die Attribute des Benutzers vom Authorization Framework des Globus Toolkit zur Grid-Ressource übermittelt werden. Die Kommunikation zwischen den Komponenten wird durch die Verwendung von SSL/TLS-Tunneln verschlüsselt, damit die übermittelten Daten vor unberechtigtem Zugriff und Manipulationen geschützt sind. Die Abbildung 9 zeigt die Architektur, die in diesem Konzept vorgeschlagen wird.

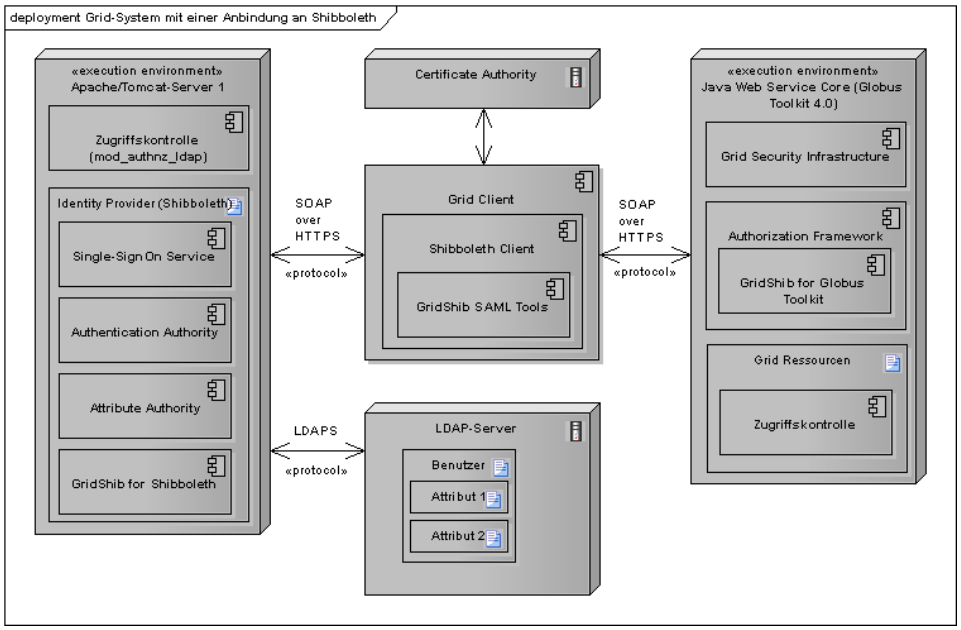

Abbildung 9: Architektur der Anbindung des Shibboleth-Frameworks an das Grid-System

5.3 Implementierungsdetails des Konzepts

In diesem Unterkapitel werden die Anpassungen erläutert, die an einzelnen Komponenten vorgenommen werden müssen, um das Konzept umsetzen zu können. Vereinzelt werden wichtige Konfigurationsdetails vermittelt.

5.3.1 Identity Provider

Auf dem Identity Provider ist die GridShib-Erweiterung „GridShib for Shibboleth“ installiert, diese stellt einen Namemapping-Service bereit, dieser muss den DN eines Benutzers auf einen Benutzernamen abbilden. Das Mapping muss in einer Text-Datei vom Administrator des Systems erstellt werden. Für jedes Mapping muss ein Eintrag in der Text-Datei nach folgendem Schema erstellt werden: `"CN=Test User,OU=People,DC=example,DC=org" testuser`. Für jeden Benutzer muss ein Eintrag in einer neuen Zeile erstellt werden. Zuerst wird der DN aus dem Proxy-Zertifikat notiert und dann folgt nach einem Leerzeichen der Benutzername. In der Konfigurationsdatei des Identity Providers muss das Namemapping auf die GridShib-Erweiterung umgestellt werden, dazu muss ein Eintrag nach folgendem Beispiel in der Konfigurationsdatei erstellt werden und der alte NameMapping-Eintrag entfernt werden.

```
1 <NameMapping
2     xmlns="urn:mace:shibboleth:namemapper:1.0"
3     id="x509"
4     format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
5     class="edu.internet2.middleware.shibboleth.common.provider.
          GridShibNameIdentifierMapping"/>
```

NameMapping-Eintrag für der GridShib-Erweiterung

Die Weitergabe der Attribute wird über die Attribute Release Policy eingeschränkt, damit kann sichergestellt werden, dass die Grid-Systeme nur die Attribute übermittelt bekommen, die diese für die Autorisierung benötigen. Die Attribute sollten aus Sicherheitsgründen nach dem Prinzip „So viele Informationen wie nötig, so wenig Informationen wie möglich!“ Bereitgestellt werden.

5.3.2 Grid-Client

Der verwendete Grid-Client muss um einige Funktionen erweitert werden. Mit der „The Java Commodity Grid Kit (v1.2)“-API [GlobA-2008g] wird ein Proxy-Zertifikat für den Benutzer erzeugt. Zu diesem Zweck muss der Benutzer das Passwort für die Entschlüsselung seines privaten Schlüssels eingeben, da dieser für die Signierung des Proxy-Zertifikats benötigt wird. Ein Shibboleth-Client muss implementiert und in

den Grid-Client integriert werden, dieser übernimmt die Authentifizierung am Identity Provider. Die Implementierung wird mit der Hilfe der GridShib SAML Tools umgesetzt. Aus den SAML-Tools wird die Java-API „GridShib Common“ verwendet. Diese Java-API stellt Klassen für die Erzeugung und Verwendung der benötigten SAML-Assertions bereit, zum Beispiel für das Einbinden des AuthenticationStatements in das Proxy-Zertifikat. Da der Identity Provider von einem Apache-Server vor unbefugtem Zugriff geschützt wird, muss sich der Benutzer am Apache Server authentifizieren, dazu wird das Authenticate-Feld im HTTP-Header eingefügt und mit dem Benutzernamen und dem Passwort versehen. Mit dem so präparierten HTTP-Request auf den Identity Provider, kann der Benutzer authentifiziert werden und bekommt vom Identity Provider ein AuthenticationStatement, dass die Authentifizierung bestätigt. Das AuthenticationStatement enthält die ID des Identity Providers, diese wird vom Grid-Service Provider benötigt um das „Identity Provider Discovery Problem“ zu lösen. Deshalb wird das AuthenticationStatement als unkritische Erweiterung in das Proxy-Zertifikats eingebunden. Dieses Proxy-Zertifikat wird zur Authentifizierung an der Grid Security Infrastructure verwendet.

5.3.3 Grid-Middleware

In das Authorization Framework der Grid-Middleware Globus Toolkit wird die GridShib-Erweiterung „GridShib for Globus Toolkit“ installiert. Diese ermöglicht dem Authorization Framework den Identity Provider des Benutzers zukontaktieren. In der Shibboleth-Metadatendatei, die allen Identity Providern und Grid-Service Providern zur Verfügung gestellt wird, muss ein Eintrag für den Grid-Service Provider erstellt werden. Im GridShib-Projekt werden alle Installationen des Globus Toolkit, auf denen die Erweiterung „GridShib for Globus Toolkit“ eingerichtet ist, als Grid-Service Provider bezeichnet. Im Anhang ist ein Beispiel für einen Eintrag eines Grid-Service Provider vorhanden. In diesem Eintrag muss das Zertifikat eingetragen werden, dass zu der Grid-Ressource gehört und der Grid-Service Provider muss als AttributeConsumingService definiert werden. Dies ist notwendig, damit der Identity Provider auf eine AttributeQuery von diesem Grid-Service Provider antwortet.

Bevor eine Autorisierungsentscheidung vom Authorization Framework getroffen werden kann, liest eine Komponente der GridShib-Erweiterung, der SAMLAssertionPushPIP, alle SAML-Assertions aus dem Proxy-Zertifikat ein. Im Proxy-Zertifikat befindet sich ein AuthenticationStatement, diesem wird die ID des ausstellenden Identity Providers entnommen und im Sicherheitskontext des Benutzers gespeichert. Dieser Sicherheitskontext wird beim Verbindungsaufbau etabliert, wenn der Benutzer erfolgreich authentifiziert wurde. Durch die Verwendung der GridShib-Erweiterung wird für den Benutzer ein „SAML Security Context“ im Globus-System etabliert. Die ID des

Identity Providers wird vom SAMLQueryPIP der GridShib-Erweiterung verwendet um den Identity Provider des Benutzers zu kontaktieren und die Attribute des Benutzers mit einer AttributeQuery abzuholen. Der Identity Provider des Benutzers antwortet auf die AttributeQuery mit einem AttributeResponse, der die Attribute des Benutzers enthält. Der AttributeAcceptancePIP der GridShib-Erweiterung extrahiert aus dem AttributeResponse die Attribute des Benutzers und speichert die Attribute, die für die Umsetzung der Sicherheitsrichtlinie benötigt werden, im Sicherheitskontext des Benutzers. Da am Identity Provider eine Attribute Release Policy eingerichtet ist, sollten keine unbenötigten Attribute in dem AttributeResponse enthalten sein. Es ist wichtig das der AttributeAcceptancePIP so konfiguriert wird, dass dieser nicht nur die Attribute ausliest, die zur Autorisierung im Authorization Framework benötigt werden, sondern auch die Attribute, die im Autorisierungssystem der Grid-Ressource benötigt werden. Der SAMLAttributePDP der GridShib-Erweiterung führt mittels der definierten Sicherheitsrichtlinie und der Attribute eine Zugriffskontrollentscheidung durch. In der Sicherheitsrichtlinie des SAMLAttributePDP wird festgelegt, welche Attribute mit welchen Werten zulässig für den Zugriff auf die gewünschte Ressource sind. Der SAMLAttributePDP ist so konzipiert, dass eine der definierten Attribute mit einem gültigen Wert ausreicht, um eine positive Zugriffskontrollentscheidung (permit) zu erhalten. Dies bedeutet, dass zwischen den Attributen eine ODER-Verknüpfung besteht. Wenn man aber eine UND-Verknüpfung benötigt, müssen mehrere SAMLAttributePDPs mit einer Sicherheitsrichtlinie erzeugt werden. Alle definierten SAMLAttributePDPs übermitteln ihre Zugriffskontrollentscheidung an den Master-PDP des Authorization Frameworks. Wenn der Master-PDP keine negative Zugriffskontrollentscheidung (deny) von den SAMLAttributePDPs erhält, wird dem Benutzer der Zugriff auf gewünschte Grid-Ressource gewährt. Im Rahmen dieses Konzeptes bildet dieses die erste Stufe der Zugriffskontrolle. Der zweite Teil der Zugriffskontrolle befindet sich in der Grid-Ressource.

5.3.4 Grid-Ressource

Die Grid-Ressource stellt ein eigenes Autorisierungssystem bereit, dieses ist ein rollenbasiertes Autorisierungssystem (RBAC). Die Entscheidung, welche Benutzerrolle einem Benutzer zugewiesen wird, soll anhand der Attribute des Benutzers getroffen werden. Dies sind die Attribute, die vom Identity Provider abgeholt wurden und vom Authorization Framework dem Sicherheitskontext des Benutzers zugeordnet wurden. Diese Attribute werden von der Grid-Ressource aus dem Sicherheitskontext des Benutzers abgeholt. Im ersten Schritt muss die Grid-Ressource den MessageContext der aktuellen Verbindung zwischen der Grid-Ressource und dem Benutzer erhalten. Der „Globus Java Web Service Core“ basiert auf dem Apache-Axis-Server. Die Java-

API des Apache-Axis-Servers stellt einen Befehl bereit, mit dem man den benötigten MessageContext erhält. Ein Programmierbeispiel für das Abholen der Attribute aus dem Sicherheitskontext des Benutzers ist das folgende Beispiel:

```
1  import org.apache.axis.MessageContext;
2  import org.globus.gridshib.gt.authorization.attributes.SAMLAttributeInformation;
3  import java.util.Iterator;
4  import java.util.Set;
5  import javax.security.auth.Subject;
6
7   ...
8  SAMLAttributeInformation benutzerAttribute = null;
9  MessageContext ctx = MessageContext.getCurrentContext();
10 Subject user = (Subject)ctx.getProperty(org.globus.wsrf.impl.security.
       authentication.Constants.PEER_SUBJECT);
11 Set userCredentials = user.getPublicCredentials();
12  Iterator  credentialsIterator  = peerCredentials.iterator();
13
14 while (credentialsIterator.hasNext) {
15    Object credential = credentialsIterator.next();
16    if (credential instanceof SAMLAttributeInformation) {
17       benutzerAttribute = credential;
18    }
19 }
20  ...
```

Ein Code-Beispiel für das Abholen der Attribute von der Grid-Ressource.

Um an die Attribute des Benutzers zukommen, benötigt man zuerst den Kontext der aktuellen Verbindung zwischen dem Benutzer und der Grid-Ressource. Mit dem Befehl `getCurrentContext()` bekommt man vom Apache-Axis-Server den aktuellen MessageContext, der diesem Thread zugeordnet ist. Aus diesem MessageContext holt man den Benutzer (Subjekt). Der Benutzer ist eine Eigenschaft dieses MessageContext und kann mit dem Befehl `getProperty(... .Constants.PEER_SUBJECT)` aus dem MessageContext geholt werden, dafür wird eine Konstante benötigt, die das Web Service Ressource Framework bereitstellt. Im Subject sind auch die Credentials des Benutzers gespeichert, dazugehören auch die Attribute des Benutzers. Mit dem Befehl `getPublicCredentials()` bekommt man alle öffentlichen Credentials des Benutzers. Diese muss man nach einer SAMLAttributeInformation durchsuchen. In der SAMLAttributeInformation sind die Attribute im SAML-Format gespeichert. Diese Attribute können zur Autorisierung verwendet werden.

Das Autorisierungssystem der Grid-Ressource definiert Benutzer, Benutzerrollen und die Berechtigungen. Den Benutzerrollen werden Berechtigungen zugewiesen, die für die Abarbeitung bestimmter Aufgaben benötigt werden. Den Benutzern werden Benutzerrollen zugewiesen und diese erhalten damit die Berechtigungen, die der

Benutzerrolle zugewiesen sind. Das Zuweisen der Benutzerrollen muss vom Administrator manuell durchgeführt werden. Dieser Schritt soll mit den Attributen der Benutzer automatisiert werden und damit den administrativen Aufwand reduzieren. Damit dies realisiert werden kann, muss das Autorisierungssystem erweitert werden. Bei jeder Autorisierungsanfrage muss entschieden werden, welche Benutzerrolle dem Benutzer zugewiesen wird, da keine feste Zuordnung zwischen einem Benutzer und einer Benutzerrolle existiert. Dazu muss eine Zuordnung zwischen den Benutzerrollen und den benötigten Benuterattribute mit den benötigten Werten erstellt werden. In diesem Konzept wird festgelegt, dass ein Benutzer alle Attribute benötigt, die einer Benutzerrolle zugeordnet wurden, dieses stellt eine UND-Verknüpfung der Attribute da. Allerdings muss der Benutzer nur einen der benötigten Werte des Attributs besitzen, dies stellt eine ODER-Verknüpfung da. Zum Erstellen der Zuordnung von Benutzerattributen zu Benutzerrollen wird eine Hashmap definiert, in der den Benutzerrollen eine weitere Hashmap zugeordnet wird. In der zweiten Hashmap werden den Attributen eine Liste mit erlaubten Werten zugeordnet. Die gesamte Hashmap kann der Administrator mittels einer grafischen Benutzeroberfläche verwalten. Die Zuordnung wird in einer Datenbank gespeichert.

Bei einer Autorisierungsanfrage müssen alle Benutzerrollen der Hashmap abgearbeitet werden. Dabei wird überprüft ob der Benutzer die benötigten Attribute besitzt und ob diese einen der zulässigen Werte haben. Wenn dies der Fall ist, wird dem Benutzer diese Benutzerrolle zugeordnet. Anhand der Berechtigungen, die der Benutzer durch die zugeordnete Benutzerrolle bekommen hat, wird dem Benutzer Zugriff auf die Grid-Ressource gewährt.

5.3.5 Auswahl der LDAP-Attribute

Für die Verwendung von Shibboleth muss eine Auswahl von geeigneten LDAP-Attributen getroffen werden. Diese gibt der Identity Provider in einem AttributeResponse, als Antwort auf eine AttributeQuery, an das Grid-System weiter. Grundsätzlich kann Shibboleth alle Attribute die sich im LDAP-Verzeichnis befinden weitergeben, natürlich sollte das gespeicherte Passwort nicht durch das Shibboleth-Framework verteilt werden. Man kann auch eigene Attribute in LDAP-Verzeichnis definieren, dies bietet ein Höchstmaß an Flexibilität. Der Shibboleth Identity Provider kann auch statische Attribute verteilen, diese kann man in den Konfigurationsdateien des Identity Providers definieren und einen festen Wert zuteilen. Die statischen Attribute werden allen Benutzer zugeordnet, wenn in der Attribute Release Policy nichts Gegenteiliges definiert wird. Besonders eignen sich LDAP-Attribute, die jedes LDAP-System im Standard bereitstellt. Deshalb sollten hauptsächlich Attribute aus dem core.schema verwendet werden.

Attribut	Beschreibung	Object Identifier
o	Die Organisation, der der Benutzer angehört.	2.5.4.10
ou	Organisationseinheiten (z. B. Abteilung)	2.5.4.11
title	Titel (z. B. Aufgabe in diesem Unternehmen, Entwickler, SW-Tester, usw.)	2.5.4.12
mail	E-Mailadresse des Benutzers	0.9.2342.19200300.100.1.3

Tabelle 6: Die verwendeten LDAP-Attribute

Für die hier vorgeschlagenen Attribute muss das verwendete LDAP-Verzeichnis die beiden LDAP-Schemata inetorgperson.schema (RFC 2798) und core.schema (RFC 2256) verwenden. Damit ein Benutzer die benötigten LDAP-Attribute erhält, müssen ihm die Objektklassen organizationalPerson und inetOrgPerson zugeordnet werden. Es wird eine weltweit eindeutige ID für die jeweiligen Benutzer benötigt, dafür bietet sich die E-Mailadresse des Benutzers an. Die E-Mailadresse (mail) stellt die Objektklasse inetOrgPerson bereit, diese Objektklasse erlaubt auch das Attribut „o“ bei einem Benutzer zuverwenden. Im core.schema ist dies nicht möglich. Auf der Ebene der Grid Security Infrastructure kann das Attribut „o“ und „ou“ für die Autorisierung verwendet werden. In der Grid-Ressource wird dem Benutzer eine Benutzerrolle zugeordnet, dafür werden die Attribute verwendet. Mit den Attributen „o“, „ou“ und „title“ werden den Benutzern die allgemeinen Benutzerrollen zugeordnet, wie zum Beispiel Entwickler, Gast, Anwender. Das Attribut „mail“ kann für die Zuordnung eines Benutzer, zu einer speziellen Benutzerrolle eingesetzt werden, wie z. B. Administrator für eine Grid-Ressource. Deshalb wird ein eindeutiges Attribut des Benutzers benötigt.

5.4 Verlauf der Authentifizierung und Autorisierung

In diesem Unterkapitel wird der Kommunikationsverlauf detailliert dargestellt, der bei der Authentifizierung und Autorisierung in diesem Konzept benötigt wird.

In der Abbildung 10 ist der Kommunikationsablauf bei der Authentifizierung in Grid-Systemen, mit der Verwendung des Shibboleth-Frameworks dargestellt.

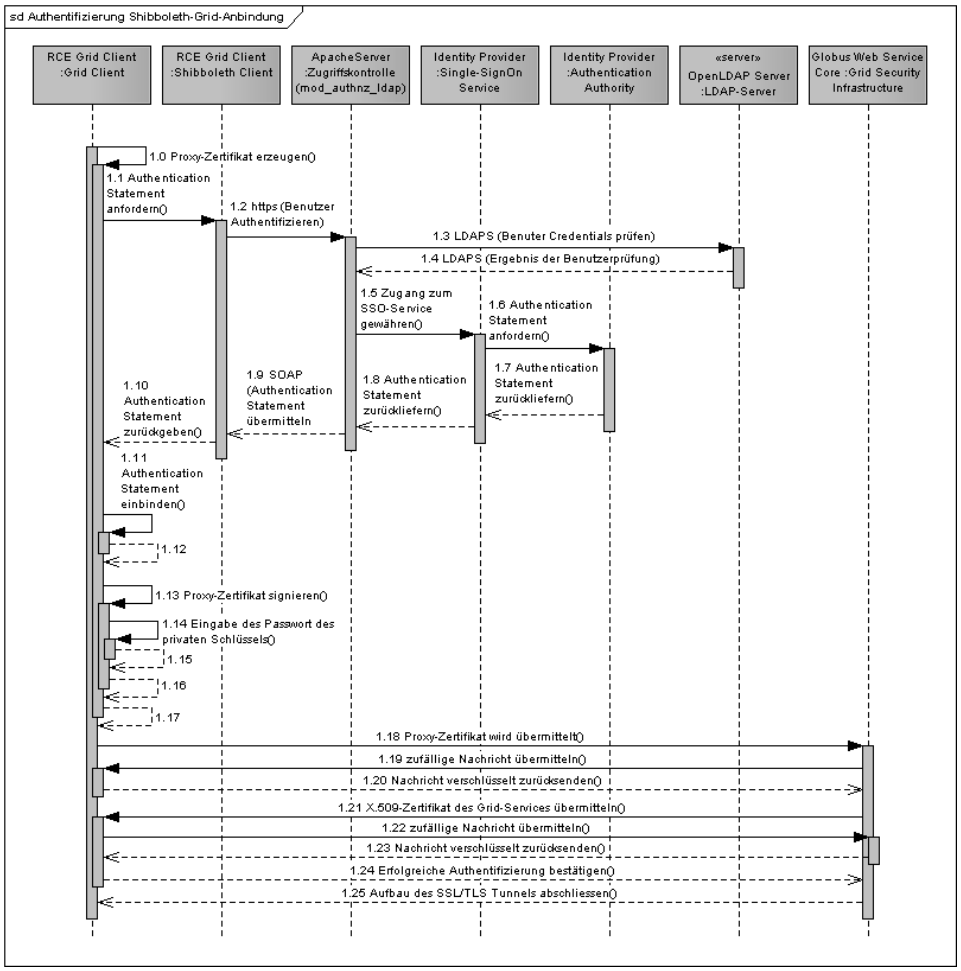

Abbildung 10: Verlauf der Authentifizierung in diesem Konzept

Für die Authentifizierung des Benutzers muss ein Proxy-Zertifikat erstellt werden (1.1 - 1.15). In das Proxy-Zertifikat soll eine AuthenticationStatement eingebunden werden, dafür wird der Shibboleth-Client eingesetzt (1.1 - 1.10). Der Shibboleth-Client kontaktiert den SSO-Service des Identity Providers, dieser Request wird von der Zugriffskontrolle des Apache-Servers abgefangen (1.2). Da durch den Shibboleth-Client das Authenticate-Feld im HTTP-Header mit dem Benutzernamen und dem Passwort ausgefüllt ist, kann der Apache-Server die Benutzer-Credentials gegen das LDAP-Verzeichnis prüfen (1.3 - 1.4). Wenn der Benutzer berechtigt ist auf den Identity Provider zuzugreifen, wird er auf den SSO-Service des IdP weitergeleitet (1.5). Damit der IdP die Authentifizierung durchführen kann, kontaktiert der SSO-Service die Authentication Authority (1.6). Die Authentication Authority wertet die Apache-Umgebungsvariable `$REMOTE_USER` aus und erstellt ein AuthenticationStatement für den Benutzer (1.7). Das AuthenticationStatement wird mit einer SOAP-Nachricht vom IdP zum Shibboleth Client übertragen (1.8 - 1.9). Der Grid-Client bindet das AuthenticationStatement in das Proxy-Zertifikat ein (1.10 - 1.12). Damit das Proxy-Zertifikat fertiggestellt werden kann, muss das Proxy-Zertifikat signiert werden (1.13 - 1.7), dafür muss der Benutzer

das Passwort des privaten Schlüssels eingeben (1.14 - 1.15). Nun kann der Benutzer sich mit dem Proxy-Zertifikat am Grid-System authentifizieren. Die Authentifizierung findet in drei Schritten statt (1.18 - 1.20). Der Benutzer übermittelt sein Proxy-Zertifikat (1.18), diese validiert der GSI und sendet dem Benutzer eine zufällige Nachricht zu, mit der Bitte, diese Nachricht zu verschlüsseln (1.19). Der Benutzer verschlüsselt die Nachricht mit seinem privaten Schlüssel und sendet diese an den GSI (1.20). Der GSI entschlüsselt die Nachricht mit dem öffentlichen Schlüssel des Benutzers und vergleicht diese mit der übermittelten Nachricht. Wenn beide Nachrichten gleich sind, ist der Benutzer erfolgreich authentifiziert. Nun muss sich die Grid-Ressource beim Benutzer authentifizieren, dies läuft nach demselben Schema wie die Authentifizierung des Benutzers ab (1.21 - 1.23). Die erfolgreiche Authentifizierung wird dem GSI mitgeteilt (1.24) und der SSL/TLS-Tunnel wird aufgebaut (1.25).

In der Abbildung 11 ist der Kommunikationsablauf bei der Autorisierung in Grid-Systemen, mit der Verwendung des Shibboleth-Frameworks und mit einer zweiten Autorisierung in der Grid-Ressource dargestellt.

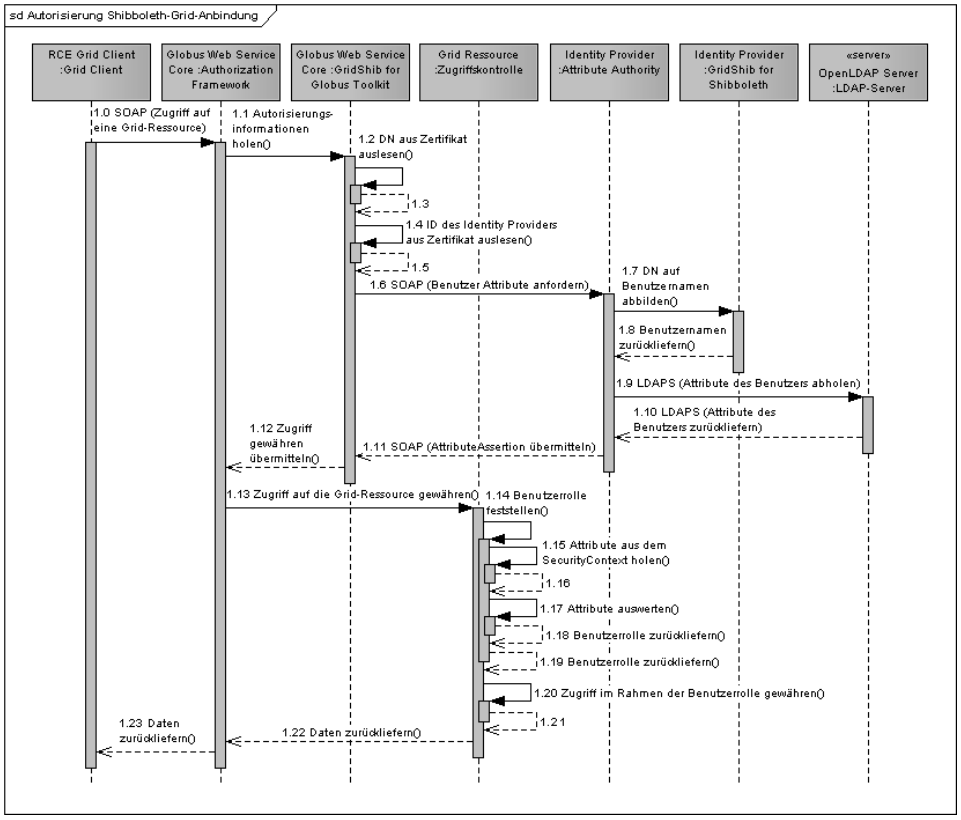

Abbildung 11: Verlauf der Autorisierung in diesem Konzept

Die Abbildung 11 ist die Fortsetzung der Kommunikation aus der Abbildung 10, in dieser wurde die Authentifizierung des Benutzers erläutert. In der Abbildung 11 wird die Autorisierung dargestellt. Der Benutzer versucht mit dem Grid-Client auf eine Grid-Ressource zuzugreifen, wird aber von dem Authorization Framework des Globus Tool-

kit abgefangen (1.0). Das Authorization Framework holt sich die Autorisierungsinformation, die es für eine Zugriffskontrollentscheidung benötigt vom Identity Provider ab. Dafür werden die Komponenten der Erweiterung „GridShib for Globus Toolkit“ verwendet (1.1). Als Erstes wir der DN aus dem Proxy-Zertifikat ausgelesen (1.2, 1.3). Im nächsten Schritt wird die ID des Identity Providers aus dem Zertifikat ausgelesen (1.4, 1.5). Eine AttributeQuery mit dem DN des Benutzers wird an den Identity Provider geschickt (1.6). Die Attribute Authority verwendet die Erweiterung „GridShib for Shibboleth“ als Namemapping-Service und übermittelt diesem den DN des Benutzers (1.7). Der Namemapping-Service liefert den Benutzernamen zurück, der zu der DN gehört (1.8). Die Attribute Authority fordert die Attribute des Benutzers von einem LDAP-Server an (1.9). Der LDAP-Server sendet die Attribute des Benutzers an die Attribute Authority (1.10). Die Attribute Authority sendet eine AttributeResponse an die Komponenten der Erweiterung „GridShib for Globus Toolkit“ (1.11). Diese übermitteln dem Authorization Framework, dass der Benutzer die benötigten Attribute besitzt (1.12). Das Authorization Framework gewährt dem Benutzer den Zugriff auf die Grid-Ressource (1.13). In der Grid-Ressource landet der Request des Benutzers in der Zugriffskontrolle, diese muss zuerst die Benutzerrolle des Benutzers feststellen (1.14 - 1.19). Die Attribute des Benutzers werden aus dem Sicherheitskontext des Benutzers ausgelesen (1.15, 1.16). Die Attribute des Benutzers werden ausgewertet (1.17) und einer Benutzerrolle zugeordnet (1.18, 1.19). In nächsten Schritt bekommt der Benutzer Zugriff auf die Grid-Ressource, aber nur in dem Umfang, wie es die Berechtigungen der Benutzerrolle erlauben (1.20 - 1.23).

Die Anwendungsfälle „Proxy-Zertifikat erzeugen“ und „Authentifizierung“ werden in der Abbildung 10 gezeigt. Von der Sequenznummer 1.0 bis 1.17 ist die Erzeugung der Proxy-Zertifikate dargestellt. Die Authentifizierung wird von der Sequenznummer 1.18 bis 1.25 illustriert. In den Anwendungsfällen „Benutzer-Credentials delegieren“ und „Zertifikat erneuern“ muss das erzeugt Proxy-Zertifikat, dass zum Delegieren der Benutzer-Credentials eingesetzt wird, ein AuthenticationStatement enthalten. Dies ist notwendig, damit die Autorisierung auch mit delegierten Proxy-Zertifikaten funktioniert.

6 Projekte im Bereich des Themengebietes

In diesem Kapitel werden Projekte vorgestellt, die sich mit der Thematik Authentifizierung und Autorisierung in Grid-Systemen beschäftigen.

6.1 IVOM

Das Projekt „Interoperabilität und Integration der VO-Management Technologien im D-Grid“ (IVOM) ist ein Projekt der D-Grid Initiative. Die D-Grid Initiative hat zum Ziel Projekte in den Bereichen Grid-Computing, e-Learning und Wissensvernetzung zu fördern. Das Bundesministerium für Bildung und Forschung (BMBF) finanziert die D-Grid Initiative. Mit den Projekten soll eine nachhaltige Entwicklung der Grid-Technologie in Deutschland gefördert werden und damit die Forschung in Deutschland vorangetrieben werden. Von der D-Grid Initiative werden die Grid-Middleware Globus Toolkit 4.0, Unicore und gLite unterstützt. Für die Authentifizierung der Benutzer werden X.509-Zertifikat und das Shibboleth-Framework eingesetzt. Die Autorisierung wird mit dem Virtual Organization Membership Service (VOMS) und den Attributen, die das Shibboleth-Framework bereitstellt, umgesetzt. Im IVOM-Projekt soll eine einheitliche Benutzer- und Rechteverwaltung für die unterstützten Grid-Middleware-Systeme geschaffen werden. Die Basis des VO-Managements soll das Shibboleth-Framework bilden. Damit soll eine Dezentralisierung der Verwaltung geschaffen und es wird auch der Datenschutz verbessert, da die Benutzerdaten von der Heimatorganisation verwaltet werden. (vgl. [DGrid-2008])

6.2 myVOCS

Das Projekt „my Virtual Organization Collaboration System“ (myVOCS) wird an der University of Alabama in Birmingham durchgeführt. Im Projekt myVOCS wird ein Framework entwickelt, dass eine Zugriffskontrolle für Webservices bietet, dass ohne eine zentrale Verwaltung der Benutzer auskommt. Als Basis dieser Entwicklung dient das Shibboleth-Framework. Alle Identity Provider werden in einer Föderation organisiert und alle Service Provider werden in einer zweiten Föderation organisiert. Zwischen den zwei Föderationen befindet sich ein SAML-IdP-Proxy. Der SAML-IdP-Proxy bildet eine Brücke zwischen beiden Föderationen. Die Authentifizierung und Autorisierung der Benutzer wird durch den SAML-IdP-Proxy organisiert. Dieser sorgt dafür das die Authentifizierungsinformationen des Benutzers an den richtigen IdP geleitet werden und die Autorisierungsinformationen vom IdP zum richtigen Service Provider gesendet werden. Durch myVOCS soll die Verwaltung von virtuellen Organisationen vereinfacht werden. (vgl. [Canto-2007a])

7 Schlussbetrachtung

7.1 Resümee

Im Rahmen dieser Arbeit wurde ein Konzept erarbeitet, in dem die Attribute eines Benutzers einem Autorisierungssystem einer Grid-Ressource bereitgestellt wurden. Die Grid-Ressource ist ein Framework, dass eine rollenbasierte Autorisierung verwendet. In diesem Autorisierungssystem sollen die Attribute des Benutzers verwendet werden.

Zu Beginn der Arbeit gab es eine ausgedehnte Analysephase. In dieser musste eine ausführliche Analyse der Authentifizierung und Autorisierung in Grid-Systemen durchgeführt werden. Damit die Anforderungen, die von diesem Konzept erfüllt werden müssen herausgearbeitet werden konnten. Das Shibboleth-Framework musste ebenfalls einer eingehender Analyse unterzogen werden, genauso wie die Erweiterungen, die es für das Shibboleth-Framework gibt. Es zeigte sich, dass nicht nur das Autorisierungssystem der Grid-Middleware und der Grid-Ressource zu erweitern ist, sondern auch die verwendete Client-Software angepasst werden muss. Eine große Vereinfachung der notwendigen Arbeiten stellten die Erweiterungen des GridShib-Projektes da. Diese Realisierten die Kommunikation zwischen dem Shibboleth-Framework und der Grid-Middleware Globus Toolkit. Die Erweiterungen des GridShib-Projektes sorgten dafür, das die Attribute des Benutzers im Sicherheitskontext des Benutzers gespeichert wurden. Aus diesem müssen die Attribute des Benutzers durch die Grid-Ressource abgerufen werden und in das eigene Autorisierungssystem eingebunden werden.

Dieses Konzept zeigt, auf welchem Weg die Attribute eines Benutzers vom Identity Provider zum Autorisierungssystem einer Grid-Ressource übermittelt werden. Es wird erläutert, wie die Attribute im rollenbasierten Autorisierungssystem zum Feststellen der Benutzerrolle eingebunden werden. Die Verwendung eines rollenbasierten Autorisierungssystems in einer Grid-Ressource, gibt den Besitzern der Grid-Ressourcen die Möglichkeit die Zugriffsrechte der Benutzer sehr feingranular festzulegen. Unter Zuhilfenahme der Attribute des Benutzers, wird es möglich bestehende Autorisierungssysteme von Grid-Ressourcen, einzubeziehen ohne das der Benutzer sich an diesem Autorisierungssystem manuell anmelden muss.

7.2 Ausblick

Der nächste logische Schritt ist die Umsetzung des Konzepts. Bei der Implementierung wird die Grid-Ressource die Reconfigurable Computing Environment (RCE) sein. In RCE wird das rollenbasierte Autorisierungssystem auf der Grundlage dieses Kon-

zeptes erweitert. In dem Projekt PartnerGrid wird diese Implementierung zum Einsatz kommen. Das PartnerGrid-Projekt hat zum Ziel eine Softwareplattform für eine unternehmensübergreifende Zusammenarbeit zuschaffen. Die Softwareplattform soll auf der Grid-Technologie basieren. Auf lange Sicht wird es notwendig sein das Konzept für die Grid-Middleware UNICORE und gLite zuerweitern. Sobald die Komponenten des GridShib-Projektes Shibboleth 2.0 und Globus Toolkit 4.2 unterstützen, ist zuuntersuchen ob das Konzept für die neuen Software Versionen angepasst werden muss.

A X.509v3 Zertifikat mit SAML Assertion

Dieses digitale Zertifikat ist ein X.509-Zertifikate in der Version 3 (X.509v3), wie es in einer Public Key Infrastruktur (PKI) eingesetzt wird. Im Erweiterungsfeld des X.509v3-Zertifikats ist eine SAML-Attribut-Assertion enthalten, von der Zeile 27 bis zur Zeile 62. Dieses Zertifikat wurde von www.OpenIdp.org erstellt, dort kann jeder nach einer kurzen Anmeldung sein solches Zertifikat erhalten, es wird nur eine E-Mailadresse und ein Name benötigt. Diese Daten werden nicht überprüft, deshalb sollten solche Zertifikate nur zu Testzwecken verwendet werden.

```
1  Version: 3 (0x2)
2  Serial Number: 320 (0x140)
3  Signature Algorithm: sha1WithRSAEncryption
4  Issuer: DC=edu, DC=uiuc, DC=ncsa, DC=computer, O=Certificate Authority,CN=
       GridShib CA
5   Validity
6     Not Before: Mar 6 10:59:27 2008 GMT
7     Not After : Mar 6 22:59:27 2008 GMT
8  Subject: DC=edu, DC=uiuc, DC=ncsa, DC=computer, O=Shibboleth User, OU=urn:
       mace:inqueue:shib13.openidp.org, CN=FrankK@openidp.org
9  Subject Public Key Info:
10   Public Key Algorithm: rsaEncryption
11   RSA Public Key: (1024 bit)
12     Modulus (1024 bit):
13       00:9e:58:5a:02:7f:9f:36:68:59:78:68:25:cd:38:
14       4a:54:a2:79:5d:0b:6c:2a:c2:b2:b3:c8:d7:83:dd:
15       cc:82:ed:db:99:01:b5:22:4f:e3:d8:df:f8:c2:e1:
16       96:4e:91:60:dc:0a:ee:1b:60:8c:07:44:f7:7d:7a:
17       3d:7c:0b:24:6d:a0:7c:a9:db:29:85:45:1e:c8:fe:
18       a3:d8:2e:ed:71:dd:1a:a1:02:56:1f:21:23:d6:c6:
19       cd:d8:db:d5:36:74:74:d8:fb:69:be:de:f1:bb:65:
20       21:c9:b8:07:a5:cc:42:a3:6d:13:2e:e1:70:3c:e5:
21       9c:16:1e:64:cb:4f:d3:89:7f
22     Exponent: 65537 (0x10001)
23 X509v3 extensions:
24    1.3.6.1.4.1.3536.1.1.1.12:...}
25   <Assertion
26   xmlns="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:saml="urn:oasis:names:tc:
       SAML:1.0:assertion"
27   xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol" xmlns:xsd="http://www.w3.
       org/2001/XMLSchema"
28   xmlns:xsi="http :// www.w3.org/2001/XMLSchema-instance" AssertionID="
       _450cba2d3da1a242f2db519a73ad3749"
29   IssueInstant="2008-03-06T10:59:26.061Z" Issuer="https://test-sp.ncsa.uiuc.
       edu/shibboleth" MajorVersion="1" MinorVersion="1">
30     <Advice>
31       <Assertion AssertionID="_7f13b2d94a1b311c3862b9c826608a14"
```

```
            IssueInstant="2008-03-06T10:58:39.255Z" Issuer="urn:mace:inqueue:
            shib13.openidp.org" MajorVersion="1" MinorVersion="1">
32          <Conditions NotBefore="2008-03-06T10:58:39.255Z" NotOnOrAfter
              ="2008-03-06T18:58:39.255Z">
33            <AudienceRestrictionCondition>
34              <Audience>https://test-sp.ncsa.uiuc.edu/shibboleth</Audience>
35              <Audience>urn:mace:inqueue</Audience>
36            </AudienceRestrictionCondition>
37          </Conditions>
38          <AttributeStatement>
39            <Subject>
40              <NameIdentifier Format="urn:mace:shibboleth:1.0:nameIdentifier"
                    NameQualifier="urn:mace:inqueue:shib13.openidp.org">
41                _29021ff1f183e31e438b06739c7c36d7
42              </NameIdentifier>
43            </Subject>
44            <Attribute xmlns:typens="urn:mace:shibboleth:1.0" AttributeName="urn:
                  mace:dir:attribute-def:eduPersonPrincipalName" AttributeNamespace
                  ="urn:mace:shibboleth:1.0:attributeNamespace:uri">
45              <AttributeValue xsi:type="typens:AttributeValueType">FrankK</
                    AttributeValue>
46            </Attribute>
47          </AttributeStatement>
48        </Assertion>
49      </Advice>
50        <SubjectStatement xmlns:samlsap="urn:oasis:names:tc:SAML:1.1:profiles:
              assertion:subject" xsi:type="samlsap:SubjectStatementType">
51          <Subject>
52            <NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:
                  X509SubjectName">
53              CN=FrankK@openidp.org,OU=urn:mace:inqueue:shib13.openidp.org,O
                    =Shibboleth User,DC=computer,DC=ncsa,DC=uiuc,DC=edu
54            </NameIdentifier>
55            <SubjectConfirmation>
56              <ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-
                    vouches</ConfirmationMethod>
57            </SubjectConfirmation>
58          </Subject>
59        </SubjectStatement>
60    </Assertion>
61  Signature Algorithm: sha1WithRSAEncryption
62    4b:81:2f:29:92:65:fe:42:56:43:28:46:72:97:60:33:ba:ca:
63    97:20:c4:61:b2:5a:c7:3a:f1:9a:78:00:c3:9f:3e:d0:b6:1f:
64    1c:0b:78:50:b0:71:0f:41:7e:5e:4b:d8:27:ff:1b:e8:a4:82:
65    5c:80:16:e7:74:a2:79:51:02:70:77:75:aa:f9:ef:6f:9a:70:
66    f1:c1:cb:f8:38:52:d8:88:59:30:0b:21:d5:6d:b0:4b:14:9a:
67    bd:44:40:25:07:ef:83:db:18:a3:8c:ff:7a:fa:e5:0c:e7:9d:
68    c2:cf:6a:b8:b7:0c:71:f7:ea:25:ee:9d:69:32:38:a0:8e:45:
```

```
69   13:32:b9:86:1e:91:dd:34:07:49:e8:c5:bc:8c:60:41:c7:cc:
70   b1:f5:f1:5b:f4:6d:96:7b:10:ce:4b:cd:86:56:f2:04:5a:f4:
71   4a:d4:98:86:fa:11:91:1b:99:36:9b:6c:31:fe:63:8f:42:3f:
72   89:3a:96:42:8d:8c:5c:f2:0d:fe:6b:54:da:52:54:a7:88:0d:
73   e0:64:96:a3:c9:77:9a:73:f1:81:f0:91:83:37:dd:de:02:33:
74   ca:59:a6:3c:69:c4:30:9f:55:fc:d4:81:51:94:01:43:20:69:
75   3e:72:d6:3b:fb:50:0d:c8:88:9c:26:ef:e3:92:e9:11:df:ae:
76   cd:5f:3a:ce
```

Textansicht eines X.509-Zertifikats in der Version 3 (X.509v3) mit SAML Assertions.

B SAML-Nachrichten

In diesem Anhang sind die wichtigsten SAML-Nachrichten, die vom Shibboleth-Framework verwendet werden dargestellt. Dazu gehören das AuthenticationAssertions, die AttributeQuery und der AttributeResponse.

Im AuthenticationAssertion ist eine AuthenticationStatement enthalten, dieses gibt Auskunft über eine erfolgreiche Authentifizierung. In welchem Zeitraum das AuthenticationAssertion gültig ist (Zeile 8, 9) und wer es ausgestellt hat (Zeile 6) steht in dem AuthenticationAssertion. Auf welchem Ziel-System das AuthenticationAssertion gültig ist (Zeile 10 - 12) wird auch festgelegt. Im AuthenticationStatement ist enthalten welcher Benutzer (Zeile 21) und von welchem Identity Provider (Zeile 20) dieser authentifiziert wurde. Es ist auch enthalten, auf welche Weise der Benutzer sich authentifiziert hat (Zeile 16).

```
1  <saml:Assertion
2    xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
3    MajorVersion="1" MinorVersion="1"
4    AssertionID="a75adf55-01d7-40cc-929f-dbd8372ebdfc"
5    IssueInstant="2004-12-05T09:22:02Z"
6    Issuer="https://idp.example.org/shibboleth">
7    <saml:Conditions
8      NotBefore="2004-12-05T09:17:02Z"
9      NotOnOrAfter="2004-12-05T09:27:02Z">
10     <saml:AudienceRestrictionCondition>
11       <saml:Audience>http://sp.example.org/shibboleth</saml:Audience>
12     </saml:AudienceRestrictionCondition>
13   </saml:Conditions>
14   <saml:AuthenticationStatement
15     AuthenticationInstant="2004-12-05T09:22:00Z"
16     AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
17     <saml:Subject>
18       <saml:NameIdentifier
```

```
19          Format="urn:mace:shibboleth:1.0:nameIdentifier"
20          NameQualifier="https://idp.example.org/shibboleth">
21          3f7b3dcf-1674-4ecd-92c8-1544f346baf8
22        </saml:NameIdentifier>
23        <saml:SubjectConfirmation>
24          <saml:ConfirmationMethod>
25            urn:oasis:names:tc:SAML:1.0:cm:bearer
26          </saml:ConfirmationMethod>
27        </saml:SubjectConfirmation>
28      </saml:Subject>
29    </saml:AuthenticationStatement>
30  </saml:Assertion>
```

Aufbau eines AuthenticationAssertions. (vgl. [Scavo-2005] Seite 7, 8)

Die AttributeQuery wird in einer SOAP-Nachricht verpackt und mit dem Hypertext Transfer Protocol übermittelt. Mit einer AttributeQuery werden die Attribute eines Benutzer von einer Attribute Authority angefordert. In einer AttributeQuery steht von welchem Benutzer (Zeile 24) die Attribute sein sollen und welche Attribute (Zeile 27 - 30) benötigt werden. Es ist auch festgehalten, wer die Attribute anfordert (Zeile 19).

```
1  POST /shibboleth/AA/SOAP HTTP/1.1
2  Host: idp.example.org
3  Content-Type: text/xml
4  Content-Length: nnn
5  SOAPAction: http://www.oasis-open.org/committees/security
6
7  <?xml version="1.1" encoding="ISO-8859-1"?>
8  <SOAP-ENV:Envelope
9    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
10   <SOAP-ENV:Header/>
11   <SOAP-ENV:Body>
12   <samlp:Request
13     xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
14     xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
15     MajorVersion="1" MinorVersion="1"
16     IssueInstant="2004-12-05T09:22:04Z"
17     RequestID="aaf23196-1773-2113-474a-fe114412ab72">
18     <samlp:AttributeQuery
19       Resource="https://sp.example.org/shibboleth">
20       <saml:Subject>
21         <saml:NameIdentifier
22           Format="urn:mace:shibboleth:1.0:nameIdentifier"
23           NameQualifier="https://idp.example.org/shibboleth">
24           3f7b3dcf-1674-4ecd-92c8-1544f346baf8
25         </saml:NameIdentifier>
26       </saml:Subject>
27       <saml:AttributeDesignator
28         AttributeName="urn:mace:dir:attribute-def:eduPersonPrincipalName"
```

```
29          AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri"/>
30        </samlp:AttributeQuery>
31      </samlp:Request>
32    </SOAP-ENV:Body>
33  </SOAP-ENV:Envelope>
```

Aufbau einer AttributeQuery. (vgl. [Scavo-2005] Seite 22)

Der AttributeResponse wird als SOAP-Nachricht mit dem Hypertext Transfer Protocol versendet. Mit einem AttributeResponse werden die Attribute von einer Attribute Authority an einen Attribute Requester übermittelt. Ein AttributeAssertion beinhaltet der AttributeResponse. Die AttributeAssertion enthält ein AttributeStatement, dass die Attribute (Zeile 40 - 48) des Benutzers enthält. Dieses enthält auch den Benutzer (Zeile 37), zu dem die Attribute gehören. In welchem Zeitraum das AttributeAssertion gültig ist (Zeile 26, 27) und wer es ausgestellt hat (Zeile 24) ist in dem AttributeAssertion enthalten. Für welches Ziel-System die Attribute bestimmt sind (Zeile 28, 30) ist auch enthalten.

```
1  HTTP/1.1 200 OK
2  Content-Type: text/xml
3  Content-Length: nnnn
4
5  <?xml version="1.1" encoding="ISO-8859-1"?>
6  <SOAP-ENV:Envelope
7    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
8    <SOAP-ENV:Header/>
9    <SOAP-ENV:Body>
10     <samlp:Response
11       xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
12       InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
13       IssueInstant="2004-12-05T09:22:05Z"
14       MajorVersion="1" MinorVersion="1"
15       ResponseID="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
16       <samlp:Status>
17         <samlp:StatusCode Value="samlp:Success"/>
18       </samlp:Status>
19       <saml:Assertion
20         xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
21         MajorVersion="1" MinorVersion="1"
22         AssertionID="a144e8f3-adad-594a-9649-924517abe933"
23         IssueInstant="2004-12-05T09:22:05Z"
24         Issuer="https://idp.example.org/shibboleth">
25         <saml:Conditions
26           NotBefore="2004-12-05T09:17:05Z"
27           NotOnOrAfter="2004-12-05T09:52:05Z">
28           <saml:AudienceRestrictionCondition>
29             <saml:Audience>http://sp.example.org/shibboleth</saml:Audience>
30           </saml:AudienceRestrictionCondition>
```

```
31          </saml:Conditions>
32          <saml:AttributeStatement>
33            <saml:Subject>
34              <saml:NameIdentifier
35                Format="urn:mace:shibboleth:1.0:nameIdentifier"
36                NameQualifier="https://idp.example.org/shibboleth">
37                3f7b3dcf-1674-4ecd-92c8-1544f346baf8
38              </saml:NameIdentifier>
39            </saml:Subject>
40            <saml:Attribute
41              AttributeName="urn:mace:dir:attribute-def:eduPersonPrincipalName"
42              AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri"
                  >
43              <saml:AttributeValue Scope="example.org">
44                userid
45              </saml:AttributeValue>
46            </saml:Attribute>
47          </saml:AttributeStatement>
48        </saml:Assertion>
49      </samlp:Response>
50    </SOAP-ENV:Body>
51  </SOAP-ENV:Envelope>
```

Aufbau eines AttributeResponse. (vgl. [Scavo-2005] Seite 9, 22, 23)

C Shibboleth Metadaten-Datei

In der Shibboleth Metadatendatei werden alle Komponenten einer Authentifizierungs- und Autorisierungsinfrastruktur (AAI) beschrieben. Welche Aufgaben die Komponente hat und welches X.509-Zertifikat dieser Komponente zugeordnet ist, wird festgehalten. Zur Beschreibung werden die SAML-Metadaten in der Version 2.0 verwendet. Von der Zeile 10 bis 49 ist eine Identity Provider und von der Zeile 51 bis 67 ist eine Service Provider beschrieben. Ein Grid-Service Provider ist von der Zeile 69 bis Zeile 94 beschrieben.

```
1  <EntitiesDescriptor
2    xmlns="urn:oasis:names:tc:SAML:2.0:metadata"
3    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
5    xmlns:shibmd="urn:mace:shibboleth:metadata:1.0"
6    xsi:schemaLocation="urn:oasis:names:tc:SAML:2.0:metadata␣../schemas/saml-
         schema-metadata-2.0.xsd␣urn:mace:shibboleth:metadata:1.0␣../schemas/
         shibboleth-metadata-1.0.xsd␣http://www.w3.org/2000/09/xmldsig#␣../
         schemas/xmldsig-core-schema.xsd"
```

```
7  Name="urn:mace:shibboleth:examples"
8   validUntil ="2010-01-01T00:00:00Z">
9
10 <EntityDescriptor  entityID="urn:mace:shibboleth.local:test-IDP:metadata">
11   <IDPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:1
         .1:protocol urn:mace:shibboleth:1.0">
12     <Extensions>
13       <shibmd:Scope>shibboleth</shibmd:Scope>
14     </Extensions>
15
16     <KeyDescriptor use="signing">
17       <ds:KeyInfo>
18         <ds:X509Data>
19           <ds:X509Certificate>
20             copy of the  certificate
21           </ds:X509Certificate>
22         </ds:X509Data>
23       </ds:KeyInfo>
24     </KeyDescriptor>
25
26     <ArtifactResolutionService index="1" Binding="urn:oasis:names:tc:SAML:1.0
           :bindings:SOAP-binding" Location="https://shibboleth/shibboleth-idp/
           Artifact"/>
27     <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
28     <SingleSignOnService Binding="urn:mace:shibboleth:1.0
           :profiles:AuthnRequest" Location="https://shibboleth/shibboleth-idp/SSO"/
           >
29   </IDPSSODescriptor>
30
31   <AttributeAuthorityDescriptor  protocolSupportEnumeration="
         urn:oasis:names:tc:SAML:1.1:protocol">
32     <Extensions>
33       <shibmd:Scope>shibboleth</shibmd:Scope>
34     </Extensions>
35
36     <KeyDescriptor use="signing">
37       <ds:KeyInfo>
38         <ds:X509Data>
39           <ds:X509Certificate>
40             copy of the  certificate
41           </ds:X509Certificate>
42         </ds:X509Data>
43       </ds:KeyInfo>
44     </KeyDescriptor>
45
46     <AttributeService Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-
           binding" Location="https://shibboleth/shibboleth-idp/AA"/>
47     <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
```

```
48      </ AttributeAuthorityDescriptor >
49    </EntityDescriptor>
50
51    <EntityDescriptor  entityID="urn:mace:shibboleth.local:test-SP:metadata">
52      <SPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:1
            .1:protocol">
53        <KeyDescriptor use="signing">
54          <ds:KeyInfo>
55            <ds:X509Data>
56              <ds:X509Certificate>
57                copy of the  certificate
58              </ds:X509Certificate>
59            </ds:X509Data>
60          </ds:KeyInfo>
61        </KeyDescriptor>
62
63        <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
64        <AssertionConsumerService index="1" isDefault="true" Binding="
              urn:oasis:names:tc:SAML:1.0:profiles:browser-post" Location="https://
              shibboleth/Shibboleth.sso/SAML/POST"/>
65        <AssertionConsumerService index="2" Binding="urn:oasis:names:tc:SAML:1
              .0:profiles:artifact-01" Location="https://shibboleth/Shibboleth.sso/SAML/
              Artifact"/>
66      </SPSSODescriptor>
67    </EntityDescriptor>
68
69    <EntityDescriptor  entityID="https://shibboleth/gridshib">
70      <md:RoleDescriptor
71        xmlns:query="urn:oasis:names:tc:SAML:metadata:ext:query"
72        xsi:type="query:AttributeQueryDescriptorType"
73        protocolSupportEnumeration="urn:oasis:names:tc:SAML:1.1:protocol␣
              urn:oasis:names:tc:SAML:2.0:protocol">
74
75        <KeyDescriptor use="signing">
76          <ds:KeyInfo>
77            <ds:X509Data>
78              <ds:X509Certificate>
79                copy of the  certificate
80              </ds:X509Certificate>
81            </ds:X509Data>
82          </ds:KeyInfo>
83        </KeyDescriptor>
84
85        <md:NameIDFormat>
86          urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName
87        </md:NameIDFormat>
88        <md:AttributeConsumingService isDefault="true" index="0">
89        <md:ServiceName xml:lang="en">
```

```
90            Shibbolized Grid Service
91          </md:ServiceName>
92        </md:RoleDescriptor>
93      </EntityDescriptor>
94    </EntitiesDescriptor>
```

Die Metadaten-Datei einer Authentifizierungs- und Autorisierungsinfrastruktur (AAI) des Shibboleth-Frameworks. (vgl. [Scavo-2005] Seite 26 - 30)

Abkürzungsverzeichnis

AA	**A**ttribute **A**uthority
AAI	**A**uthentication and **A**uthorization **I**nfrastructure
AR	**A**ttribute **R**equester
CA	**C**ertificate **A**uthority
CAS	**C**ommunity **A**uthorization **S**erver
CPS	**C**ertification **P**ractice **S**tatement
CRL	**C**ertificate **R**evocation **L**ist
DAC	**D**iscretionary **A**ccess **C**ontrol
DN	**D**istinguished **N**ame
EEC	**E**nd **E**ntity **C**ertificates
GSI	**G**rid **S**ecurity **I**nfrastructure
GT4	**G**lobus **T**oolkit **4**.x
HTTP	**H**yper**t**ext **T**ransfer **P**rotocol
IdP	**Id**entity **P**rovider
JDBC	**J**ava **D**ata**b**ase **C**onnectivity
LDAP	**L**ightweight **D**irectory **A**ccess **P**rotocol
MAC	**M**andatory **A**ccess **C**ontrol
OASIS	**O**rganization for the **A**dvancement of **S**tructured **I**nformation **S**tandards
OCSP	**O**nline **C**ertificate **S**tatus **P**rotocol
OGSA	**O**pen **G**rid **S**ervice **A**rchitecture
OTP	**O**ne-**T**ime **P**assword
PDP	**P**olicy **D**ecision **P**oint
PIP	**P**olicy **I**nformation **P**oint
PKI	**P**ublic **K**ey **I**nfrastruktur
QoS	**Q**uality **o**f **S**ervice
RA	**R**egistration **A**uthority
RBAC	**R**ole**B**ased **A**ccess **C**ontrol
RCE	**R**econfigurable **C**omputing **E**nvironment
SAML	**S**ecurity **A**ssertion **M**arkup **L**anguage
SLC	**S**hort **L**ived **C**ertificates
SOAP	**S**imple **O**bject **A**ccess **P**rotocol
SP	**S**ervice **P**rovider
SSL	**S**ecure **S**ocket **L**ayer
SSO	**S**ingle **S**ign**O**n
TLS	**T**ransport **L**ayer **S**ecurity
URL	**U**niform **R**esource **L**ocator

VO	**V**irtual **O**rganization
VOMS	**V**irtual **O**rganization **M**embership **S**ervice
WAYF	**W**here **A**re **Y**ou **F**rom?
WSRF	**W**eb **S**ervice **R**esource **F**ramework
XACML	e**X**tensible **A**ccess **C**ontrol **M**arkup **L**anguage
XML	**E**xtensible **M**arkup **L**anguage

Literatur

[Barto-2006] Barton, Tom; et al.: *Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, GridShib, and MyProxy*. http://grid.ncsa.uiuc.edu/papers/gridshib-pki06-final.pdf, April 2006, April 2008.

[Canto-2007] Cantor, Scott: *User Authentication and Subject Identifiers in Shibboleth*. https://spaces.internet2.edu/display/SHIB/IdPUserAuthnConfig, September 2007, April 2008.

[Canto-2007a] Cantor, Scott: *Introduction to myVocs*. https://spaces.internet2.edu/display/GS/MyVocs, September 2007, Mai 2008.

[DataG-2003] DataGrid: *DataGRID : SECURITY DESIGN*. https://edms.cern.ch/file/414762/2.2/DataGrid-07-D7.7-0207-Security-2.2.pdf, März 2003, April 2008.

[DGrid-2008] D-Grid Initiative: *Interoperabilität und Integration der VO-Management Technologien im D-Grid*. http://www.d-grid.de/index.php?id=314&L=0, Mai 2008.

[Ecker-2008] Eckert, Claudia: *IT-Sicherheit : Konzepte - Verfahren - Protokolle*. Oldenbourg Wissenschaftsverlag GmbH, München, 2008.

[Foste-1998a] Forster, Ian; Kesselmann, Carl: *The GRID : Blueprint for a New Computing Infrastructure*. Morgen Kaufmann Publisher, San Francisco USA, 1998.

[Foste-2001] Forster, Ian; et al.: *The Anatomy of the Grid : Enabling Scalable Virtual Organizations*. In: *International Journal of Supercomputer Applications* 15 2001.

[Foste-2004] Forster, Ian; Kesselmann, Carl: *The GRID 2 : Blueprint for a New Computing Infrastructure*. 2.Auflage, Morgen Kaufmann Publisher, San Francisco USA, 2004.

[Foste-2006] Forster, Ian; et al.: *The Open Grid Service Architecture, Version 1.5*. http://www.ogf.org/documents/GFD.80.pdf, Juli 2006, Mai 2008.

[gLite-2008] Enabling Grids for E-sciencE (EGEE): *gLite documentation*. http://glite.web.cern.ch/glite/documentation/default.asp, Februar 2008.

[GlobA-2008a] Globus Alliance: *Globus Toolkit 4.0 Release Manuals*. http://www.globus.org/toolkit/docs/4.0/, Februar 2008.

[GlobA-2008b] Globus Alliance: *GridShib SAML Tools*. http://gridshib.globus.org/docs/gridshib-saml-tools-0.3.2/readme.html, März 2008, April 2008.

[GlobA-2008c] Globus Alliance: *GridShib-CA Documentation*. http://gridshib.globus.org/docs/gridshib-ca-0.5.1/index.html, April 2008.

[GlobA-2008d] Globus Alliance: *GT4 Delegation Service Developer's Guide*. http://www.globus.org/toolkit/docs/4.0/security/delegation/developer-index.html#s-delegation-developer-archdes, April 2008.

[GlobA-2008e] Globus Alliance: *GT 4.0 Security : Key Concepts*. http://www.globus.org/toolkit/docs/4.0/security/key-index.html, April 2008.

[GlobA-2008f] Globus Alliance: *GT 4.0 WS GRAM Approach*. http://www.globus.org/toolkit/docs/4.0/execution/wsgram/WS_GRAM_Approach.html#id2530452, April 2008.

[GlobA-2008g] Globus Alliance: *The Java Commodity Grid Kit (V1.2)*. http://www.cogkit.org/release/4_1_4/api/jglobus/index.html, Mai 2008.

[Housl-2002] Housley, R.; et al.: *Internet X.509 Public Key Infrastructure : Certificate and Certificate Revocation List (CRL) Profile*. http://www.rfc-editor.org/rfc/rfc3280.txt, April 2002, April 2008.

[Inet2-2008] Internet2 Middleware Initiative: *About*. http://shibboleth.internet2.edu/about.html, Mai 2008.

[Jacob-2005] Jacob, Bart; et al.: *Introduction to Grid Computing*. IBM Corporation, Armonk USA, Dezember 2005.

[LangB-2006] Lang, Bo; et al.: *A Multipolicy Authorization Framework for Grid Security*. In: Proceedings of the Fifth IEEE Symposium on Network Computing and Application, Institute of Electrical and Electronics Engineers, Cambridge USA, Juli 2006, April 2008.

[Nagar-2002] Nagaratnam, Nataraj; et al.: *The Security Architecture for Open Grid Services*. http://www.cs.virginia.edu/~humphrey/ogsa-sec-wg/OGSA-SecArch-v1-07192002.pdf, Juli 2002, Februar 2008.

[OASIS-2007a] Organization for the Advancement of Structured Information Standards (OASIS): *Metadata Profile for the OASIS Security Assertion*

Markup Language (SAML) V1.x. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml1x-metadata-os.pdf, November 2007, April 2008.

[OASIS-2007b] Organization for the Advancement of Structured Information Standards (OASIS): *Metadata Extension for SAML V2.0 and V1.x Query Requesters*. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-metadata-ext-query-os.pdf, November 2007, April 2008.

[OASIS-2007c] Organization for the Advancement of Structured Information Standards (OASIS): *Subject-based Profiles for SAML V1.1 Assertions*. http://www.oasis-open.org/committees/download.php/26572/sstc-saml1-profiles-assertion-subject-draft-01.pdf, Dezember 2007, April 2008.

[Pearl-2003] Pearlman, L.; et al.: *The Community Authorization Service : Status and Future*. http://http://www.globus.org/alliance/publications/papers/CAS_update_CHEP_03-final.pdf, März 2003, März 2008.

[Scavo-2005] Scavo, Tom; Cantor, Scott: *Shibboleth Architecture : Technical Overview*. http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-tech-overview-latest.pdf, June 2005, März 2008.

[Tueck-2004] Tuecke, S.; et al.: *Internet X.509 Public Key Infrastructure (PKI) : Proxy Certificate Profile*. http://www.rfc-editor.org/rfc/rfc3820.txt, Juni 2004, April 2008.

[Unico-2008] UNICORE Forum e.V.: *Unicore : Documentation*. http://www.unicore.eu/documentation/, Februar 2008.

[Welch-2005] Welch, V.: *Globus Toolkit Version 4 Grid Security Infrastructure : A Standards Perspective*. http://www.globus.org/toolkit/docs/4.0/security/GT4-GSI-Overview.pdf, September 2005, April 2008.