D-Grid Integrationsprojekt (DGI)

Fachgebiet 3-4 – Aufbau einer AA-Infrastruktur
für das D-Grid

Analyse von AA-Infrastrukturen in Grid-Middleware

Koordination
Christian Grimm (grimm@rvs.uni-hannover.de)
Marcus Pattloch (pattloch@dfn.de)

Version 1.1, 20. März 2006

D-Grid Integrationsprojekt (DGI)

Autoren

Tobias Dussa (SSCK, RZ, Universität Karlsruhe)
Ursula Epting (Forschungszentrum Karlsruhe)
Bartol Filipovic (Fraunhofer-Institut SIT Darmstadt)
Gerti Foest (DFN-Verein)
Jürgen Glowka (Forschungszentrum Karlsruhe)
Joachim Götze (ICSY, TU Kaiserslautern)
Christian Grimm (RRZN, Universität Hannover)
Markus Hillenbrand (ICSY, TU Kaiserslautern)
Christian Kohlschütter (Forschungszentrum L3S, Universität Hannover)
Rudolf Lohner (SSCK, RZ, Universität Karlsruhe)
Siegfried Makedanz (RZ, Alfred-Wegener-Institut, C3-Grid)
Paul Müller (RHRK/ICSY, TU Kaiserslautern)
Marcus Pattloch (DFN-Verein)
Stefan Piger (RRZN, Universität Hannover)
Tobias Straub (Fraunhofer-Institut SIT Darmstadt)
Jan Wiebelitz (RRZN, Universität Hannover)

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01AK800B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Inhaltsverzeichnis

1 Einleitung und Motivation

Im Rahmen internationaler Projekte der Großforschung, beispielsweise aus den Gebieten der Hochenergiephysik, der Astrophysik, der Klimaforschung und der Genomforschung, besteht der Bedarf an enormer Rechenleistung und Speicherkapazität, die einzelne Rechenzentren nicht bieten können. Um diesen Bedarf bedienen zu können wurde das Konzept des verteilten Rechnens zum Konzept des Grid Computing erweitert.

Grid Computing ermöglicht dem Benutzer den transparenten Zugriff auf einen Pool von Ressourcen. Diesem Pool können dynamisch Ressourcen hinzugefügt und entzogen werden, ohne dass der Benutzer davon erfährt. Der Pool kann Ressourcen beinhalten, die selten genutzt werden, wie zum Beispiel spezielle Rechnerarchitekturen oder spezielle Messgeräte und -apparaturen.

Benutzer und Ressourcen bilden über die Grenzen von Unternehmen, Organisationen und Institutionen hinweg sogenannte Virtuelle Organisationen (VO), die ihnen eine Zusammenarbeit an gemeinsamen Projekten ermöglichen. Die verschiedenen Grid Middlewares ermöglichen den Benutzern im Rahmen des VO-Konzeptes den transparenten Zugriff auf Ressourcen.

Eine Grid Middleware bietet dem Benutzer dabei die notwendigen Dienste, um einen Job zu bearbeiten und diesem die benötigten Daten bereitzustellen. Dazu gehören die Prüfung der Identität des Benutzers (Authentifizierung) und dessen Berechtigungen (Autorisierung), die Abgabe von Jobs in das Grid (Job Submission) und die Speicherung und Verteilung von Daten (Storage). Weitere Dienste einer Grid Middleware sind die Überwachung von Ressourcen und Jobs (Monitoring) und die Speicherung von Informationen zur Ressourcennutzung und Rechnungsstellung (Accounting und Billing). Um die Verfügbarkeit der Dienste zu gewährleisten, bieten Grid Middlewares weiterhin Dienste für Lastverteilung und Service Discovery.

Mit diesem Bericht legt das Fachgebiet 3-4 des D-Grid Integrationsprojekts eine Analyse der AA-Infrastrukturen in den Grid Middlewares Globus Toolkit 2, Globus Toolkit 4, LCG, gLite und UNICORE vor.

Neben einer allgemeinen Beschreibung der zugrunde liegenden Verfahren ist es das wesentliche Ziel dieser Analyse, die übergreifend über verschiedene Middlewares gemeinsam nutzbaren Komponenten einer AA-Infrastruktur zu identifizieren. Die Ergebnisse dieser Analyse werden in dem nächsten Projektabschnitt für die Planung einer einheitlichen AA-Infrastruktur im D-Grid verwendet.

2 Einführung

Eine *Authentication and Authorization Infrastructure* (AAI) stellt einen unabdingbaren und komplexen Bestandteil jeder Grid-Infrastruktur dar. Eine AAI ist ein zentrales Framework, über das sich Grid-Ressourcen, Benutzer und virtuelle Organisationen gegenseitig in Abhängigkeit ihrer Policies verifizieren. Hierzu werden in der Regel verteilte Beschreibungen von Berechtigung (Attribute) und Identität (Zertifikate) genutzt. Dabei integriert eine AAI sowohl Zertifizierungs- als auch Verzeichnisdienste und stellt Protokolle für den Zugriff auf diese Dienste zur Verfügung.

Dieses Kapitel fasst zunächst allgemein die grundlegenden Begriffe und Verfahren für Authentifizierung und Autorisierung in verteilten Systemen zusammen. In den weiteren Kapiteln werden darauf aufbauend die spezifischen Merkmale verschiedener Grid Middlewares analysiert.

2.1 Authentifizierung

Authentifizierung ist das Beweisen einer Identität. Davon zu unterscheiden ist die Autorisierung, die gewöhnlich nach erfolgter Authentifizierung die Frage regelt, welche Berechtigungen einem konkreten Benutzer zugestanden werden. Diese beiden Konzepte werden – gerade in der Informationstechnologie – leicht verwechselt; eine genaue Abgrenzung ist mitunter nicht einfach, aber von großer Wichtigkeit, wenn komplexe Systeme betrachtet werden.

Im Allgemeinen werden drei Klassen von Authentifizierungsverfahren unterschieden:

- Authentifizierung durch eine bestimmte Eigenschaft,
- Authentifizierung durch Besitz bestimmter Hardware und
- Authentifizierung durch Kenntnis bestimmter Informationen.

Es existieren für alle drei Klassen von Verfahren zahlreiche Beispiele; bei manchen konkreten Verfahren handelt es sich auch um Kombinationen mehrerer dieser Ansätze.

Authentifizierung durch eine bestimmte Eigenschaft ist im Alltag die älteste Form der Authentifizierung, in der Informationstechnologie aber der jüngste Ansatz, der unter der Bezeichnung „Authentifizierung durch biometrische Daten“ bekannt ist. Grund hierfür ist die Schwierigkeit, persönliche Eigenschaften von Anwendern für Computer zuverlässig, aber auch sicher erfassbar zu machen. Ein Beispiel für diese Art der Authentifizierung ist das Scannen und Vergleichen von Fingerabdrücken oder der Iris. Diese Klasse von Verfahren ist darauf angewiesen, über einen sicheren, also im Sinne der Authentifizierung vertrauenswürdigen Datenpfad vom Gerät, mit dem das geforderte Merkmal erfasst wird, zum Rechnersystem, demgegenüber der Benutzer sich authentifizieren will, zu verfügen.

Die zweite Art der Authentifizierung, die Authentifizierung durch den Besitz geeigneter Hardware, ist im Computerumfeld auch mitunter zu finden. Beispiel hierfür ist ein sogenanntes Dongle, also ein Stück Hardware, das den Anwender zum Beispiel als Käufer eines bestimmten Softwareproduktes ausweist. Diese Art der Authentifizierung bezieht ihre Sicherheit aus der Tatsache, dass die Hardware schwierig zu fälschen ist. Sie wird heute häufig in Kombination mit der Authentifizierung durch Kenntnis bestimmter Informationen eingesetzt. Speziell in Anwendungen mit höheren Sicherheitsanforderungen ist diese Kombination – beispielsweise in Form eines PIN-Generators – häufig anzutreffen.

Die Authentifizierung durch Kenntnis bestimmter Informationen ist in der Informationstechnologie sehr häufig zu finden. Bei der klassischen Authentifizierungsmethode, der Authentifizierung durch Angabe des Benutzernamens und des dazugehörigen Kennworts findet die eigentliche Authentifizierung durch die Kenntnis des geheimen Passwortes statt. Dieser Klasse von Authentifizierungsverfahren gehören die meisten heute eingesetzten Methoden an.

Es gibt einige Probleme, die allen Methoden zur Authentifizierung prinzipiell gemein sind. Insbesondere ist es im Allgemeinen schwierig, eine zuverlässige und sichere Authentifizierung

gegenüber entfernten Partnern durchzuführen. Für das Einloggen auf einem entfernten Rechner sind kryptographische Verfahren notwendig, um einen angemessenen Grad von Sicherheit zu erreichen. Darüber hinaus wird eine sichere Authentifizierung umso schwieriger und aufwendiger, je mehr Partnern gegenüber ein Benutzer authentifiziert werden soll, denn die erforderlichen Daten müssen auf allen Systemen aktuell gehalten werden.

Bei der Authentifizierung spielen in mehrerer Hinsicht kryptographische Verfahren eine Rolle:

- Es ist zumindest wünschenswert, für ein gewisses Maß an Sicherheit sogar notwendig, dass die Daten, die auf den Rechnern zur Authentifizierung eines Benutzers gehalten werden, vor unbefugtem Zugriff geschützt werden. Wird beispielsweise ein Passwort ungeschützt im Klartext gespeichert, dann kann jeder, der Zugriff auf den fraglichen Rechner erlangt, dieses Passwort lesen und sich in Zukunft als der zugehörige Benutzer ausgeben, denn er kennt dessen geheime Authentifizierungsinformation. Daher werden Kennwörter im Allgemeinen entweder verschlüsselt, noch besser mit Hilfe einer geeigneten Einwegfunktion unkenntlich gemacht. Hier kommt also entweder ein Verschlüsselungs- oder ein Hash-Algorithmus zum Einsatz.
- Bei Authentifizierung über ungesicherte Netzwerke hinweg müssen die Authentifizierungsdaten vetraulich übertragen werden. Dies kann entweder durch Verschlüsselung des Datenstroms oder durch Verwenden von Einmalpasswörtern geschehen. Beide Ansätze erfordern kryptographische Funktionen; in einem Fall werden Verschlüsselungs-, im anderen Fall Algorithmen zur Generierung von Pseudozufallszahlen benötigt.

2.1.1 Grundlegende Verfahren

Im Folgenden werden die gängigsten grundlegenden Verfahren zur Authentifizierung umrissen.

2.1.1.1 Username/Password

Das Username/Password-Verfahren ist vermutlich das älteste noch angewandte Authentifizierungsverfahren in der Informationstechnologie. Hierbei wird auf dem System, dem gegenüber der Benutzer sich authentifizieren soll, eine Tabelle mit allen bekannten Benutzern sowie geheimen Kennwörtern gehalten. Die Kennwörter werden in der Regel verschlüsselt oder als Hash-Werte gespeichert, um zu verhindern, dass Unbefugte die Kennwörter erfahren, falls sie in den Besitz dieser Tabelle gelangen sollten. Der klassische Unix-Login verwendet diesen Ansatz. Als Abwandlung dieses Verfahrens kann jedem Benutzer eine ganze Passwortliste zugeordnet werden, von der jedes Kennwort genau einmal verwendet werden kann. Auf diese Weise wird verhindert, dass unbefugtes Mitlesen des Authentifizierungsvorgangs dazu führt, dass ein Angreifer sich als der belauschte Benutzer authentifizieren kann. Dieser Vorteil wird allerdings durch die Notwendigkeit zum regelmäßigen Verteilen ganzer Passwortlisten erkauft, was unter Umständen sehr schwierig oder gar unmöglich ist. Dieses Verfahren wird häufig mittels TAN-Listen beim Online-Banking benutzt.

2.1.1.2 Challenge/Response

Beim Challenge/Response-Verfahren handelt es sich um eine Verallgemeinerung des Username/Password-Verfahrens. Auch hier wird eine Tabelle mit Benutzern gehalten, die sich authentifizieren können; zusätzlich werden die geheimen Authentifizierungsdaten gespeichert. Der wesentliche Unterschied zum Username/Password-Verfahren liegt darin, dass hier die geheime Information nicht statisch ist, sondern dynamisch, sich also regelmäßig ändert. Eine verbreitete Möglichkeit, dies zu implementieren, ist eine nummerierte TAN-Liste; hierbei wird dem sich authentifizierenden Benutzer die Nummer einer bisher unbenutzten TAN vorgegeben, woraufhin die entsprechende TAN eingegeben werden muss. Eine andere Möglichkeit besteht im Einsatz spezieller Hardware. Es gibt beispielsweise Passwortgeneratoren von der Größe einer Kreditkarte, die abhängig von der aktuellen Uhrzeit eine Pseudozufallszahl anzeigen. Jedem Benutzer wird ein solches Gerät

ausgehändigt. Zum Authentifizieren gibt der Anwender die aktuell angezeigte Kennzahl als Passwort an; die Gegenstelle verfügt über einen weiteren, synchron laufenden Passwortgenerator und ist so in der Lage, die Authentifizierung zu prüfen. Dieses Konzept lässt sich darüber hinaus weiter absichern, indem beispielsweise eine geheime PIN, die jedem Benutzer individuell zugeordnet wird, mit der generierten Zahl verknüpft wird.

2.1.1.3 Public-Key-Verfahren

Public-Key-Verfahren basieren auf so genannten asymmetrischen Chiffren. Hierbei handelt es sich um Chiffren, bei denen es zwei korrespondierende Schlüssel gibt: den öffentlichen und den geheimen Schlüssel. Aus dem einen Schlüssel alleine kann der andere Schlüssel nach heutigem Kenntnisstand nicht effizient berechnet werden. Der öffentliche Schlüssel wird frei verteilt. Da er öffentlich ist, besteht keine Notwendigkeit, ihn vor Mitlesen Unbefugter zu schützen; lediglich eine unbefugte Änderung muss verhindert werden. Der zugehörige geheime Schlüssel darf jedoch nur dem Benutzer bekannt sein, den das Schlüsselpaar authentifizieren soll. Da mit dem einen Schlüssel chiffrierte Nachrichten nur mit dem anderen Schlüssel wieder dechiffriert werden können, genügt es zur Authentifizierung, wenn ein Anwender zeigt, dass er im Besitz des zu seinem öffentlichen Schlüssel passenden geheimen Schlüssels ist.

Public-Key-Verfahren ermöglichen die gegenseitige Authentifizierung der Kommunikationspartner. Einem Server kann ein eigenes Schlüsselpaar zugeordnet werden, mit dessen Hilfe sich der Server ebenfalls gegenüber dem Anwender authentifiziert.

2.1.2 Infrastrukturen zur Authentifizierung

Die bisher vorgestellten grundlegenden Verfahren zur Authentifizierung haben ein gemeinsames Problem, nämlich das der zuverlässigen und sicheren Verteilung der zur Authentifizierung benötigten Daten. Bei allen Verfahren muss garantiert werden, dass diese Daten nicht unbemerkt von Angreifern verändert werden können; bei Username/Passwort- und Challenge/Response-Verfahren müssen die Daten zusätzlich geheim gehalten werden. Es gibt zwei verbreitete Ansätze, dieses Problem zu lösen, indem eine Authentifizierungsinfrastruktur aufgebaut wird. Generell wird hierbei eine neue Instanz eingeführt, der alle beteiligten Parteien vertrauen müssen. Auf diese Weise kann eine Zentralisierung der eigentlichen Authentifizierungsdaten erfolgen, so dass weniger Austausch sensibler Daten notwendig ist.

2.1.2.1 Kerberos 5

Beim Kerberos-5-Protokoll meldet sich der Benutzer bei einem zentralen Authentifizierungsserver mit Hilfe regulärer Username/Passwort-Authentifizierung an. Durch diese Anmeldung erhält er vom Authentifizierungsserver ein Paket, ein so genanntes Ticket-Granting-Ticket (TGT), das unter anderem verschlüsselte Informationen über den Benutzer sowie Zeitstempel enthält. Dieses TGT ist zeitlich begrenzt gültig, um Missbrauch zu minimieren, falls das TGT in die falschen Hände geraten sollte. Das TGT versetzt den Anwender in die Lage, von einem weiteren Server Diensttickets zu erhalten. Die Diensttickets sind beschränkt auf den jeweiligen Dienst, für den sie ausgestellt wurden; zusätzlich sind auch sie zeitlich begrenzt.

In Kerberos 5 authentifiziert sich der Anwender durch sein Passwort gegenüber dem Authentifizierungsserver, alle weiteren Dienste haben jeweils einen gemeinsamen kryptographischen Schlüssel mit dem Authentifizierungsserver. Dieser zentrale Ansatz minimiert den Gesamtaufwand, was die Verwaltung von Benutzern betrifft. Zusätzlich sind alle benutzerspezifischen Authentifizierungsdaten auf ein einziges System konzentriert, das vergleichsweise einfach vor unbefugtem Zugriff zu schützen ist. Ein weiterer Vorteil des Kerberos-Protokolls ist es, dass implizit eine gegenseitige Authentifizierung durchgeführt wird, da ein Diensttickets nur von dem Server entschlüsselt werden kann, für den es bestimmt ist. Ein Angreifer kann also keinen Dienstserver simulieren, ohne den eigentlichen Server kompromittiert zu haben.

2.1.2.2 PKI-Konzepte

X.509 stellt eine Infrastruktur für die Authentifizierung auf der Basis der Public-Key-Kryptographie bereit. Mit diesem Verfahren werden die öffentlichen Schlüssel, die für jeden Benutzer existieren, vor unbefugter Manipulation geschützt. Um sicherzustellen, dass ein öffentlicher Schlüssel nicht verändert wurde, also tatsächlich dem Benutzer gehört, dem er zugeordnet ist, werden auch hier vertrauenswürdige Instanzen, die Certification Authorities (CAs, s. Kapitel 3), eingeführt. Die CAs verfügen selbst über ein Schlüsselpaar. Sie signieren den öffentlichen Schlüssel eines Benutzers mit ihrem geheimen Schlüssel. Das so erstellte Zertifikat des Benutzers kann dann verteilt werden. Ein Server kann davon ausgehen, dass der in dem Zertifikat enthaltene öffentliche Schlüssel tatsächlich dem Benutzer gehört, der im Zertifikat angegeben ist. Voraussetzung hierfür ist, dass der Server der ausstellenden CA vertraut und ihren öffentlichen Schlüssel kennt. Mit einem zertifizierten Schlüssel können weitere Zertifikate ausgestellt werden, wodurch eine Zertifizierungshierarchie entsteht. Die erstellten Zertifikate sind – ähnlich wie die Tickets beim Kerberos-Protokoll – zeitlich begrenzt ausgestellt. Zusätzlich existiert die Möglichkeit, ein Zertifikat ungültig zu machen, indem es auf eine definierte schwarze Liste von Zertifikaten, der so genannten Certificate Revocation List (CRL), aufgenommen wird. Jede Stelle, die Authentifizierung aufgrund von Zertifikaten implementiert, muss die zur Verfügung stehenden CRLs aktuell halten und dagegen prüfen.

Mit diesem Ansatz wird das Problem der sicheren Verteilung der öffentlichen Benutzerschlüssel auf die sichere Verteilung der öffentlichen Schlüssel der CAs, die die Benutzerschlüssel zertifizieren, reduziert. Die Anzahl der zu verteilenden Schlüssel ist also erheblich kleiner, das Problem damit praktisch leichter lösbar.

Der Einsatz von X.509 bietet noch einen weiteren Vorteil. Der geheime Schlüssel sollte in der Regel durch ein Passwort von unbefugtem Zugriff geschützt sein. Dies bedeutet, dass der Anwender jedes Mal, wenn er sich authentifizieren muss, seinen geheimen Schlüssel durch Eingabe des Passwortes freischalten muss. Durch die Möglichkeit, Schlüsselpaare zu signieren, ist aber auch ein so genannter Proxy-Mechanismus möglich. Hierbei wird zu Beginn einer Arbeitssitzung ein neues Schlüsselpaar generiert, das nicht passwortgeschützt ist. Anschließend wird der öffentliche Schlüssel mit dem normalen Schlüssel des Benutzers zertifiziert, wofür einmal der geheime Schlüssel des Benutzers durch Eingabe des Passwortes freigeschaltet werden muss. Dieses neu erstellte Proxy-Zertifikat ist zeitlich eng begrenzt; in der Regel hat es eine Lebensdauer von wenigen Stunden. Da es nur dann gültig ist, wenn das normale Schlüsselpaar des Anwenders ebenfalls gültig ist, kann es anstelle des normalen Zertifikats verwendet werden. Auf diese Weise wird vermieden, dass der Benutzer bei jeder Authentifizierung das Passwort seines geheimen Schlüssels eingeben muss. Da X.509 auf Public-Key-Verschlüsselung basiert, ist es auch hier sehr einfach, gegenseitige Authentifizierung zu implementieren, indem gefordert wird, dass jeder Server ebenfalls ein Schlüsselpaar besitzen und beim Authentifizierungsvorgang ein gültiges Zertifikat präsentieren muss.

2.1.3 Single Sign-On

Das Konzept des Single Sign-On bezeichnet den Umstand, dass der Benutzer die ihn authentifizierenden Informationen – zum Beispiel Fingerabdruck, Passwort, geheimer Schlüssel – bei jeder Arbeitssitzung nur ein einziges Mal zur Verfügung stellen muss. Im einfachsten Fall wird diese relevante Information von der benutzten Clientsoftware zwischengespeichert. Solange das Programm läuft, ist also keine weitere Passworteingabe notwendig. Insbesondere das Kerberos-Protokoll und X.509 verfügen über Mechanismen, die Single Sign-On unterstützen beziehungsweise direkt implementieren. Im Falle von Kerberos braucht ein Benutzer nur ein einziges Mal sein Kennwort einzugeben. Das Ticket-Granting Ticket, das er daraufhin erhält, erlaubt die weitere Authentifizierung gegenüber angeschlossenen Diensten ohne Kennwort, solange es gültig ist. Bei X.509 kann mit Hilfe von Zertifizierung ein so genanntes Proxy-Zertifikat erstellt werden, das nicht durch Kennwort geschützt, aber nur kurzzeitig gültig ist. Dieses Proxy-Zertifikat kann durch die Zertifizierung mit dem eigentlichen Benutzerzertifikat stellvertretend zur Authentifizierung genutzt werden.

2.2 Autorisierung

2.2.1 Was ist Zugriffskontrolle?

Das Ziel von Zugriffskontrolle bzw. Autorisierung ist es, Aktionen und Operationen von Benutzern zu beschränken, um zu verhindern, dass gegen Sicherheitsrichtlinien verstoßen wird, die in einer Organisation gelten. Dazu gehört auch, Programme einzuschränken, die von einem Benutzer ausgeführt werden oder einem Benutzer Rechte zu verschiedenen Bereichen eines Systems zu gewähren oder zu verwehren. Die Autorisierung setzt üblicherweise einen Authentifizierungsprozess voraus, da auf diese Weise die Identität des Benutzers festgestellt und bestätigt wurde.

Um die Zugriffe auf Objekte im System einzuschränken, werden alle Anfragen an einen Autorisierungsmonitor geschickt. Der Autorisierungsmonitor überprüft, ob die vom Benutzer gewünschte Aktion auf diesem Objekt erlaubt ist. Dazu stellt der Autorisierungsmonitor eine Anfrage an die Autorisierungsdatenbank, um die Zugriffsrechte des Benutzers für den Zugriff auf das Objekt zu beziehen. Daraufhin wird der Zugriff gewährt bzw. abgelehnt.

Der Bereich, den ein Autorisierungsmonitor oder eine Gruppe von zusammengehörigen Autorisierungsmonitoren überwacht, wird als (Autorisierungs-)Domäne (engl.: domain) bezeichnet. Wenn verschiedene unabhängige Autorisierungsmonitore den Autorisierungsaussagen aller anderen Autorisierungsmonitore dieser Gruppe vertrauen, dann gehören die auf diese Weise überwachten Objekte zu einer gemeinsamen Vertrauensdomäne (engl.: trust domain).

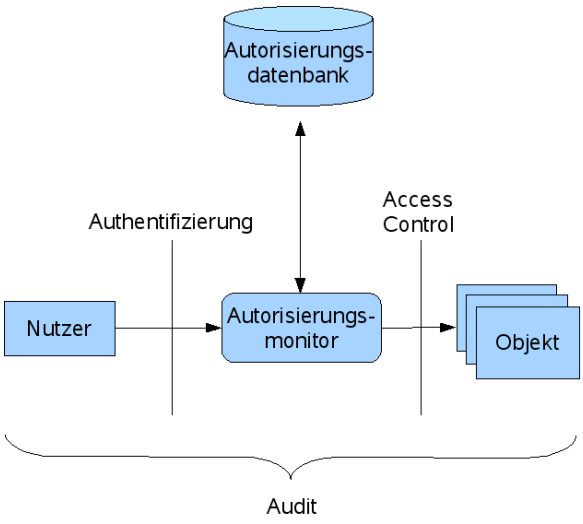

Abbildung 1: Zugriffskontrolle

Die Autorisierungsdatenbank wird von einem oder mehreren Sicherheitsadministratoren entsprechend der Sicherheitsrichtlinien der Organisation gepflegt. Es kann aber auch der Fall sein, dass Benutzer berechtigt sind, gewisse Bereiche der Datenbank selbstständig zu verändern, um beispielsweise Rechte an persönlichen Dateien festzulegen. Die Audit-Komponente überwacht alle Anfragen und Zugriffswünsche und protokolliert diese aus Gründen der Nachvollziehbarkeit mit. Dadurch ist eine nachträgliche Analyse der Zugriffe möglich und versuchte oder erfolgreiche Verstöße gegen die Sicherheitsrichtlinien können aufgespürt werden.

Um das Thema Zugriffskontrolle zu diskutieren, werden üblicherweise Objekte und Subjekte der Autorisierung voneinander unterschieden. Dabei sind Objekte alle Ressourcen, die von Computersystemen verwaltet werden und in denen Daten gespeichert sind. Aktivitäten im System werden von Subjekten ausgelöst, wobei es sich um Benutzer oder vom Benutzer ausgeführte Programme handelt. Ein Benutzer kann durch verschiedene Subjekte repräsentiert werden die unterschiedliche Rechte besitzen. Subjekte können wiederum selber Objekte sein. Zum Beispiel können Prozesse, die ein Benutzer in einer verteilten Umgebung startet, von diesem unterbrochen

oder beendet werden. Die Bedeutung von Zugriffsrechten kann je nach Art der Objekte unterschiedlich sein. Bei Dateien sind die Zugriffsrechte typischerweise Lesen, Schreiben, Ausführen und Besitzen, wobei Besitzen bedeutet, dass der Benutzer in der Lage ist, die Zugriffsrechte der Datei zu verändern.

Die Zugriffsmatrix ist ein Modell, das für jedes Subjekt und jedes Objekt die vorhandenen Zugriffsrechte festlegt. Jedes Subjekt hat eine Zeile und jedes Objekt eine Spalte in der Matrix. Jedes Element in der Matrix enthält für das entsprechende Subjekt die Zugriffsrechte auf das entsprechende Objekt. Die Aufgabe der Zugriffskontrolle ist es nun, nur solche Operationen zuzulassen, die in der Zugriffsmatrix erlaubt sind. Dazu verwendet man einen Autorisierungsmonitor, der jeden Zugriff auf das System überprüft. In einem realen System wird die Zugriffsmatrix sehr groß und die meisten Einträge sind üblicherweise leer. Deshalb ist die Zugriffsmatrix sehr selten wirklich als Matrix implementiert.

2.2.2 Zugriffskontrolle im Bezug auf andere Sicherheitsmaßnahmen

Autorisierung ist eine von mehreren Komponenten, die zur Umsetzung von Sicherheitsrichtlinien in Computersystemen dient. Dabei kann Zugriffskontrolle aber auf keinen Fall alleine stehen, da Zugriffskontrolle eine erfolgreiche und korrekte Identifikation voraussetzt, die von der Autorisierungskomponente gewährleistet werden muss.

Die Zugriffskontrolle kann nur den Zugriff von legitimierten Benutzern beschränken. Wenn sich ein Angreifer der Identität eines Benutzers bemächtigt, ist die Zugriffskontrolle nicht in der Lage, das System vor unbefugtem Zugriff zu schützen.

2.2.3 Autorisierungskonzepte

2.2.3.1 Access Control List (ACL)

Ein weit verbreiteter Ansatz zur Realisierung eines Systems zur Rechteverwaltung basiert auf einer Zugriffskontrollliste bzw. Access Control List, in der für jede Anwendung die autorisierten Benutzer, möglichst mit spezifischen Zugriffsrechten bzgl. einer Anwendung, aufgelistet sind.

Diese Listen können entweder lokal auf den jeweiligen Servern gespeichert sein oder sie werden zentral in einem eigens dafür geschaffenen Autorisierungsdienst verwaltet. Die zentrale Verwaltung der Zugriffsrechte ist aus Sicherheitsgründen und vom Aufwandsaspekt her gesehen generell einer dezentralen Lösung vorzuziehen.

Jedem Objekt ist eine ACL zugeordnet, die für jedes Subjekt die entsprechenden Zugriffsrechte enthält. Mit einem Blick auf eine ACL können alle berechtigten Subjekte festgestellt werden. Man kann auch einfach alle Zugriffe auf ein Objekt verbieten, in dem man die ACL durch eine leere ersetzt. Will man aber für ein Subjekt alle Zugriffsrechte ermitteln, ist es erforderlich, die ACLs für alle Objekte zu überprüfen. Wenn ein Benutzer aus einer Organisation ausscheidet, wird in der Regel einfach der Benutzer gelöscht. Wechselt der Benutzer aber nur das Aufgabengebiet, müssen alle ACLs angepasst werden.

2.2.3.2 Capability List

Capabilities basieren auf einem Konzept analog zu ACLs. Dabei wird jedem Subjekt eine Capability-Liste mit Objekten zugeordnet, auf die zugegriffen werden darf. Bei Capability Lists ist es einfach, alle Zugriffsrechte eines Subjekts zu bestimmen. Um aber alle Zugriffsrechte für ein Objekt zu ermitteln, muss man wieder alle Capability Lists durchgehen.

In den 70er Jahren wurden einige Systeme mit diesem Ansatz entwickelt, konnten sich aber nicht durchsetzten. Heutige Betriebssysteme setzen hauptsächlich auf ACL-basierte Ansätze.

2.2.4 Richtlinien zur Autorisierung

2.2.4.1 Discretionary Access Control (DAC)

Bei Discretionary Access Control wird anhand der Identität des Benutzers und der Berechtigungen, die für jeden Benutzer und jedes Objekt den Zugriffsmodus festlegen, entschieden, ob ein Zugriff gestattet wird oder nicht.

Die Flexibilität von DAC hat zu einer weiten Verbreitung in Systemen geführt, die von der Industrie eingesetzt werden. Ein Nachteil von DAC ist, dass es nicht möglich ist, den Informationsfluss wirksam zu regeln, z.B. kann ein Benutzer, der ein Objekt lesen kann, diese Informationen an jeden beliebigen Benutzer weitergeben.

Bei Richtlinien, die auf expliziten Erlaubnissen (positiven Autorisierungen) für den Zugriff beruhen, spricht man von geschlossenen Richtlinien. Die standardmäßige Entscheidung des Autorisierungsmonitors ist eine Verweigerung des Zugriffs.

Offene Richtlinien erlauben grundsätzlich den Zugriff und nur wenn ein explizites Verbot (negative Autorisierungen) für den Zugriff besteht, lehnt der Referenzmonitor den Zugriff ab. Es können auch positive und negative Autorisierungen gemeinsam eingesetzt werden, allerdings vergrößert diese Mischung aus Verboten und Berechtigungen die Komplexität des Systems und den Aufwands zur Administration.

2.2.4.2 Mandatory Access Control (MAC)

Bei Mandatory Access Control wird der Zugriff auf Subjekte und Objekte durch eine Klassifikation in Sicherheitsstufen geregelt. Die Sicherheitsstufe eines Objekts spiegelt die Empfindlichkeit der Informationen dar, zum Beispiel den potentiellen Schaden, wenn die Information in falsche Hände gerät.

Die Sicherheitsstufe, die ein Benutzer erhält, spiegelt die Vertrauenswürdigkeit des Benutzers wieder. Im einfachsten Fall ist die Sicherheitsstufe ein Element einer geordneten Menge. In Regierungskreisen in den USA besteht diese Menge normalerweise aus den Elementen „Top Secret“ (TS), „Secret“ (S), „Confidential“ (C) und „Unclassified“ (U) mit der Relation TS > S > C > U. Jede Sicherheitsstufe dominiert sich und alle niedrigeren in dieser Hierarchie.

Zugriff auf ein Objekt wird nur gewährt, wenn die Sicherheitsstufe eine bestimmte Relation zwischen den Sicherheitsstufen erfüllt:

- Read Down: Die Sicherheitsstufe eines Subjekts muss die des Objekts, das gelesen werden soll, dominieren.
- Write Up: Die Sicherheitsstufe eines Subjekts muss von der Sicherheitsstufe des zu schreibenden Objekts dominiert werden.

Auf diese Weise ist es nur möglich, dass Informationen in höhere oder gleiche Sicherheitsstufen weitergegeben werden. Durch diese Regeln entsteht der Nebeneffekt, dass jemand, der Sicherheitsstufe S besitzt, Objekte mit Sicherheitsstufe TS überschreiben kann. Deswegen wird üblicherweise der Schreibzugriff auf die gleiche Sicherheitsstufe beschränkt. Weiterhin kann ein Benutzer keine Objekte mit niedrigerer Klassifizierung bearbeiten. Um das zu erreichen muss sich der Benutzer als Subjekt mit entsprechend niedrigerer Sicherheitsstufe anmelden. Dazu kann sich jeder Benutzer mit allen Sicherheitsstufen anmelden, die von seiner Sicherheitsstufe dominiert werden.

Der Hauptgrund für die Existenz der Write-Up-Regel stellt in erster Linie die Absicherung gegen bösartige Software dar. Auf diese Weise wird verhindert, dass Software Informationen an niedrigere Stufen weitergibt. Der Benutzer genießt das Vertrauen und es wird davon ausgegangen, dass Benutzer keine Informationen weitergeben. Wenn man die Regeln umkehrt, bekommt man eine Zugriffskontroll-Richtlinie, die die Integrität von Daten sicherstellt.

2.2.4.3 Role-Based Access Control (RBAC)

Das Prinzip der rollenbasierten Zugriffskontrolle wurde 1992 erstmals von David Ferrariolo und Richard Kuhn veröffentlicht. Im Februar 2004 wurde RBAC als offizieller ANSI Standard anerkannt.

Role-Based Access Control wurde entwickelt, da sowohl das zu schwache DAC als auch das zu restriktive MAC ungeeignet sind, um den Anforderungen kommerzieller Anwendungen nachzukommen. Dabei wird die Tätigkeit eines Benutzers im System stärker berücksichtigt. Diese Entwicklung ist insbesondere für den Einsatz in großen Organisationen, mit einer großen Anzahl von Benutzern und einer Vielzahl von Anwendungen, von Vorteil.

Durch die maßgebliche Erhöhung der Flexibilität des Systems lässt sich die Zugriffskontrolle realitätsnah an die Begebenheiten anpassen. Mit Hilfe der zu identifizierenden Rollen wird die Abbildung der Organisationsstrukturen möglich. Die Verwaltung der Zugriffskontrolle vereinfacht sich durch die Zusammenfassung von Einzelrechten in Rollen, die den Benutzern zugeordnet werden, wobei je nach Konzept ein Benutzer mehrere Rollen besitzen kann, üblicherweise aber nur im Kontext einer Rolle arbeiten kann. Technisch setzt das rollenbasierte Konzept ein zentrales Zugriffssystem voraus, auf dem die Rollen und die Zuweisungen zu den Mitarbeitern verwaltet werden.

RBAC ist in Hinsicht auf Richtlinien und Organisationsstrukturen sehr flexibel. Eine seiner größten Stärken sind die Möglichkeiten, die es für die Administration bietet. Wenn die Zuordnung von Einzelrechten zu den jeweiligen Rollen in einem System einmal definiert ist, ändert sie sich relativ selten. Die Aufgaben des Administrators bestehen in erster Linie darin, die Mitgliedschaften in der Menge der spezifizierten Rollen zu verwalten. Wenn ein neuer Benutzer in die Organisation eintritt, bekommt er einfach eine bestehende Rolle zugewiesen. Ändert eine Person ihre Tätigkeit in der Organisation, bekommt sie einfach eine neue Rolle zugewiesen und die alten Mitgliedschaften werden aufgehoben. Wenn ein Benutzer die Organisation verlässt, werden alle Mitgliedschaften in Rollen gelöscht. Zusätzlich kann noch eine Rollen-Hierarchie definiert werden, die eine weitere Vereinfachung der Administration mit sich bringt.

2.2.4.4 Attribute-Based Access Control (ABAC)

Die Grundidee attributbasierter Zugriffskontrolle besteht darin, Zugriffsrechte zwischen den Subjekten und Objekten nicht statisch zu definieren, sondern ihre Eigenschaften oder Attribute dynamisch als Grundlage der Autorisierung zu nutzen. Die Attribute der Benutzer werden in diesem Zusammenhang auch Credentials genannt. Dabei kann es sich um allgemeine Eigenschaften wie beispielsweise die Position des Benutzers im Unternehmen handeln. Sofern erforderlich, insbesondere bei Zugriff durch Unternehmensexterne (z.B. Kunden), treten aber auch Attribute wie das Alter, die Lieferadresse, oder sogar erworbene Credentials (z.B. Abonnements) an diese Stelle. Attribute können (im Gegensatz zu relativ statisch definierten Rollen) sehr dynamisch sein. Beispielsweise könnte man sich in einem mobilen Umfeld die Verwendung des aktuellen Standorts eines Benutzers als Attribut vorstellen. Zur Kodierung der Benutzerattribute kann beispielsweise X.500 verwendet werden, ein Standard auf dem auch LDAP (Lightweight Directory Access Protocol) basiert, obwohl nicht alle Anforderungen umgesetzt wurden. Auf der Seite der Sicherheitsobjekte lassen sich die Inhalte der Dokumente durch Metadaten beschreiben, beispielsweise basierend auf dem Dublin Core Metadatenstandard. Auch diese Metadaten können als Attribute zur Zugriffskontrolle herangezogen werden.

In der Literatur wurden bereits einige attributbasierte Ansätze vorgeschlagen, z.B. das Digital Library Access Control Model (DLAM) für Sicherheit in digitalen Bibliotheken, welches Zugriffsrechte anhand von Benutzerattributen und mit Objekten assoziierten Konzepten definiert. Andere Arbeiten haben ihre Ursprünge im Bereich der Public Key und Privilege Management Infrastructures und basieren auf X.509-Attribut-Zertifikaten. Mittlerweile haben sich einige Projekte und Systeme in diesem Umfeld entwickelt, z.B. PERMIS und Shibboleth. Auch der XML-Dialekt zur Definition von Zugriffskontrollpolitiken XACML basiert auf Benutzer- und Objektattributen.

ABAC kann als Generalisierung traditioneller Zugriffskontrollmodelle gesehen werden, da diese auf ABAC abgebildet werden können. Sowohl DAC und MAC als auch RBAC kann mittels Attribute-Based Access Control formuliert werden. Dabei geschieht zum Beispiel eine Abbildung von RBAC in ABAC dermaßen, dass ein Attribut „Role“ eingeführt wird, dessen Werte auf der Subjektseite die Rollenzugehörigkeiten darstellen. Daneben wird für jede Rolle ein entsprechender Subjektdeskriptor definiert. In Kombination mit Objektdeskriptoren, die jeweils ein bestimmtes Objekt bezeichnen, bildet die Rechtezuweisung zwischen den Deskriptoren das Analogon zur Beziehung zwischen Rollen und Berechtigungen.

2.2.5 Grundbegriffe verteilter Autorisierung

2.2.5.1 Kommunikationskonzepte zur Autorisierung

Zwischen Dienstanbieter, Dienstnutzer und dem Autorisierungsservice wird während der Autorisierung kommuniziert, um die Berechtigung des Benutzers für den Dienstzugriff sicherzustellen. Dabei werden drei Strategien unterschieden:

- The Agent Sequence: Bei dieser Strategie handelt der Autorisierungsservice als Agent im Auftrag des Dienstnutzers. Ist die Berechtigung des Benutzers sichergestellt, wird der angeforderte Dienst durch den Autorisierungsservice bereitgestellt.

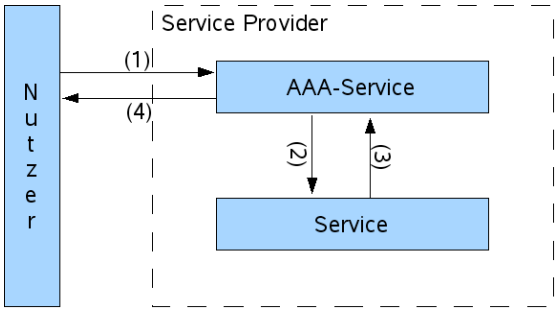

Abbildung 2: Agent Sequence

- The Pull Sequence: Wenn ein Benutzer auf einen Dienst zugreift, dann muss der Dienst, um die Berechtigung des Benutzers festzustellen, mit dem Autorisierungsservice kommunizieren. Für den Dienstnutzer ist diese Kommunikation völlig transparent.

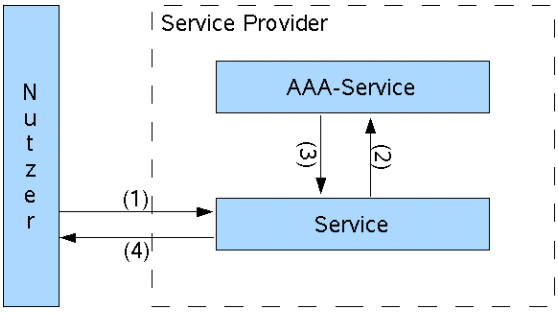

Abbildung 3: Pull Sequence

- The Push Sequence: Will ein Benutzer bei dieser Strategie einen Dienst nutzen, dann muss sich der Benutzer bereits im Vorfeld beim Autorisierungsservice einen Nachweis seiner Berechtigung anfordern. Mittels dieses Nachweises kann der Dienst die Berechtigung des Benutzers überprüfen.

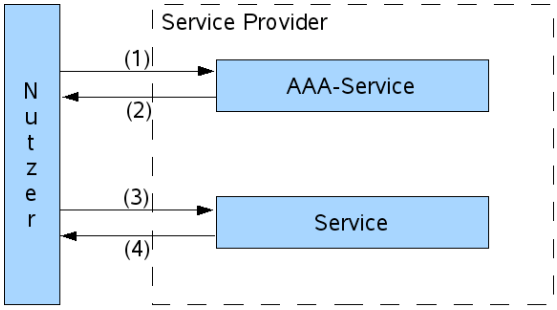

Abbildung 4: Push Sequence

Diese drei Strategien sind in den Abbildungen nur für Szenarien mit einer Autorisierungsdomäne dargestellt. Durch leichte Modifikation kann jedes dieser Szenarien erweitert werden, so dass auch der Zugriff auf Dienste in einer fremden Autorisierungsdomäne möglich ist (vorausgesetzt zwischen den Autorisierungsdomänen besteht eine Vertrauensbeziehung).

2.2.5.2 Autorisierung und Richtlinien

Beim Einsatz von Richtlinien (Policies) zur Autorisierung in einer verteilten Umgebung ist es von großer Bedeutung, wie Richtlinien abgefragt, umgesetzt und schließlich durchgesetzt werden können. Innerhalb einer Autorisierungsinfrastruktur lassen sich daher vier wesentliche Punkte identifizieren, die bei dieser Aufgabe von entscheidender Bedeutung sind:

- Policy Retrieval Point (PRP): Die Aufgabe des PRP ist die Bereitstellung des Zugriffs auf die gespeicherten Richtlinien. Dabei wird der Zugang zur Datenbank ermöglicht, in der die Richtlinien verwaltet werden, und eine Auswahl aus den Richtlinien getroffen, die im aktuellen Autorisierungsvorgang von Bedeutung sind.
- Policy Information Point (PIP): Insbesondere in verteilten Umgebungen spielt das Auffinden infrage kommender Richtlinien eine große Rolle. Diese Aufgabe löst der Policy Information Point, in dem die entsprechenden Aufbewahrungsorte für die jeweiligen Richtlinien ermittelt werden.
- Policy Decision Point (PDP): Das Ziel des PDP ist die Auswertung aller relevanten Richtlinien. Im Rahmen dieser Auswertung müssen häufig Informationen mit in Betracht gezogen werden, die nicht allein an einem Punkt vorliegen, z.B. ist es für den Autorisierungsdienst üblicherweise nicht möglich festzustellen, ob der Benutzer beim Zugriff auf den gewünschten Dienst, die in der Richtlinie geforderte Bandbreite auch zur Verfügung hat.
- Policy Enforcement Point (PEP): Die Durchsetzung der festgelegten Richtlinien wird vom Policy Enforcement Point sichergestellt.

Diese vier Punkte, die den Einsatz von Richtlinien in Autorisierungsumgebungen ermöglichen, sind üblicherweise nicht an einer zentralen Stelle innerhalb der Umgebung platziert. Wie bereits vorher angedeutet, müssen eine Vielzahl von Informationen bei der Auswertung mit in Betracht gezogen werden. In der folgenden Grafik sind die möglichen Stellen für diese Punkte daher skizziert.

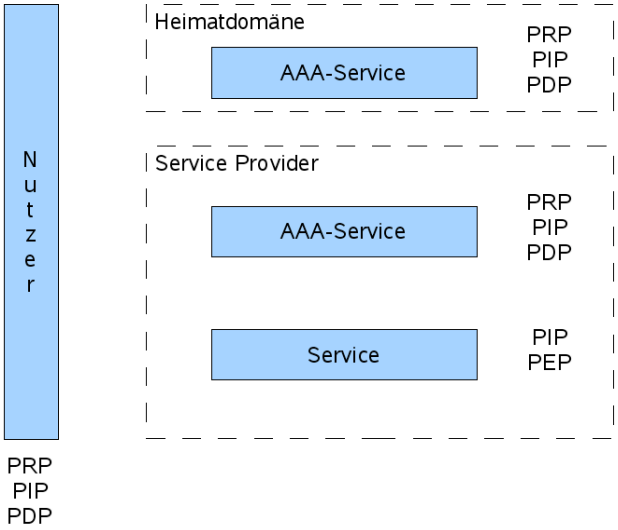

Abbildung 5: PRP, PIP, PDP und PEP

3 Public Key Infrastructure in Grid-Umgebungen

Die Authentifizierung im D-Grid findet zurzeit ausschließlich über Zertifikate gemäß Standard X.509 statt. Um die internationale Einbindung der D-Grid Partner zu gewährleisten, müssen die Zertifizierungsstellen (CAs) von der European Grid Policy Management Authority (EUGridPMA) [EGP06] akzeptiert werden. Bei der EuGridPMA handelt es sich um eine im Jahre 2004 von europäischen CA-Managern gegründete Organisation, die grundlegende Bedingungen für den Betrieb von Zertifizierungsstellen festlegt und durch ein einheitliches Verfahren Vertrauensbeziehungen zwischen wissenschaftlichen Einrichtungen in europäischen Ländern herstellt. 2005 erfolgte der Zusammenschluss mit The Americas Grid Policy Management Authority (TAGPMA) und der Asia Pacific Grid Policy Management Authority (APGridPMA) zur International Grid Trust Federation (IGTF) [IGTF05].

Die beiden deutschen EUGridPMA-konformen Zertifizierungsstellen werden vom Forschungszentrum Karlsruhe (GridKA-CA) und vom DFN-Verein (DFN-Grid-CA) betrieben. Zertifikate dieser beiden CAs sind gleichwertig. Die Wahl der CA für ihre Zertifikate ist den D-Grid Partnern freigestellt.

Das Forschungszentrum Karlsruhe und der DFN-Verein werden ein gemeinsames und tragfähiges Konzept im Laufe des D-Grid Projektes erarbeiten. Dabei wird geprüft, wie eine Integration der beiden CAs zu einer einheitlichen nationalen GridCA Dienstleistung gemäß den generellen Forderungen der EUGridPMA erfolgen kann.

3.1 Zertifikate

3.1.1 Informationen in Zertifikaten

Das generelle Namensschema für den Distinguished Name (DN) eines Zertifikates lautet (die Angaben in eckigen Klammern sind optional):

	DFN-Grid-CA	**GridKa-CA**
C[1]=	DE	DE
O[2]=	GridGermany	GermanGrid
OU[3]=	Name der Institution	Kurzform des Namens der Institution
[OU=]	Weitere OUs, z.B. Name der Abteilung	–
CN[4]=	Vorname Nachname	Vorname Nachname
	[bzw. Hostname.domäne.de oder Service/ Hostname.domäne.de]	[bzw. Hostname.domäne.de oder Service/ Hostname.domäne.de]
[...] weitere Informationen		
SubjectAlternative Name	E-Mail-Adresse des Benutzers oder Hostname.domäne.de	E-Mail-Adresse des Benutzers oder Hostname.domäne.de

[1] Country

[2] Organisation

[3] Organizational Unit

[4] Common Name

Da beide Zertifizierungsstellen in der EUGridPMA anerkannt sind, ist eine Unterscheidung des Namensraumes (O=GridGermany, O=GermanGrid) notwendig. Nur so kann die Eindeutigkeit des Distinguished Names international gewährleistet werden.

Bei den Zertifikaten handelt es sich entweder um persönliche oder so genannte Host- oder Server-Zertifikate. Da sowohl Personen als auch Ressourcen in mehreren Projekten teilnehmen können, werden im Zertifikat keine Informationen über Projekte angegeben. Die Gründe hierfür sind: Die Zugehörigkeit einer Person zu einem bestimmten Projekt und ihre Autorisierung wird durch die Zugehörigkeit zu einer virtuellen Organisation (VO) ausgedrückt. VOs wurden dafür entwickelt, um institutsübergreifende Tätigkeiten in Projekten zu ermöglichen. Hingegen existieren Zertifikate, um die Zusammengehörigkeit eines Zertifikats (bzw. des darin enthaltenen öffentlichen Schlüssels) zu einer bestimmten Person und somit ihre Authentifizierung zu gewährleisten. Durch die Trennung dieser beiden Bereiche Autorisierung und Authentifizierung erreicht man eine Reduktion der Komplexität und somit ein höheres Maß an Sicherheit.

Die beiden Wurzelzertifikate der DFN-Grid-CA und GridKa-CA, sowie das Benutzer- und Server-CA-Zertifikat der DFN-Grid-CA werden in allen Test- und Produktionsumgebungen des D-Grid installiert, d.h. im entsprechenden Verzeichnis hinterlegt, so dass die Communities ihre Arbeit ungehindert beginnen können. Alle Zertifikate werden auf dem D-Grid-Portal zum Download zur Verfügung gestellt werden.

Vom Aufbau weiterer Zertifizierungsstellen in Deutschland wird abgesehen, da diese neuen Zertifikate nicht international anerkannt werden würden.

Um den Registrierungsprozess für neue D-Grid Benutzer zu vereinfachen, werden im Rahmen von D-Grid so genannte Registrierungs-Autoritäten bzw. Registrierungsstellen (RA) aufgebaut. Jede wissenschaftliche Einrichtung kann eine oder mehrere solcher RA betreiben und damit den Zertifizierungsprozess aktiv unterstützen. Dazu muss die Policy der DFN-CA bzw. GridKA-CA anerkannt werden. Beim Aufbau solcher RAs an den verschiedenen D-Grid Standorten sind die CAs bei FZK und DFN-Verein behilflich. Die Anforderungen an Registrierungsstellen im D-Grid sind in Rahmen der „minimum requirements“ der EUGridPMA [MinR05] sowie in den Grid-Policies des Forschungszentrums Karlsruhe [GCP05] und des DFN-Vereins [DCP05] festgelegt.

3.1.2 Integration von nicht-akademischen Partnern

Anstelle des Institutionsnamens (oder der Kurzform) erscheint der Name (oder die Kurzform) des Partners in der „OU“ des Zertifikats. Zertifikate für nicht-akademische Partner können bei jeder RA von D-Grid beantragt werden. Nicht-akademische Partner können selbst eine RA betreiben. Dazu muss die Policy der DFN-CA bzw. GridKA-CA anerkannt werden.

3.1.3 Verwendung von Sonderzeichen

In Zertifikatnamen dürfen nur bestimmte Sonderzeichen verwendet werden. Alle im Folgenden nicht genannten Zeichen sind für Zertifikatnamen im D-Grid ausgeschlossen.

Erlaubte Zeichen sind:

```
a-z A-Z 0-9 ' ( ) , - . / : _ @ & Leerzeichen
```

Deutsche Sonderzeichen werden wie folgt ersetzt:

```
ä=ae, ö=oe, ü=ue, Ä=Ae, Ö=Oe, Ü=Ue, ß=ss
```

4 Shibboleth

Shibboleth ist ein Projekt des Middleware Architecture Committee for Education (MACE) im Internet2 Konsortium [Shibboleth]. In ihm werden Architekturen, Policy-Strukturen und Technologien entwickelt. Unter Shibboleth wird jedoch generell ein bestimmtes Produkt dieses Projektes verstanden: die Open-Source-Implementierung eines verteilten Systems zur inter-institutionellen Nutzung von zugangsgeschützten Web-Ressourcen. Im Bibliotheks- und eLearning-Bereich findet das System international eine stark zunehmende Verbreitung. Über diese Bereiche hinaus erfährt Shibboleth als allgemeine AA-Infrastruktur eine breite Beachtung. In den USA, der Schweiz, Großbritannien, Finnland und Australien werden bereits nationale Shibboleth-Infrastrukturen aufgebaut. In Deutschland ist sie derzeit im Aufbau. Im Grid-Umfeld wird Shibboleth als mögliche Ergänzung bzw. sogar Ersatz für PKI-Strukturen angesehen [Gri06][ESP06].

Eine Shibboleth-Transaktion erfolgt zwischen Identity Provider (IdP) und Service Provider (SP). Der IdP basiert auf dem Identity Management (IdM) der Heimat-Einrichtung des Nutzers, der SP steht für die Web-Ressource. IdPs und SPs bilden üblicherweise auf nationaler Ebene eine Föderation, deren Policy die Vertrauensbasis der teilnehmenden Partner ist. In der Policy wird zudem festgelegt, welches Attributschema eingesetzt werden muss. Als internationaler Quasi-Standard gilt das Schema „eduPerson“, das gegebenenfalls durch nationale Erweiterungen ergänzt wird [Haz06]. Shibboleth nutzt die Security Assertion Markup Language (SAML) um Zusicherungen über die Autorisierungen eines Nutzers vom IdP an einen SP zu übertragen.

Shibboleth ist derzeit in der Version 1.3 (Freigabe im Juli 2005) verfügbar und nutzt SAML Version 1.1. Shibboleth 2.0 auf der Basis von SAML 2.0 wird als Beta Release im Mai/Juni 2006 erwartet, das Final Release gegen Ende des Sommers 2006.

4.1 Shibboleth-Architektur

In Shibboleth kommt das Modell der Attribute-Based Access Control (ABAC, siehe 2.2.4.4) zum Einsatz. Shibboleth nutzt ggf. vorhandene Single-Sign-On-Systeme (SSO) als Authentifizierungskomponente. Autorisierungen werden in standardisierten Attributen (eduPerson) im institutionellen IdM abgelegt und in Form einer Zusicherung (SAML Assertion) an den SP übertragen. Dabei ist der Schutz der Privatsphäre des Nutzers möglich: Der Nutzername kann dem SP verborgen bleiben, wenn die zwischen IdPs und SPs getroffene Vereinbarung (Federation Policy) dies vorsieht.

Shibboleth gliedert sich in drei zentrale Komponenten:

- Identity Provider (IdP): Jede Einrichtung, die Mitglied einer Föderation wird, benötigt einen IdP. Der IdP setzt sich primär aus vier Teilen zusammen: der Attribute Authority (AA), dem Handle Service (HS), dem institutionellen IdM (Directory Server oder Datenbank) und dem lokalen SSO. AA und HS sind Bestandteil von Shibboleth. IdM und SSO muss der IdP-Betreiber bereitstellen.
- Service Provider (SP): Jede Einrichtung, die einen Dienst in einer Föderation bereitstellt, benötigt einen SP. Die wichtigsten Komponenten des SP sind der Assertion Consumer Service (ACS), der Attribute Requester (AR) und der Resource Manager (RM).
- Where Are You From (WAYF): Der WAYF-Dienst ist die zentrale Stelle einer Föderation, die alle teilnehmenden IdPs und deren Adressen kennt. Er gestattet es einem Nutzer seinen IdP zu identifizieren.

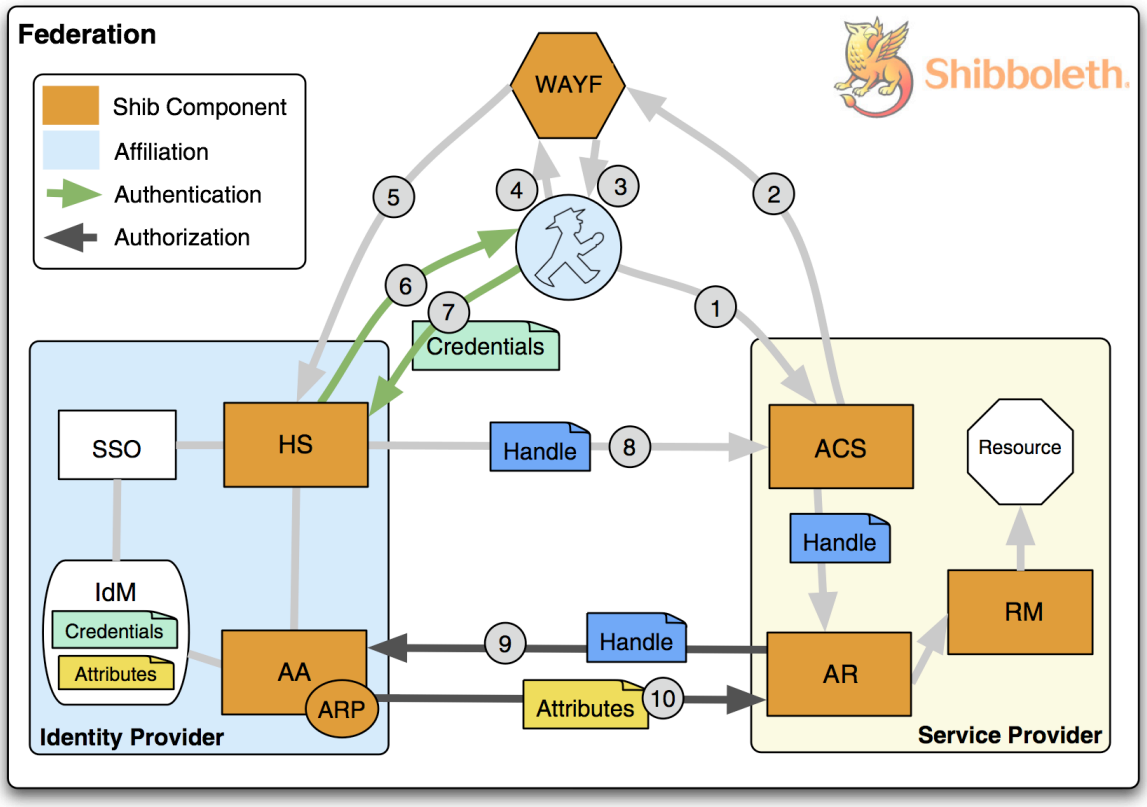

Abbildung 6: Shibboleth

Die Abbildung stellt überblicksartig den Ablauf einer Shibboleth-Transaktion beim Browser-basierten Zugriff auf eine Web-Ressource dar [Shi06b]:

1) Ein Nutzer möchte Zugriff auf eine durch Shibboleth geschützte Ressource eines Service Providers (SP) erhalten.

2) Der Nutzer wird zum Where Are You From (WAYF) umgeleitet.

3, 4) Der Nutzer wählt seine Heimat-Institution (IdP) aus.

5) Der Nutzer wird zum Handle Service seines IdP weitergeleitet.

6) ,7) Der Nutzer authentifiziert sich auf vertrautem Wege gegenüber seinem IdP.

8) Der Handle Service (HS) generiert eine eindeutige ID (Handle) und leitet den Nutzer zurück zum Assertion Consumer Service (ACS) des SP. Der ACS überprüft die mitgelieferte Assertion, generiert eine Session und übergibt an den Attribute Requester (AR).

9, 10) Der AR nutzt das Handle, um Attribute des Nutzers bei der Attribute Authority (AA) des IdP abzufragen. Die AA liefert unter Berücksichtigung der Attribute Release Policy (ARP) eine Attribute Assertion an den AR zurück. Der SP entscheidet anhand der erhaltenen Attribute über die Gewährung und Art des Zugangs.

4.2 Shibbolisierung des Grid

Unter Shibbolisierung des Grid wird eine umfassende Integration von Shibboleth und Grid verstanden. Damit wird auch verbunden, die Authentifizierung mit Shibboleth-Mitteln zu realisieren. In der Folge des ursprünglichen Ansatzes von GridShib (siehe Kapitel 5.6) sind in Großbritannien, Australien und der Schweiz Projekte entstanden bzw. in Vorbereitung, die diese weitergehende Integration von Shibboleth in Grids anstreben (Auswahl):

- GridShib mit erweitertem Konzept (USA, Laufzeit bis Q2/2007)

- MAMS (Australien, Laufzeit bis Ende 2006)
- SHEBANGS/GridSite (Großbritannien, Laufzeit 03/2006 bis 02/2007)
- Shibboleth in EGEE2 (Schweiz, 04/2006 bis 03/2008)
- ShibGrid (Großbritannien, Laufzeit 03/2006 bis 02/2007)

Die Gemeinsamkeiten in den Aufgabenstellungen dieser Projekte verdeutlichen das Potential des Shibboleth-Einsatzes in Grids.

4.3 VO-Management per Shibboleth

Im Shibboleth-Konzept ist der IdP der Heimateinrichtung eines Nutzers die Quelle der Autorisierung (AA, Attribute Authority). Eine VO als eine eigene „virtuelle Organisation“ ist in diesem Sinne eine weitere, eigene Quelle der Autorisierung (zu VO siehe auch 5.4.1). Shibboleth sieht jedoch nur eine AA als Bestandteil des IdP vor. Unter dieser Voraussetzung werden mehrere Ansätze diskutiert, um mit Shibboleth-Mitteln eine VO abzubilden:

1. Die VO-Verwaltung wird verteilt von den IdPs der zugehörigen Mitglieder übernommen. Die an der VO beteiligten Parteien einigen sich auf ein Schema und die Attribute.
2. Die VO wird durch einen eigenen IdP realisiert, was dem VOMS-Modell mit seinen Vor- und Nachteilen entsprechen würde. Mehrere Grid-Shibboleth-Integrationsprojekte haben eine Shibbolisierung von VOMS vorgeschlagen.
3. Eine Person und ihre Attribute werden im Identity Management (IdM) der Heimateinrichtung (IdP) verwaltet, die VO-spezifischen Attribute von der VO. D.h. die VO hat bestimmte Schreibrechte auf das IdM der IdPs. Für dieses Szenario geeignete Provisioning-Werkzeuge wären Grouper und Signet, die in Internet2 Middleware-Projekten entwickelt werden [Gro06][Sig06].

Die Ansätze 1 und 3 entsprechen dem Shibboleth-Konzept eine Nutzeridentität mit seinen Rechten und Rollen an einem Ort zu verwalten. Ein bestimmtes Maß an Vertrauen zwischen den IdPs und SPs einer VO vorausgesetzt, ist der Ansatz 1 der einfachste. Er kann mit jetzt verfügbaren Mitteln umgesetzt werden und gegebenenfalls bei Produktionsreife von Grouper und Signet in den Ansatz 3 überführt werden.

5 Komponenten und Verfahren

Die in diesem Bericht betrachteten Grid Middlewares basieren zum Teil auf denselben Ansätzen zur Realisierung der jeweiligen AA-Infrastrukturen. Deshalb stellt dieses Kapitel die allgemein verwendeten Komponenten und Verfahren von AA-Infrastrukturen in Grid-Umgebungen zusammen. Darauf aufbauend werden in Kapitel 6 die auf die jeweilige Grid Middleware bezogenen spezifischen Merkmale der AAI näher erläutert.

5.1 Transport Level Security und Message Level Security

Um eine sichere Kommunikation zwischen Kommunikationspartnern über eine unsichere Netzwerkverbindung zu ermöglichen, gibt es verschiedene Strategien. Im Grid-Kontext sind Transport Level Security und Message Level Security bedeutsam. Während bei der Transport Level Security ein sicherer Kanal zwischen den beiden Kommunikationspartnern ausgehandelt wird, über den die gesamte weitere Kommunikation verschlüsselt stattfindet, wird bei der Message Level Security üblicherweise jede Nachricht separat verschlüsselt, wobei auch nur ein Teil einer Nachricht verschlüsselt werden kann. Beide Verfahren sind in ihrer Anwendung transparent für die darüber liegende Anwendungsschicht, so dass auf diese Weise auch eine Absicherung von Klartext-Kommunikation möglich ist.

Üblicherweise werden die Protokolle SSL (Secure Socket Layer) oder TLS (Transport Layer Security) zur Realisierung von Transport Level Security eingesetzt. SSL ist ein von Netscape entwickeltes Protokoll, das erstmals im Web-Browser Mosaic und später im Netscape Navigator zum Einsatz kam. Im Rahmen der Standardisierung des Protokolls durch die IETF (Internet Engineering Task Force) wurden einige kleinere Veränderungen und Erweiterungen vorgenommen, so dass das resultierende Protokoll (das nicht mehr kompatibel zu SSL ist) einen neuen Namen erhalten hat: TLS.

Message Level Security hat entscheidet an Bedeutung gewonnen durch die Entwicklung von Web Services; daher sind die Standards WS-Security und WS-SecureConversation in diesem Zusammenhang von besonderer Bedeutung.

5.1.1 Protokolle zur Gewährleistung von Transport Level Security

SSL (und TLS – da aber die Unterschiede zwischen SSL und TLS im Bereich der Authentifizierung nicht wesentlich ins Gewicht fallen, wird im folgenden SSL und TLS synonym benutzt werden) erlaubt den Kommunikationspartnern, eine gesicherte Sitzung aufzubauen, d.h. einen Kontext, in dem beide Partner authentifiziert sind und Vertraulichkeit und Integrität aller Datenpakete gewährleistet ist. Das bedeutet aber auch, dass die Authentizität einer Nachricht Dritten gegenüber nicht nachgewiesen werden kann, da sie keine digitale Signatur trägt, sondern nur einen Message Authenticity Code (MAC). Für die Absicherung des SSL-Tunnels werden im SSL Record Protocol symmetrische und daher sehr effiziente kryptografische Verfahren (3DES, AES, HMAC etc.) auf die einzelnen Datenpakete (Größe bis zu 16 KByte), in die der Nutzdatenstrom zerlegt wird, angewandt.

Die verwendeten Algorithmen lassen sich beim Aufbau der Sitzung während des so genannten Handshakes abhängig von den Fähigkeiten und Sicherheitspolicies der Endpunkte dynamisch aushandeln. Auf diese Weise kann individuell den jeweiligen Anforderungen Rechnung getragen werden. Der Handshake dient dazu, ein „Master Secret“ zu erzeugen, aus dem die Partner jeweils die zufälligen symmetrischen Schlüssel für MAC und Verschlüsselung ableiten können. Diese Schlüssel werden auch als Sitzungsschlüssel oder kurzlebig (ephemeral) bezeichnet, da sie nur für die jeweilige SSL-Session gültig sind.

Demgegenüber verwenden die Partner im Grid-Umfeld asymmetrische Schlüsselpaare, die über mehrere Sitzungen hinweg gültig sein können. Die privaten Schlüssel und die zugehörigen X.509-Zertifikate (End-Entity-Zertifikate oder Proxy-Zertifikate, siehe Abschnitt 5.2.1) dienen der gegenseitigen Authentifizierung der Partner, die dabei die Kenntnis des privaten Schlüssels implizit nachweisen. Dabei stehen als Algorithmen das RSA-Kryptosystem, die Signaturverfahren RSA und

DSA oder Diffie-Hellman als Key Agreement-Verfahren zur Verfügung. Welchen dieser Algorithmen eine Seite unterstützt, ist in ihrem Zertifikat festgelegt, es können unterschiedliche Verfahren gewählt werden, d.h. der Client kann sich z.B. mit Hilfe des RSA-Kryptosystems authentifizieren während der Server DSA verwendet. Im Gegensatz zum Einsatz von SSL im WWW, wo sich meist nur der Server mit einem Zertifikat ausweist, ist beim Einsatz von SSL im Grid eine zertifikatsbasierte Authentifizierung des Clients obligatorisch. Eine Authentifizierung ist dann erfolgreich, wenn neben dem Nachweis der Kenntnis des privaten Schlüssels auch die gesamte Zertifikatskette im Schalenmodell als gültig erkannt wird. Dies bedeutet, dass die Gegenseite zum einen die ausstellende CA als vertrauenswürdig ansieht (was im Grid der Fall sein sollte, wenn es sich um ein Zertifikat der nationalen Root-CA wie o.g. handelt). Zum anderen muss der Gültigkeitszeitraum aller dazwischen liegenden Zertifikate (null oder mehrere Sub-CA-Zertifikate, genau ein EEZ, null oder mehrere PZe) den Prüfzeitpunkt enthalten und diese Zertifikate müssen vom jeweiligen Vorgänger-Zertifikat gültig signiert sein.

Das in diesem Kontext verwendete Protokoll ist daher genau genommen kein TLS im Sinne von RFC 2246 [DiAl99], da dort das spezielle Gültigkeitsmodell von Proxy-Zertifikaten nicht berücksichtigt ist (siehe Diskussion der Regeln für Path Validation nach RFC 3820 [TWE+04] im vorhergehenden Abschnitt). Abgesehen davon jedoch verläuft das Protokoll analog zu TLS: Die Kommunikationspartner senden ihr Zertifikat (EEZ oder PZ), evtl. auch die gesamte Zertifikatskette, um die Verarbeitung zu vereinfachen, und weisen die Kenntnis des zugehörigen privaten Schlüssels nach [JTE01].

5.1.2 Message Level Security

Seit der Einführung von Web Services, z.B. in GT4 wird zur Kommunikation SOAP über HTTP genutzt, werden Standards für Web Services entwickelt um Nachrichtenteile zu verschlüsseln. Realisiert wird dies mit den Spezifikationen zu WS-Security und WS-SecureConversation (siehe Abbildung 7), die auf weiteren Standards aus diesem Umfeld aufbauen.

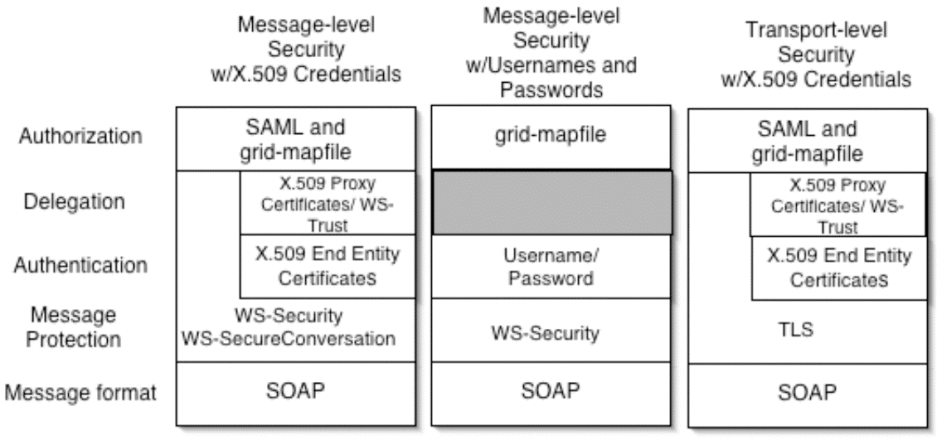

Abbildung 7: GSI-Alternativen in GT4 (aus [Welc05])

WS-Security erweitert die Nachrichtenübermittlung mittels SOAP um Verschlüsselung von Nachrichtenteilen (unter Verwendung von XML-Encryption), Nachrichtenintegrität und Authentifizierung. Weiterhin wird ein generischer und erweiterbarer Mechanismus zum Austauschen von Sicherheits-Tokens bereitgestellt. Insbesondere werden X.509-Zertifikate und Kerberos-Tickets unterstützt.

WS-SecureConversation baut auf WS-Security und WS-Policy auf und sorgt für eine sichere Kommunikation zwischen Diensten. Durch den Aufbau eines so genannten Sicherheits-Kontextes wird zwischen den Kommunikationspartnern eine sichere Verbindung aufgebaut, die gegen unterschiedliche Angriffe resistent ist, z.B. Replay-Attacken.

5.2 Nutzung und Verwaltung von Zertifikaten

5.2.1 Proxy-Zertifikate

5.2.1.1 Motivation

Die Ausführung eines Grid-Jobs findet naturgemäß nicht auf einem einzigen, vollständig vom Benutzer kontrollierten Computer statt, sondern erstreckt sich auf mehrere, unabhängig voneinander administrierte Systeme (Ressourcen), die per Internet miteinander verbunden sind. Der Zugriff auf einzelne Ressourcen ist üblicherweise nur nach vorheriger Authentifizierung bzw. Autorisierung mittels Credentials (X.509 Zertifikat und privater Schlüssel) möglich. Zudem kann möglicherweise nur indirekt über bestimmte andere, vorgeschaltete Ressourcen darauf zugegriffen werden (z.B. könnte ein Messinstrument nur über einen Instrument-spezifischen „Access Manager“ bedient werden, der wiederum nur von einem bestimmten Gateway-Rechner erreicht werden kann usw.). Ebenfalls denkbar sind komplexe Operationen, die eine Reihe von einfachen Grid-Jobs zusammenführen. In allen Fällen müssen Authentifizierungsmerkmale weitergegeben (delegiert) werden.

Setzt man hierfür ein einfaches X.509 End-Entity-Zertifikat (EEZ) plus entsprechendem privaten Schlüssel (= „Credential“) ein, müsste man entweder für jeden Authentifizierungsschritt auf den Rechner zurückgreifen, der das EEZ-Schlüsselpaar bereithält (Variante 1) oder das Schlüsselpaar zuzüglich Passphrase über das Netzwerk an die beteiligten Rechner verteilen (Variante 2). Beide Ansätze sind im Allgemeinen nicht anwendbar. Variante 1 zwingt den Benutzer, für die gesamte Bearbeitungsdauer des Grid-Jobs seine Credentials auf einem einzigen Rechner bereitzuhalten. Einerseits erfordert dies eine permanente Erreichbarkeit dieses Rechners, andererseits muss der Rechner die gesamte Kette an erforderlichen Operationen kennen, um gültige Zertifikat-Anfragen von unzulässigen zu unterscheiden. Variante 2 verletzt das Public-Key Sicherheitskonzept grundlegend, da der private Schlüssel und die dazugehörige Passphrase nicht mehr nur dem Benutzer selbst (z.B. auf einem selbst administrierten Computer), sondern unter Umständen auch Dritten zur Verfügung steht. Aufgrund der langen Laufzeit von EEZs (üblicherweise bis zu 13 Monaten) kann hierdurch ein beträchtlicher Schaden entstehen (gefolgt von einem administrativen Aufwand, dem Zertifikat-Widerruf mittels CRL).

5.2.1.2 Funktionsweise von Proxy-Zertifikaten in GSI

Eine praktikable Alternative stellen Proxy-Zertifikate (PZ) dar. Ein PZ übernimmt die Identität eines EEZ, erhält jedoch eine wesentlich kürzere Gültigkeit (wenige Stunden). Darüber hinaus kann das PZ mit Nutzungsbeschränkungen (mit Hilfe einer Policy) versehen werden, um die Gefahr eines Missbrauchs weiter einzudämmen. Um Benutzer-Interaktion bei Verwendung des PZ zu vermeiden, ist der zum PZ gehörende private Schlüssel nicht durch eine Passphrase geschützt, sondern nur durch die Dateizugriffsrechte auf dem jeweiligen System.

Proxy-Zertifikate sind ein entscheidender Bestandteil der Grid Security Infrastructure (GSI). Ihr Einsatz bringt folgende Vorteile:

- *Single Sign-On / Delegation* (siehe auch unten)*:* Authentifizierung per EEZ bzw. Passphrase erfolgt nur einmal, z.B. auf dem Client-Rechner des Benutzers, Prozesse können ohne weitere Authentifizierung im Namen des Benutzers laufen.
- *Einfache Integration in die bestehende PKI:* Verwenden von PZ anstelle von EEZ (mit leichten Änderungen am Zertifikatsgültigkeitsmodell).
- *Eindämmung des Schadens im Falle der Kompromittierung:* Aufgrund ihrer kurzen Gültigkeit und der Möglichkeit, das PZ mit Restriktionen zu verknüpfen, wird das Ausmaß eines potentiellen Schadens deutlich reduziert. Der Einfachheit halber wird daher im Grid kein Sperrmechanismus für PZe implementiert. Allerdings kann sich dies auch nachteilig auf die

Sicherheit auswirken, da vor dem Ablauf der Gültigkeitsdauer Rechte nur indirekt entzogen werden können (etwa mittels User Ban Lists).

- *Schutz der privaten Schlüssel:* Eine Entität A kann ein Proxy-Zertifikat zu einer Entität B delegieren, ohne dass B den privaten Schlüssel von A kennt und umgekehrt.

Beim Einsatz von PZe im Grid verwendet man gelegentlich die Begriffe Single Sign-On (SSO) und Delegation, um zwischen den folgenden beiden Varianten zu unterscheiden (siehe auch Abbildung 8). Allerdings lässt sich SSO auch als Spezialfall der Delegation auffassen, bei dem beide Parteien unter der Kontrolle des Delegierenden sind.

- SSO soll dem Benutzer mehrfaches Authentifizieren abnehmen. Dazu wird einmalig mit dem langlebigen EEZ und dem zugehörigen, gut geschützten privaten Schlüssel ein PZ für ein neues, lokal erzeugtes Schlüsselpaar, erstellt. Der zum PZ gehörige private Schlüssel ist lediglich durch die Dateizugriffsrechte geschützt, was es Prozessen erlaubt, das Schlüsselpaar auch ohne Benutzerinteraktion zu verwenden. Eine Bestätigung durch Passwort-Eingabe im Einzelfall oder Ähnliches wäre unpraktikabel.
- Delegation impliziert üblicherweise die Weitergabe von Privilegien an eine zweite Partei. Im Grid-Kontext handelt es sich dabei um Delegation über eine Netzwerkverbindung, die lediglich Authentizität und Integrität, jedoch nicht zwingend auch Vertraulichkeit gewährleisten muss. Nach einer beidseitigen Authentifizierung mit Hilfe von SSL erzeugt der Proxy ein Schlüsselpaar und einen mit dem neuen privaten Schlüssel signierten Certificate Request (Format: PKCS#10), den er an den Delegierenden schickt. Dieser erzeugt dann für den öffentlichen Schlüssel ein PZ mit den gewünschten Policy-Einschränkungen und schickt es über das Netzwerk zurück.

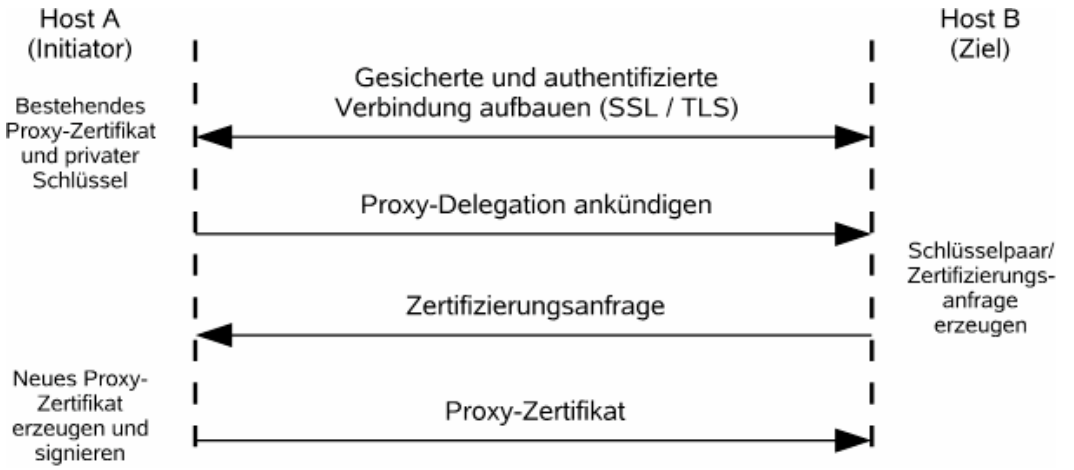

Abbildung 8: Delegation mit Proxy-Zertifikaten (aus [WFK+04])

5.2.1.3 Technische Details

GSI verwendet verschiedene Typen von Zertifikaten, die von ihrer *Syntax* her alle konform zum Standard X.509v3 [ITU97] sind, wenngleich die *Semantik*, d.h. die Gültigkeit, bei Proxy-Zertifikaten abweichend von jener bei Identitäts-Zertifikaten durch [TWE+04] erklärt ist.

Zum einen sind dies CA- oder SubCA-Zertifikate, mit denen weitere Zertifikate ausgestellt werden können. Solche Zertifikate haben gemäß dem Standard eine entsprechende Version 3-Erweiterung (`BasicConstraints: CA=true`). In End-Entity-Zertifikaten fehlt diese Erweiterung oder sie trägt den Wert `false`. In den Minimum Requirements der EuGridPMA [MinR05] wird explizit verlangt, dass die Extension bei EEZen enthalten und als „critical“ markiert sein muss[5]. Dies verhindert, dass mit dem

[5] http://www.gridpma.org/docs/IGTF-AP-classic-20050828-4-01.pdf

zugehörigen privaten Schlüssel weitere Identitäts-Zertifikate signiert werden können. Um dennoch die Möglichkeit zu schaffen, dass Benutzer, die nur über ein EEZ verfügen, weitere Zertifikate für die Delegation – jedoch keine allgemeinen Identitäts-Zertifikate – ausstellen können, wurde der Typ des Proxy-Zertifikats eingeführt. Proxy-Zertifikate sind X.509v3-Zertifikate mit einer Extension, die als kritisch markiert ist (das bedeutet, dass Anwendungen, die nicht mit Proxy-Zertifikaten umgehen können, diese generell als ungültig ablehnen). Mit Proxy-Zertifikaten lassen sich lediglich weitere Proxy-Zertifikate ausstellen.

Im Gegensatz zu Zertifikaten, die von einer CA ausgestellt werden, gibt es keine Möglichkeit, Proxy-Zertifikate direkt zurückzurufen. Auch wenn die Gültigkeitsdauer von PZ nur relativ kurz ist, etwa wenige Stunden, kann die Möglichkeit, sie zu widerrufen, nützlich sein und wird möglicherweise zukünftig ein Feature der GSI werden [WFK+04]. Auf die Gültigkeit von PZ lässt sich auch jetzt schon Einfluss nehmen, indem das übergeordnete EEZ gesperrt wird (die oben referenzierten nationalen Grid-Root-CAs unterstützten Sperrungen mittels X.509v1 oder v2-CRLs). Allerdings ist dies für den Benutzer mit weit reichenden Konsequenzen verbunden, da sich sämtliche in seinem Namen laufenden Prozesse nicht mehr authentifizieren können und er sich außerdem ein neues EEZ ausstellen lassen muss (was ggf. eine aufwändige Neuidentifikation durch die CA erfordert). Eine Alternative zur Verwendung von Sperrlisten für PZe ist es, PZe mit relativ kurzer Gültigkeit zu verwenden, die periodisch automatisch verlängert werden sofern die Berechnung andauert und kein „Sperrgrund“ vorliegt.

5.2.1.4 Policy-Einschränkungen

In RFC 3820 [TWE+04] ist für Proxy-Zertifikate die Extension ProxyCertInfo (OID: 1.3.6.1.5.5.7.1.14, auch als PCI-Extension bezeichnet) vorgeschrieben[6], die aus einer optionalen Längenbegrenzung der Kette aufeinander folgender Proxy-Zertifikate (pCPathLenConstraint) sowie einer verpflichtenden Angabe der ProxyPolicy besteht. Diese Autorisierungs-Policy kann alle oder keine Rechte beinhalten (siehe unten) oder beliebig fein (so genannte „Restricted Delegation“) formuliert werden. Dazu lässt sich eine beliebige Policy-Sprache verwenden, die jedoch von allen beteiligten Systemen auch verstanden werden muss. Im Falle der Restricted Delegation werden allerdings die Implementierungen aufwändiger. Relying Parties müssen nicht nur die Semantik der Delegations-Policy verstehen, sondern auch die entsprechenden Beschränkungen durchsetzen können. Da diese Policies oft anwendungsspezifisch sind, ist es für die darunter liegende Security Library, die die Authentifizierung über PZ abwickelt, schwierig zu erkennen, welche Beschränkungen die Anwendung tatsächlich kennt und durchsetzen kann. Ohne diese Gewähr kann kein PZ zuverlässig akzeptiert werden. Dieses Problem wurde durch eine Erweiterung der GSS-API adressiert (siehe unten). Da der Ansatz aber aufgrund der Abhängigkeit von der Anwendung kompliziert ist, wird diese Technik in der Praxis nur wenig eingesetzt [WFK+04].

Wo eine detaillierte Policy direkt in das PZ codiert wird, wird die Policy-Sprache über einen Object Identifier ausgewählt, der die Interpretation des entsprechenden Inhalts im Feld Policy regelt.

```
ProxyCertInfo ::= SEQUENCE {
      pCPathLenConstraint   INTEGER (0..MAX) OPTIONAL,
      proxyPolicy           ProxyPolicy }

ProxyPolicy ::= SEQUENCE {
      policyLanguage        OBJECT IDENTIFIER,
      policy          OCTET STRING OPTIONAL }
```

[6] In Middleware die auf dem Globus Toolkit basiert wird, statt die Extension ProxyCertInfo zu verwenden, "CN=proxy“ an den Distinguished Name des Zertifikatsinhabers angehängt.

Im RFC sind bereits zwei spezielle „Sprachen“ definiert, die von allen Systemen verarbeitet werden können müssen. Zum einen ist dies eine „inheritAll“-Policy, bei der – entgegen dem Least Privilege-Prinzip – sämtliche Rechte vom Aussteller an den Proxy delegiert werden (auch als „Impersonation Mode“ bezeichnet [WFK+04]), zum anderen eine „independent“-Policy, bei der überhaupt keine Rechte delegiert werden. Die Independent-Policy hat dennoch ihre Berechtigung, da diese PZs alleine zwar nicht für Berechnungen im Grid, jedoch aber z.B. für GridFTP eingesetzt werden können [WSF+03]. Darüber hinaus lassen sich über ein Attribut-Zertifikat oder auf anderem Wege dem Proxy doch noch und unabhängig von dem PZ wieder Rechte zuweisen [WFK+04]. Ein Nachteil dieser Variante ist jedoch die fehlende Protokollunterstützung für Attribut-Zertifikate in den Anwendungen. Web Services könnten hier eine Lösung sein, siehe [WSF+03].

5.2.2 Credential Wallets

Unter „Credential Wallet” (auch: „Credential Repository”) versteht man allgemein einen Speicher für Identitäts- bzw. Authentifizierungsmerkmale (üblicherweise X.509-Zertifikate und Schlüssel). Hier werden verschiedene Credentials eines Benutzers oder einer größeren Organisationseinheit konzentriert und in verschlüsselter Form abgelegt und wieder zur Verfügung gestellt. Der Benutzer erhält mit einem einzigen Schlüssel Zugang auf alle von ihm abgelegten Credentials. Die Metapher der Brieftasche (Wallet) bezieht sich auf die Tatsache, dass ein Benutzer nicht nur ein, sondern eine ganze Reihe unterschiedlicher Credentials besitzt, jeweils für unterschiedliche Zwecke bestimmt und ggf. durch unterschiedliche CAs zertifiziert.

Ein bekanntes Credential Repository System ist die Smart-Card, auf der ein oder mehrere Schlüsselpaare abgelegt werden können. Der Zugriff auf die Credentials wird z.B. durch Eingabe einer PIN kontrolliert. Die Smart-Card fungiert somit als sicherer und insbesondere mobiler Datenträger, da er in jedem kompatiblen Lesegerät verwendet werden kann.

Übertragen auf die Grid-AAI bedeutet diese Mobilität, dass Credentials von allen Rechnern (Ressourcen) aus online angefordert werden können, die der Benutzer für die Bearbeitung von Grid-Jobs benötigt. Hierbei wäre eine Smart-Card am Terminal des Benutzers zu unflexibel, da eben dieses nicht mobil sein dürfte, und zudem ständig für den Zugriff durch andere Grid-Ressourcen erreichbar sein müsste.

Eine Alternative zur physischen Smart-Card ist das Online Credential Repository, die „Virtual Smart-Card“. Hier werden die Credentials online auf einem gehärteten Server aufbewahrt. Der Zugriff erfolgt nicht mehr zwangsläufig durch Eingabe einer PIN, sondern kann alle Formen von Authentifizierung nutzen (Passwort, Public-Key Verfahren usw.).

Man unterscheidet zwei Typen von Online Credential Repositories:

- mechanismus-neutrale:
  Sie dienen dem sicheren Ablegen und Abrufen verschiedener Typen von Credentials, deren Struktur vom Repository nicht explizit unterstützt werden muss.
  Beispiel: IETF SACRED-Protokoll (RFC 3760); SACRED Implementierung siehe [BSXH05]
- mechanismus-bewusste:
  Sie erlauben nur das Ablegen bestimmter Credential-Formate. Dabei machen sie sich die Struktur der gespeicherten Credentials zunutze, um z.B. anstelle des gespeicherten X.509-Zertifikats ein Proxy-Zertifikat zurückzuliefern.
  Beispiel: MyProxy (siehe Abschnitt 5.2.3)

In beiden Fällen ist eine ähnlich professionell verwaltete Systemumgebung (Betriebssystem und Netzwerk) wie bei Online-CAs unabdingbar (in Bezug auf Sicherheit, Performanz, Skalierbarkeit, Verwerden; Ansätze hierzu finden sich in [LBK04].

5.2.3 MyProxy

MyProxy ist ein mechanismus-bewusstes Online Credential Repository-System zur Verwaltung von X.509-Zertifikaten. Das System wurde insbesondere für Proxy-Credentials konzipiert, mittlerweile (ab Version 3.0) unterstützt es auch das reine Ablegen/Abrufen von EEZ. MyProxy wird seit dem Jahr 2000 erfolgreich in verschiedenen Grid-Umgebungen eingesetzt (u.a. NEESgrid, TeraGrid, EU DataGrid, NASA Information Power Grid) und ist seit April 2005 fester Bestandteil des Globus Toolkit 4 (allerdings nur in Version 0.6.5).

Vergleichbar mit SACRED besteht MyProxy aus einem offenen Protokoll [Bas05] und einer Referenz-Implementierung [NCSAMyProxy]. Die Referenz-Implementierung, geschrieben in C, bietet verschiedene Kommandos sowohl für die Client- als auch für die Server-Seite. Neben der Referenz-Implementierung gibt es u.a. Clients für Java (JavaCoG), Python (pyGlobus) und Perl (Gridport 2).

Das Protokoll definiert eine TCP/IP-basierte, Client-Server-Schnittstelle, über die Zertifikate zum einen gespeichert bzw. verwaltet werden können (Statusabfrage bzw. Löschen), zum anderen können für die gespeicherten Zertifikate Proxy-Zertifikate angefordert werden. Das Repository übernimmt hierbei die Ausstellung signierter X.509-Proxy-Zertifikate anhand entsprechender Zertifizierungsanfragen.

Der Ablaufplan zum Erzeugen eines Proxy-Credentials mit MyProxy sieht wie folgt aus:

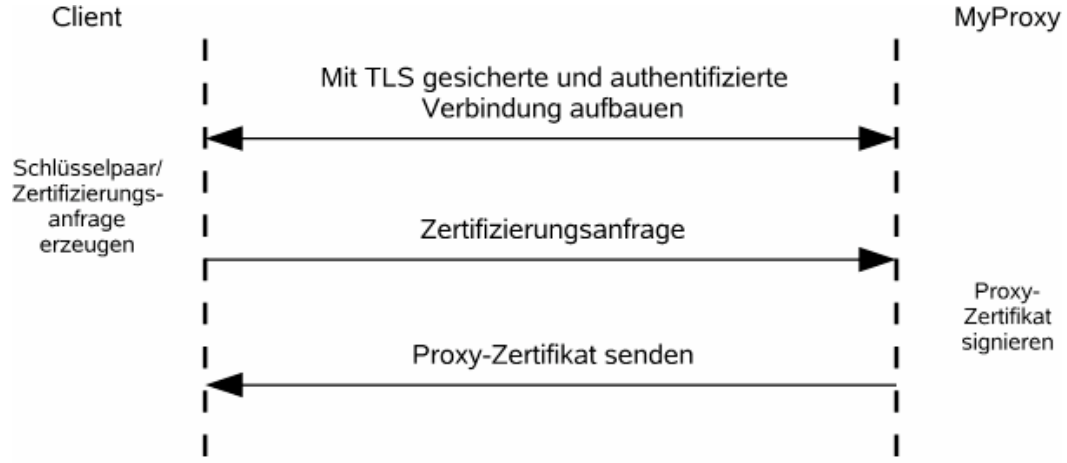

Abbildung 9 Proxy-Zertifikate mit MyProxy

Die Verbindung vom Client zum Server wird per TLS gesichert. Je nach Befehl wird die TLS-Identität zur Autorisierung verwendet oder nicht (bei manchen Anfragetypen ist sie optional). Die Autorisierung kann neben der reinen TLS-Identifizierung per SASL-GSSAPI [Mye97] erfolgen (auch mittels PAM, Kerberos, PubCookie, Einmal-Passwörtern, Subject-Policies[7] o.ä.)

Nach erfolgreicher Authentifizierung kann der Client eine Anfrage an den Server stellen. Jede Anfrage wird vom Server mit einer Antwort quittiert.

5.2.3.1 Proxy-Renewal

Eine wichtige Funktionalität des MyProxy-Servers ist das Erneuern von Proxy-Zertifikaten. Dies ist für solche Jobs wichtig, deren Abarbeitung einen größeren Zeitraum benötigt, als die Proxy-Zertifikate gültig sind. Hierfür muss der Client, dies kann ein Computing Element oder auch ein Workload-Management System sein, rechtzeitig vor Ablauf des Proxy-Zertifikats beim MyProxy-Server ein neues Proxy-Zertifikat anfordern.

---

[7] Hiermit kann der Zugriff auf MyProxy anhand der verwendeten Credential-Subjects eingeschränkt werden, z.B. um nur das Speichern von D-Grid Credentials zuzulassen oder um das Anfordern von Proxy-Credentials nur einer bestimmten Gruppe von Ressourcen zu ermöglichen.

Die Authentifizierung erfolgt hierbei durch ein Challenge/Response-Verfahren, wobei überprüft wird, ob der Client tatsächlich im Besitz des Proxy-Credentials ist. Der MyProxy-Server sendet hierzu eine Challenge, die vom Client mittels privatem Schlüssel des Proxy-Credentials verschlüsselt an den Server zurückgegeben wird. Lässt sich dieser wieder mit dem auf dem MyProxy-Server gespeicherten öffentlichen Schlüssel entschlüsseln, kann davon ausgegangen werden, dass der Client im Besitz des Proxy-Credentials ist und damit zur Verlängerung berechtigt ist. Ist dies der Fall, lässt sich ein neues Credential anfordern. Um Missbrauch zu verhindern, kann das Credential auf dem MyProxy-Server durch ACLs geschützt werden, die eine Erneuerung des Proxy-Zertifikates nur für bestimmte Entitäten zulassen.

Ein Beispiel für ein Job-Verwaltungssystem, das Proxy-Renewal unterstützt, ist Condor-G [Con06].

5.3 Grid Security Infrastructure

Die Grid Security Infrastructure (GSI) ist 1997 ursprünglich als Sicherheitsschicht des Globus Toolkit (GT) entwickelt worden [FKTT98]. Die Dienste des GSI werden jedoch auch oftmals in anderen Middlewares wie LCG und gLite (siehe Abschnitte 6.3 und 6.4) und Projekten verwendet, beispielsweise in SAMD (Seamless Access to Multiple Datasets[8]) oder SRB (Storage Resource Broker[9]). Die folgende Beschreibung geht von der ursprünglichen GSI-Architektur in Globus Toolkit 2.x (GT2) aus (d.h. ohne Web Services, „pre-WS“). Die technischen Details über GT2 sowie die aktuelle Version 4 (GT4) finden sich in den Abschnitten 6.1 und 6.2.

5.3.1 Begriffe und Annahmen der Security Policy

Die Begriffe „Subjekt“, „Objekt“, „Credential“ und „Domäne“ (Trust Domain) im Zusammenhang mit Authentifizierung und Autorisierung wurden bereits in Abschnitt 2.1 eingeführt. Im Folgenden wird die Security Policy für ein Grid zusammengefasst, wie sie der Architektur für GSI zugrunde liegt [FKTT98]. In dieser Grid Security Policy, auch „globale“ Policy in Unterscheidung zur „lokalen“ Policy der einzelnen Domänen genannt, sind die folgenden Annahmen in Form von Regeln über Subjekte, Objekte und deren Beziehungen getroffen.

1. Das Grid ist eine heterogene und dynamische Umgebung aus vielen verschiedenen Domänen.
2. Operationen über Domänengrenzen hinweg erfordern eine gegenseitige Authentifizierung der Kommunikationspartner.
3. Für Operationen innerhalb einer einzelnen Domäne ist ausschließlich die lokale Policy mit den entsprechenden Sicherheitsmechanismen maßgebend.
4. Es gibt lokale und globale Subjekte. Die Domänen können (in Eigenregie und unabhängig voneinander) ein globales Subjekt auf ein lokales Subjekt abbilden.
5. Ein globales Subjekt, das auf ein lokales abgebildet wird, ist in der entsprechenden Domäne vollständig äquivalent zu diesem.
6. Über den Zugriff auf Objekte in einer Domäne wird ausschließlich lokal entschieden (auf Basis des jeweils lokalen Subjekts).
7. Delegation: Prozesse können im Namen eines Benutzers mit dessen Rechten oder einer Teilmenge davon ausgeführt werden.

[8] http://www.sve.man.ac.uk/Research/AtoZ/SAMD/

[9] http://www.sdsc.edu/srb/

8. Prozesse verwenden Credentials gemeinsam, falls sie innerhalb derselben Domäne im Namen desselben Subjekts laufen.[10]

Abbildung 10: GSI-Komponenten

5.3.2 Übersicht Schutzziele und Sicherheitsmechanismen

Die (Schutz-)ziele von GSI sind die folgenden[11]:

1. sichere, d.h. authentische und/oder vertrauliche[12], Kommunikation zwischen Grid-Komponenten, wobei
2. die Sicherheit auch in einer dynamischen und offenen Umgebung, d.h. über organisatorische Grenzen hinweg und unter dezentraler Kontrolle gewährleistet sein soll,
3. Unterstützung eines einfachen und benutzerfreundlichen Single Sign-On (SSO) für Grid-Benutzer sowie eine Möglichkeit zur Delegation von persönlichen Credentials an Dienste, die im Namen des Benutzers auf dem Grid laufen sollen.

Die oben genannten Sicherheitsziele werden durch die entsprechenden technischen Mechanismen umgesetzt wie sie im Folgenden genannt und in Abbildung 10 schematisch dargestellt sind.

1. Mit einer Absicherung der Kommunikationskanäle durch SSL/TLS (s. Kap. 4.1.1),
2. einer Public Key Infrastructure mit mehreren unabhängigen Certification Authorities (CAs), die Zertifikate für die beteiligten Benutzer und Systeme ausstellen,
3. sowie durch Proxy-Zertifikate und der dadurch verbundenen Möglichkeit, Benutzerrechte (komplett oder teilweise) vorübergehend, d.h. für die Dauer einer Berechnung auf dem Grid, an unabhängige Prozesse zu delegieren.

Im Folgenden werden die für die Grid Security Policy nötigen Mechanismen im Detail erläutert. Die hier betrachteten Techniken

- PKI (siehe Kapitel 3),
- X.509 Version 3-Zertifikate [ITU97],
- SSL/TLS [DiAl99],
- GSS-API [Linn00]

sind (alt-)bekannte und etablierte Standards, während mit

- gewissen Erweiterungen der GSS-API [MWTE04],
- Proxy-Zertifikaten [TWE+04],
- und – darauf aufbauend – Single Sign-On und Delegation [JTE01, ABM04]

neue, Grid-spezifische Mechanismen geschaffen worden sind.

---

[10] Die Privilegien verschiedener Prozesse lassen sich einfacher trennen, wenn man ein eigenes Schlüsselpaar für jeden Prozess verwendet. Auch die Revokation der Delegation durch einfaches Löschen des Private Key ist dann möglich [WFK+04].

[11] http://www.globus.org/security/overview.html

[12] Um mögliche Exportbeschränkungen zu vermeiden, ist Verschlüsselung optional [FKTT98].

Abbildung 11 zeigt eine Übersicht über die Grundoperationen von GSI: Ein Benutzer, der über ein X.509-End-Entity-Zertifikat (EEZ) verfügt, erstellt damit ein User Proxy-Zertifikat (UPZ) mit kurzer Gültigkeitsdauer (schraffiert dargestellt), das zur Stellvertreter-Authentifizierung der vom Benutzer initiierten Prozesse etc. dient. Beim Zugriff auf Ressourcen (im Beispiel solche der Domäne 1) ist eine beidseitige Authentifizierung von Host und Zielrechner erforderlich. Das Zertifikat der Ressource ist hier mit RZ_1 bezeichnet. Nach erfolgreicher Authentifizierung wird der global Benutzer mit dem im EEZ angegebenen, global eindeutigen Distinguished Name, einem lokalen Benutzer zugeordnet. Dies ist dadurch gewährleistet, dass alle Mitglieder der IGTF (darunter auch die EUGridPMA) mit disjunkten Namensräumen arbeiten. Der globale Benutzer erhält somit die Rechte des entsprechenden lokalen Subjekts in der Domäne 1 und kann dort entsprechende Prozesse starten. Diese Prozesse erhalten wiederum Proxy-Zertifikate (im Beispiel mit PPZ_1 bis PPZ_k gekennzeichnet), diesmal ausgestellt von UPZ. Die im Namen des Benutzers laufenden Prozesse können sich somit wiederum einer anderen Domäne (im Beispiel als Domäne 2 bezeichnet) gegenüber authentifizieren und dort ebenfalls Prozesse starten, die entsprechende Credentials (Proxy-Zertifikate PPZ'_1 bis PPZ'_l) erhalten. Auch hier findet wiederum eine Abbildung des durch EEZ gegebenen globalen Subjekts auf ein lokales Subjekt der Domäne 2 statt. Wie oft sich dieser Schritt der Delegation wiederholen lässt, kann durch entsprechende Angabe einer „Pfadlänge“ in UPZ beschränkt werden. Zwischen Proxy-Zertifikaten, die auf dasselbe EEZ zurückgehen, besteht im Grid implizites Vertrauen, d.h. dass die zugehörigen Prozesse (im Beispiel mit den Zertifikaten: UPZ, PPZ_1, …, PPZ_k, PPP'_1, …, PPZ'_l) kooperieren und untereinander vertraulich und authentifiziert Daten austauschen können.

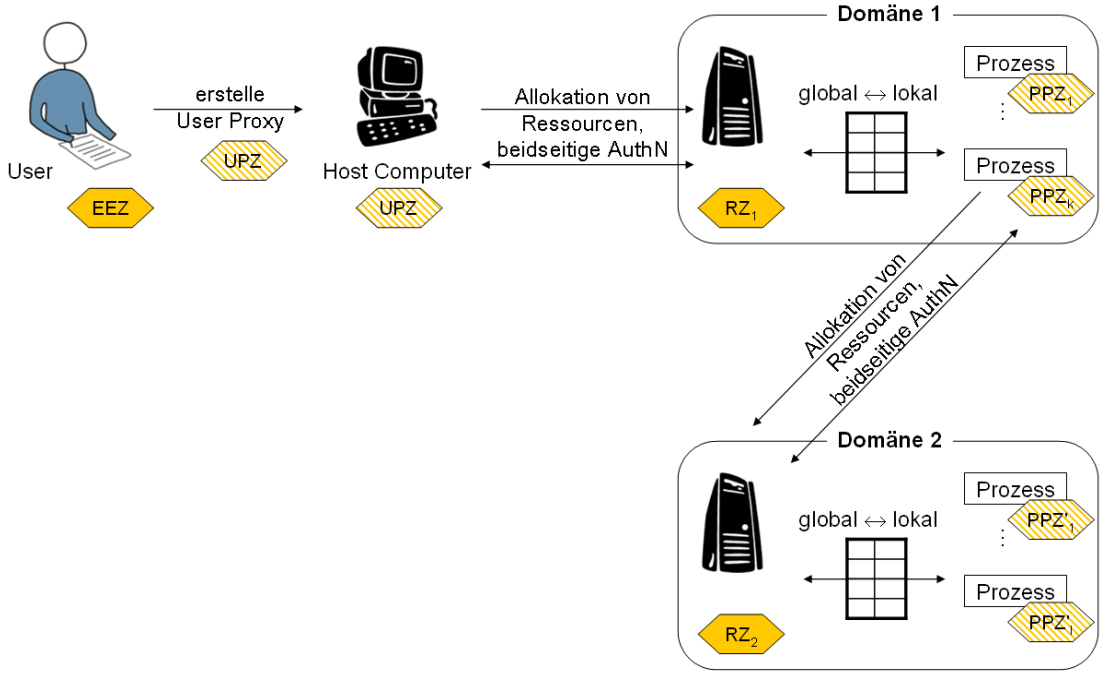

Abbildung 11: Grundoperationen von GSI (schematisch)

5.3.3 Nutzung von Public Key Infrastructure (PKI)

Subjekte und Objekte authentifizieren sich gegenseitig mit Hilfe starker Kryptografie unter Verwendung von X.509-Zertifikaten und dem zugehörigen privaten Schlüssel. Dies geschieht im Rahmen des geringfügig modifizierten Handshake Protocols innerhalb von SSL/TLS (s. Kap. 5.1.1). Die dafür notwendigen Personen- oder Maschinen-Zertifikate können entweder mit entsprechenden Tools (z.B. der im Globus Toolkit enthaltenen SimpleCA) selbst erstellt oder von einer bekannten CA (Certification Authority) bezogen werden. Entsprechende CAs werden vom Globus Certificate

Service[13] (ohne aufwändige Identitätsprüfung, daher sind diese Zertifikate nur für Testzwecke geeignet) oder nationalen Einrichtungen für Benutzer in dem jeweiligen Land betrieben[14].

Die Grid-PKI ist im Detail bereits in Kapitel 3 beschrieben worden. Im Folgenden werden daher lediglich einige Gründe aufgeführt, die die Entscheidung pro PKI als Sicherheitsmechanismus im Grid und contra andere Techniken beeinflusst haben (nach [WSF+03]).

- *Kerberos* erfordert eine explizite Beziehung zwischen verschiedenen Domänen und ist damit administrativ aufwändiger. Zwar unterstützt Kerberos auch Delegation, jedoch nur unter Beteiligung einer Third Party, während X.509 Proxy-Zertifikate unilateral ausgestellt werden können.
- *CRISIS*[15] stellt zwar eine einheitliche und skalierbare Infrastruktur für verteilte Systeme zur Verfügung, adressiert jedoch nicht das Problem der Interoperabilität mit lokalen Sicherheitsmechanismen.
- *SSH* (Secure Shell) unterstützt starke (d.h. Public Key-basierte) Authentifizierung und vertrauliche Kommunikation, erlaubt aber keine einfache Delegation vergleichbar dem Mechanismus der PZe.
- *Legion*[16] besitzt ein ähnliches Sicherheitsmodell wie das Globus Toolkit und verwendet ebenfalls X.509-Zertifikate für Delegation.

5.3.4 GSS-API (bisheriger Umfang und GSI-Erweiterungen)

Das Generic Security Service Application Programming Interface (GSS-API, RFC 2743/2744) unterstützt die einfache Programmierung von Sicherheitsmechanismen für Netzwerk-Anwendungen. Es folgt hier nur eine kurze Einführung in die Konzepte (siehe etwa [Sun00] für eine ausführlichere Darstellung).

Die GSS-API ist insofern generisch, als dass sie die Implementierung unabhängig von einem bestimmten Sicherheitsmechanismus (mechanism independence, z.B. in Bezug auf das Credential-Format), dem Transportprotokoll (protocol independence, möglich sind etwa RPC oder Sockets) oder auch einer bestimmten Plattform hält. Dies fördert in besonderem Maße die Portabilität. GSS-API implementiert jedoch selbst keine Sicherheitsdienste, sondern stellt nur ein standardisiertes Framework bereit, über das Anwendungen generisch Mechanismen wie etwa Kerberos oder PKI nutzen können. Die Auswahl der konkreten zugrunde liegenden Verfahren kann der Programmierer GSS-API überlassen und Vorgabewerte wählen oder selbst einzelne Verfahren auswählen (so genannte Quality of Protection – QOP). In letzterem Fall kann allerdings die Portabilität beeinträchtigt werden. GSS-API lässt sich unter C und Java verwenden.

GSS-API leistet zweierlei:

1. Es erlaubt Anwendungen, einen vertrauenswürdigen „Security Context“ herzustellen, in dem sie untereinander Daten austauschen können.
2. GSS-API stellt in jedem Fall Authentifizierungsverfahren und – sofern dies die zugrunde liegende Technologie unterstützt – Integritätsschutz und Verschlüsselung bereit.

---

[13] http://gcs.globus.org:8080/gcs/index.html

[14] Eine Liste der CAs findet sich etwa unter http://marianne.in2p3.fr/ca/ca-table-ca.html

[15] http://www.cs.utexas.edu/users/less/publications/research/crisis.usenix98.pdf

[16] http://www.cs.virginia.edu/~humphrey/papers/NDSS00.pdf

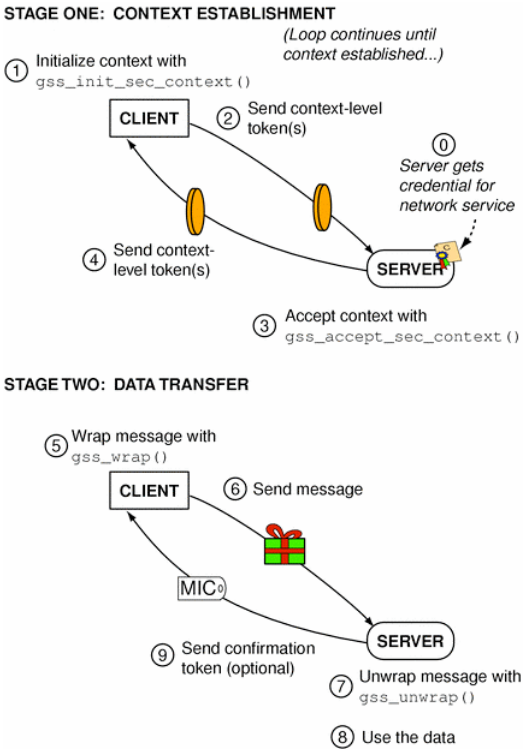

Abbildung 12: Verwendung der GSS-API (aus [Sun00])

Die GSS-API verwendet den Begriff „Principal“ zur einheitlichen Bezeichnung von Netzwerkentitäten wie Benutzern, Programmen oder Maschinen, zwischen denen GSS-API-basierte Transaktionen stattfinden. Als „Token“ wird ein Datenpaket bezeichnet, das Principals austauschen. Context-Level Tokens enthalten Informationen zum Aufbau oder Management eines Security Contexts, etwa ein Client Credential und den Namen des Zielrechners. Ein Message-Level Token ist etwa ein Message Integrity Code (MIC), der den Nutzdaten zur Integritätssicherung hinzugefügt wird. Die Verwendung von Tokens ist in Abbildung 12 dargestellt.

Die GSI-Erweiterungen von GSS-API sind in [MWTE04, Enge04] beschrieben. Diese relativ geringfügigen Änderungen betreffen den

1. *Import und Export von Credentials* zwischen Prozessen (auch GSS-API-fremden Anwendungen),
2. Das Ermöglichen der *Delegation zu einem beliebigem Zeitpunkt* (GSS-API in der ursprünglichen Version erlaubt dies nur während des Aufbaus des Security Context; diese Funktion kann nun dafür verwendet werden, Credentials zu aktualisieren),
3. eine Erweiterung des *Handlings von Credentials* um die Möglichkeit, Certificate Extensions oder Policy-Informationen zu verarbeiten,
4. die *Parameter-Übergabe an den Security Context*, z.B. um Verschlüsselung auszuschalten oder Beschränkungen für die Delegation festzulegen.

Details der C-APIs finden sich in [MWTE04]. Anwendungen, die auf der GSS-API aufsetzen und GSI unterstützen, sind etwa SSH (OpenSSH[17] und Putty[18], beide unterstützen über die GSS-API auch Kerberos anstelle von GSI), FTP (GSIFTP[19], GridFTP[20]) sowie CVS (GridCVS[21]).

[17] http://grid.ncsa.uiuc.edu/gssapi-mechglue/openssh/

[18] http://meta.cesnet.cz/cms/opencms/en/docs/environment/tokens/globus/

[19] http://www.globus.org/toolkit/docs/2.4/datagrid/deliverables/gsiftp-tools.html

[20] http://www.globus.org/grid_software/data/gridftp.php

5.3.5 Das Grid-Mapfile

Das Grid-Mapfile ist eine lokale Datei, die in Globus-Toolkit basierten Grid-Middlewares eingesetzt wird. Mit diesem Mechanismus werden zwei Funktionen realisiert. Der Eintrag des Distinguished Name (DN) aus dem Subject seines End-Entity-Zertifikates in das Grid-Mapfile autorisiert einen Benutzer für den Zugriff auf die entsprechende Grid-Komponente. Darauf aufbauend wird im Grid-Mapfile die Abbildung eines Benutzers auf einen separat oder gemeinschaftlich genutzten lokalen UNIX-Account vorgenommen.

In Middlewares, die über LCAS- bzw. LCMAPS-Komponenten die Autorisierung bzw. das Abbilden auf (Pool-)Accounts vornehmen, werden statt des DN die VOMS-Attribute in das Grid-Mapfile eingetragen.

Die aktuelle Zuordnung von Benutzern auf Pool-Accounts wird in einem Verzeichnis, dem Gridmapdir gespeichert.

5.4 Unterstützung von Virtuellen Organisationen

5.4.1 Virtuelle Organisation (VO)

Eine Virtuelle Organisation (VO) ist ein Zusammenschluss von Benutzern und Ressourcen verschiedener Institutionen zum Zweck der Kollaboration in einem gemeinsamen Projekt. Dabei können sich die teilnehmenden Institutionen jeweils mit Personen und Ressourcen an der VO beteiligen, es ist aber auch möglich, sich nur mit Personen oder auch ausschließlich mit Ressourcen zu beteiligen.

Im Rahmen des von der Europäischen Union geförderten *European Data Grid* Projektes wurde ein Attribute Assertion Service zur rollenbasierten Autorisierung in Virtuellen Organisationen (VOMS, siehe 5.4.2), sowie zwei Module für die Site-Local Autorisierung (LCAS, siehe 5.4.3 und LCMAPS, siehe 5.4.4) entwickelt. Diese werden sowohl in LCG als auch in der gLite Middleware eingesetzt.

5.4.2 Virtual Organisation Membership Service (VOMS)

Der *Virtual Organization Membership Service* (VOMS) dient der Verwaltung der Mitglieder einer Virtuellen Organisation (VO), bzw. von Information über den Status eines Benutzers. Die Eigenschaften des Benutzers in einer VO werden über die Zugehörigkeit zu Gruppen sowie der Zuordnung von Rollen und Capabilities definiert. Diese Informationen werden in einem Pseudo-Zertifikat abgelegt, das im Gegensatz zu einem X.509 Zertifikat keine Schlüssel enthält, aber vom VOMS-Server mit dem privaten Schlüssel seines X.509-Host-Zertifikats signiert wird. Das Pseudo-Zertifikat integriert der Benutzer als nichtkritische, private Zertifikatserweiterung in sein Proxy-Zertifikat.

Ein Proxy-Zertifikat kann mehrere Attribut-Zertifikate von verschiedenen Virtuellen Organisationen enthalten, d.h. der Benutzer kann sich an verschiedenen VOs anmelden und somit mehrere VO-Attribute in einem Proxy-Zertifikat nutzen. Auch ist es dem Benutzer möglich, Untermengen seiner Attribute zu definieren, die in einem Proxy-Zertifikat abgelegt werden sollen. Dies kann z.B. dann sinnvoll sein, wenn er dem zu erstellenden Proxy-Zertifikat nicht alle Rechte übertragen möchte, über die er verfügt.

Ein VOMS-Server bietet zwei Zugriffsmöglichkeiten. Eine GSI-Verbindung wird von den CLI-Befehlen verwendet, über die der User Proxy-Zertifikate generiert. Die Administration eines VOMS-Servers erfolgt über eine Web-Oberfläche per HTTPS. Über diese kann vom Benutzer die Mitgliedschaft in einer VO beantragt werden, bzw. wird sie vom VO-Administrator verwendet, um Rollen und Capabilities zu definieren.

---

[21] http://www.dutchgrid.nl/Admin/GridCVS/

5.4.3 Local Center Authorization Service (LCAS)

Der Name des Moduls LCAS steht für *Local Centre Authorization Service* und implementiert einen *Policy Decision Point* (PDP, siehe Kapitel 2.2.5), der eingesetzt wird, um die Autorisierung von eingehenden Job-Requests zu überprüfen. LCAS ist modular aufgebaut und unterstützt die Autorisierungsentscheidung anhand einer Vielzahl von Attributen. Die verfügbaren PDP-Plug-Ins entscheiden anhand folgender Kriterien:

- Attribute, die von einem VOMS-Server erstellt werden. Diese befinden sich in X.509 Certificate Extensions des Proxy-Zertifikats und beinhalten den Namen der VO. Optional können die Mitgliedschaft in Gruppen einer VO, die zugewiesenen Rollen sowie Capabilities enthalten sein. Die Autorisierungsentscheidung kann feingranular durch Überprüfung aller Attribute durchgeführt werden.
- Ein weiteres Modul unterstützt die Autorisierungsentscheidung mittels des Distinguished Name (DN) im Subject des Proxy-Zertifikats. Es ist möglich, sowohl eine Liste mit unerlaubten Benutzern (*banned-userlist)* als auch mit einer Whitelist (*allowed-userlist*) zu führen.
- Über das *wall-time limiting* Modul kann festgelegt werden, wann die entsprechende Ressource für Grid-Benutzer zur Verfügung steht. Es ist damit z.B. möglich, nur zu Zeiten außerhalb des normalen Betriebes einen Dienst für das Grid zur Verfügung zu stellen.

LCAS kann mittels eigener PDP-Plug-Ins flexibel erweitert werden. Zugriffsentscheidungen werden über eine logische „Und“-Verknüpfung der Ergebnisse der einzelnen PDP-Plug-Ins gefällt.

5.4.4 Local Credential Mapping Service (LCMAPS)

LCMAPS steht für *Local Credential Mapping Service*. Dieser Dienst stellt Funktionalitäten zur Verfügung, um User-Credentials flexibel auf lokale Accounts und Gruppen abbilden zu können. Dabei wird eine Reihe von Mechanismen unterstützt:

- Mapping vom Distinguished Name (DN) im Subject des User-Credentials auf einen UNIX-Account basierend auf dem Grid-Mapfile.
- Abbildung des DN auf einen Pool-Account und zugehöriger UNIX-Gruppe (Gridmapdir).
- Durch die Unterstützung von VOMS-Attributen kann eine Abbildung von VO, Gruppen, Rollen und Capabilities auf UNIX-Gruppen vorgenommen werden.
- Eine Unterstützung von Kerberos und AFS wird über die Abbildung von DN auf Kerberos- und AFS-Token realisiert.

LCMAPS ist ebenso wie LCAS über Plug-Ins erweiterbar. In LCMAPS wird zwischen zwei Sorten von Plug-Ins unterschieden, dem *Acquisition*- und *Enforcement*-Typus. Der *Acquisition*-Typus sammelt Informationen über die User-Credentials, während der *Enforcement*-Typus Entscheidungen fällt und durchsetzt.

5.5 GridFTP

Für die Datenübertragung zwischen verschiedenen Systemen im Grid wurde im Globus Toolkit 2 mit GridFTP ein leistungsfähiges, sicheres und robustes Protokoll entwickelt. GridFTP basiert auf dem FTP-Protokoll und fügt diesem für Grid-Umgebungen notwendige Erweiterungen hinzu:

- Third-Party-Transfers ermöglichen es, von einem entfernten GridFTP-Client die direkte Datenübertragung zwischen zwei GridFTP-Servern zu steuern.
- Partial File Access für den Dateitransfer von definierten Segmenten erfolgt durch die Angabe eines Offsets und der Länge des gewünschten Bereiches.
- paralleler Datentransfer über mehrere TCP-Datenströme zur Erhöhung des gesamten Durchsatzes oder zum gleichzeitigen Lesen bzw. Schreiben auf Speicherelemente im Grid.
- Verschlüsselung des Kommunikationskanals

Für die Authentifizierung nutzt GridFTP GSI (siehe 5.3), also X.509 Zertifikate (siehe 0 und 5.2.1).

Die Autorisierung in GridFTP bezieht sich ausschließlich auf den Dateizugriff. Der GridFTP-Server bildet authentifizierte Benutzer mit Hilfe des Grid-Mapfile (s. Kap.5.3.5) auf einen lokalen Benutzer-Account ab und speichert die übertragenen Daten mit den Rechten dieses Accounts. Da nicht gesichert ist, dass ein Benutzer immer wieder auf denselben lokalen Benutzer-Account abgebildet wird, hat auch die UNIX-Gruppe Zugriff auf die Dateien.

Beim Globus Toolkit 4 wurde eine Erweiterung für GridFTP eingeführt, die das Abrufen von Autorisations-Informationen der Grid-Benutzer über einen Community Authorization Service (CAS, siehe Kapitel 6.2.6) ermöglicht.

5.6 GridShib

Infolge der stetigen Weiterentwicklung von virtuellen Organisationen (VO) zu verteilten Kollaborationen zwischen vielen Partnern ist eine sichere Authentifizierung und Autorisierung zu einer immer größeren Herausforderung geworden. Üblicherweise werden Virtuelle Organisationen in existierenden Grid-Infrastrukturen statisch administriert, was aber bei der wachsenden Komplexität immer schwieriger zu verwalten ist und daher als schlecht skalierbar angesehen werden muss..

GridShib ist ein Projekt des National Center for Supercomputer Applications (NCSA) und der University of Chicago [Shi06a][BBF+05]. Es ist das erste Projekt, dessen Ziel es ist, das Globus Toolkit und Shibboleth zu integrieren. Im Ergebnis soll GridShib dazu dienen, Virtuelle Organisationen im Grid durch verteilte Nutzerattribute zu bilden. Verwalter von Ressourcen sollen von der Last befreit werden, Berichtigungen auf der Basis der Identität eines jeden Nutzers in der VO zu verwalten. Die Zugangskontrolle soll anhand der Attribute eines Nutzers entschieden werden, anstatt sie mittels der Identität eines Nutzers zu verwalten. Damit müssen die Verwalter von Ressourcen nicht mehr die Identität ihrer Nutzer kennen, sondern nur ihre Attribute.

GridShib besteht aus zwei Plugins, eines für das Globus Toolkit 4.0 und eines für den Shibboleth IdP:

- GridShib for Globus Toolkit erweitert das GT4. Es hat die Aufgabe, Attribute eines Nutzers von der AA des IdPs abzufragen und anhand dieser Attribute eine Zugangskontrollentscheidung zu treffen.
- GridShib for Shibboleth erweitert einen Shibboleth IdP um eine Namenszuordnung (name mapping). Es dient der Zuordnung eines Zertifikats-DN zu einem lokalen Namen, um zu diesem Namen Attributanfragen eines SP zu beantworten. Diese Namenszuordnungen werden derzeit in einer Textdatei gespeichert. Im GridShib Projekt wurde dies als Skalierungs-Engpass erkannt.

GridShib ist seit September 2005 als Beta-Release verfügbar. Eine Version 1.0 soll mit GT 4.2 veröffentlicht werden; allerdings ist noch kein Datum für eine Veröffentlichung von GT 4.2 bekannt. Das Projekt hat eine Laufzeit bis Q2/2007. Derzeit wird an einer Integration von GridShib und einem „shibbolisierten“ MyProxy gearbeitet.

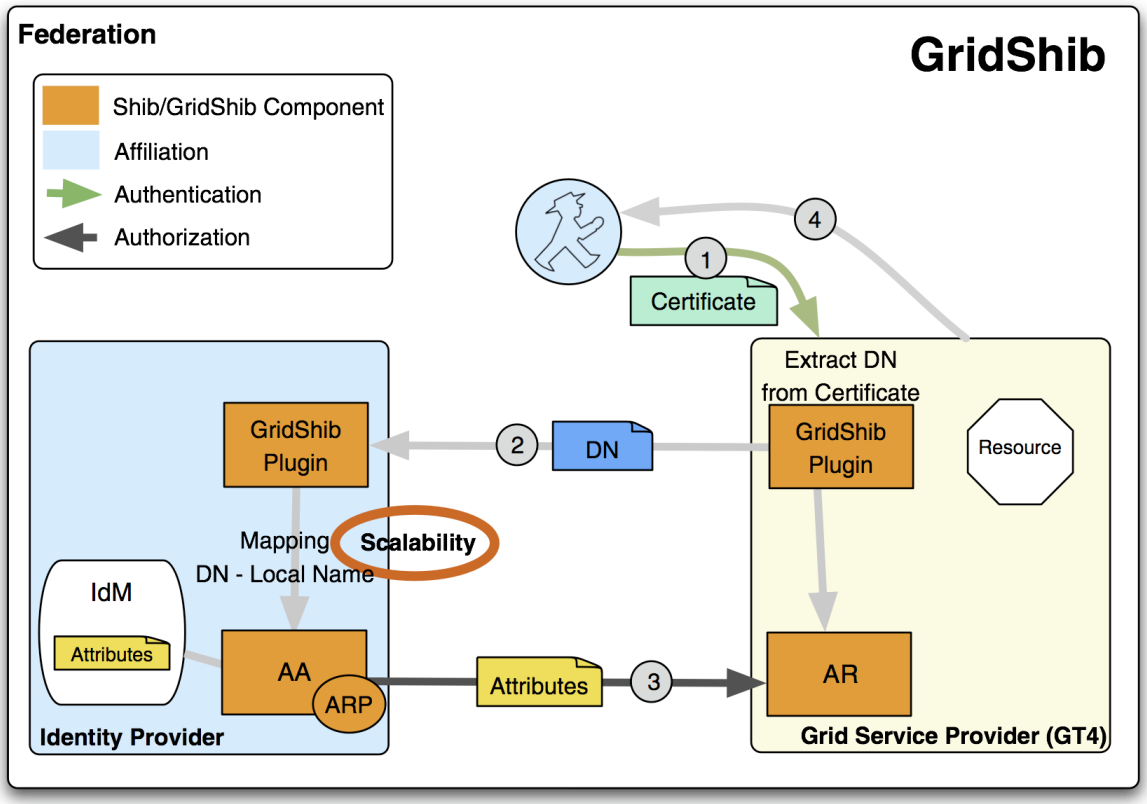

Abbildung 13: GridShib (Pull Modus)

Die Abbildung skizziert den Ablauf einer GridShib-Transaktion, bei der der Grid Service Provider die Attribute vom IdP anfordert (Pull Modus):

1. Ein Nutzer authentifiziert sich mit seinem X.509 Zertifikat an einem Grid Service Provider (Grid SP), um einen Grid-Dienst zu nutzen. Der Grid SP entnimmt dem Zertifikat den Distinguished Name (DN).
2. Der Grid SP authentifiziert sich am IdP des Nutzers und sendet eine SAML-Attributabfrage. Als „Schlüssel“ dient der DN.
3. Die Attribute Authority (AA) des IdP authentifiziert die Attributabfrage, bildet die DN auf einen lokalen Namen ab und sendet unter Vorbehalt der Attribute Release Policy (ARP) eine Zusicherung (Assertion) mit den gewünschten Attributen an den Grid SP zurück.
4. Der Grid SP trifft anhand der Attribute eine Entscheidung über die Gewährung und die Art des Zugangs, bearbeitet im positiven Fall die Anfrage und liefert das Ergebnis an den Nutzer zurück.

Als Einschränkung wird in GridShib derzeit die mangelnde Skalierbarkeit des dateibasierten Name Mapping angesehen. Die Datei soll zukünftig durch eine Datenbank ersetzt werden. Es fehlt zudem ein Mechanismus, um den Identity Provider einer Person automatisch zu ermitteln. Alternativ zum Pull Modus wird ein Push Modus diskutiert: der IdP übergibt dem Nutzer (Client) die Attribute Assertion, die der Nutzer direkt bei der Kommunikation mit einem IdP verwenden kann.

5.6.1 SAML

SAML (Security Assertion Markup Language) ist eine auf XML basierende Beschreibungssprache, mit der auf einfache Art und Weise Informationen zur Authentifizierung und Autorisierung ausgetauscht werden können.

SAML liefert einen standardisierten Weg für die Beschreibung von existierenden Sicherheits-Modellen, ermöglicht den Austausch sicherheitsrelevanter Informationen zur Authentifizierung und Autorisierung und basiert darüber hinaus auf plattform- und herstellerunabhängigen Standards.

Die drei Hauptbestandteile der SAML-Spezifikation sind:

- Assertions: Hier werden Informationen zu Authentifizierung, Autorisierung sowie weitere Session-Attribute hinterlegt.
- Protokoll: Hier wird definiert, wie SAML-Assertions angefordert und übermittelt werden.
- Bindings und Profile: Hier wird festgelegt, wie SAML-Dokumente (Assertions) in Standard-Transport- und Messaging-Frameworks eingebunden werden.

SAML wird in Shibboleth, GridShib und GT4, sowie Liberty [Lib06] genutzt. Im Rahmen der Entwicklung der Open Grid Services Architecture (OGSA) soll SAML ebenfalls zu Autorisierungszwecken eingesetzt werden.

5.6.1.1 SAML Assertions

Kern von SAML sind die so genannten Assertions, also vertrauenswürdige Aussagen von Endanwendern oder Services, die sich jeweils über eine bestimmte digitale Identität definieren. Es gibt drei verschiedene Arten von SAML-Assertions:

- Authentication Assertions: bestätigen, dass bestimmte Benutzer auf geschützte Ressourcen zugreifen dürfen.
- Attribute Assertions: bestätigen, dass einem Benutzer oder einem Web Service bestimmte statische Attribute (Rollen, Funktionen) oder dynamische Attribute (z.B. Kontostand-Informationen) zugeordnet sind. Attributinformationen spielen bei der Zuweisung von Zugangsberechtigungen eine wichtige Rolle.
- Authorisation Decision Assertions: stellen fest, ob und wie auf eine spezifische Ressource zugegriffen werden darf.

SAML Assertions können digital signiert werden. Assertions werden von so genannten Autoritäten ausgegeben, die zur Veröffentlichung von Bestätigungen bevollmächtigt sind (Assertion Issuing Authorities). In der Praxis können alle drei Arten von Assertions von einer Autorität erzeugt werden.

Sämtliche SAML Assertions beinhalten die folgenden allgemeinen Informationen:

- die ID des Ausstellers und das Ausgabedatum
- die Assertion-ID
- ein Subjekt (beispielsweise ein öffentlicher Schlüssel)
- Bedingungen, unter denen eine Assertion gültig ist
- Angaben, wie die Assertion erstellt wurde

Die SAML Attribute Statements und die SAML Authorisation Decision Statements sind für verteilte Transaktionen und Autorisierungsdienste definiert. Ein SAML Attribute Statement bescheinigt, dass einem Subjekt S verschiedene Attribute (A, B etc.) mit bestimmten Werten a, b etc. zugeordnet sind. Ein SAML Authorisation Decision Statement wird von einer so genannten Autorisierungs-Instanz ausgegeben, die Zugriffsentscheidungen (Permit, Deny, Indeterminate) anhand der logischen Kombination von Principal-Attributen trifft.

6 Grid Middleware

Die grundlegenden Komponenten und Verfahren zur Realisierung von AA-Infrastrukturen in Grid Middleware wurden in Kapitel 4 erläutert. Darauf aufbauend wird in diesem Kapitel die spezifische Umsetzung in den verschiedenen Grid Middlewares analysiert. Zur Vorbereitung einer einheitlichen AA-Infrastruktur im D-Grid werden dabei sowohl die Komponenten zur Realisierung der jeweiligen AAI erläutert als auch die Komponenten aufgeführt, die im Wesentlichen die Dienste der AAI nutzen.

6.1 Globus Toolkit 2

Die Ursprünge des Globus Toolkits gehen auf das Projekt I-Way zurück, das „Information Wide Area Year“ (siehe [Korn04] für eine ausführliche Darstellung). Dessen Ziel war ein Verbund von Entwicklungs- und Supercomputer-Zentren zur Ausführung verteilter Anwendungen. Die Geburtsstunde des Globus Toolkits liegt im Jahr 1996. Anfänglich federführend waren Universitäten und Forschungsinstitute, mittlerweile unterstützen aber auch Industriepartner wie IBM und Sun die Entwicklung.

Das Globus Toolkit greift auf die Grid Security Infrastructure (GSI) zurück, die Sicherheitsmechanismen für den Zugriff auf verteilte Ressourcen bereitstellt. Mit Globus-Werkzeugen lässt sich eine Infrastruktur aufbauen, in der Anwender ihre Rechenaufträge in einem heterogenen und verteilten Umfeld sicher ausführen können. Dabei unterstützt etwa Single Sign-On die Autorisierung und Authentifizierung der Benutzer, eine PKI erlaubt die zuverlässige Authentifizierung der beteiligten Parteien und Systeme. Auf Version 2 (GT2) basieren die meisten namhaften Grid-Projekte, z.B. das European Data Grid oder das LHC Computing Grid.

Momentan hat das GT2 noch eine beachtenswerte Verbreitung. Langfristig ist aber die Ablösung durch die Version 4 zu erwarten, verbunden mit dem Schritt hin zu Web-Services (WS). In diesem Zusammenhang unterscheidet man im Globus Toolkitzwischen WS- und pre-WS-Komponenten.

6.1.1 Proxy-Zertifikate in GT2

Das Globus Toolkit 2 arbeitet mit Proxy-Zertifikaten. Es werden jedoch teilweise nicht zu RFC 3820-konforme Proxy-Zertifikate vom Globus Toolkit erzeugt und benutzt [Wel05]. Diese haben „proxy“ oder „limited proxy“ als Distinguished Name-Komponente (und keine PCI-Erweiterung, etwa in GT2) bzw. verwenden eine andere OID (1.3.6.1.4.1.3536.1.222) als die im RFC vorgegebene[22]. Das Globus Toolkit nutzt eine modifizierte Version von OpenSSL, mit der sich PZe ausstellen und validieren lassen[23]. Erst seit Globus Toolkit Version 3 folgen die PZe dem RFC 3820 [Wel04]. Die unterschiedlichen Typen dürfen in einer Zertifikatskette nicht gemischt werden, d.h. ein GT2 PZ kann nur weitere GT2 PZe ausstellen während umgekehrt mit einem RFC-konformen PZ keine GT2 PZe signiert werden dürfen.

6.1.2 Globus Resource Allocation Manager (GRAM)

Das Absetzen von Jobs geschieht in Globus durch den *Globus Resource Allocation Manager (GRAM)*. Die schematische Funktionsweise von GRAM ist in Abbildung 14 dargestellt. Auf der Client-Seite dient die Komponente `grid-proxy-init` dazu, ein temporäres, kurzlebiges Schlüsselpaar und das zugehörige Proxy-Zertifikat zu erzeugen. Unter Verwendung des Proxy-Zertifikats übermittelt die Komponente `gram-submit` bzw. das `globusrun` Kommando einen Job-Auftrag zum Server wo der Auftrag vom `Gatekeeper`-Daemon entgegengenommen wird. Der `Gatekeeper` konsultiert ein serverseitig vorhandenes Grid-Mapfile um festzustellen, ob dort ein Eintrag vorhanden ist, um den im

[22] Die Vorgabe-Einstellung in GT4 erzeugt X.509-Proxy-Zertifikate die nicht RFC-konform sind. Durch die Option `-rfc` läst sich jedoch die Ausstellung RFC-konformer PZ erzwingen.

[23] http://www.openssl.org/docs/HOWTO/proxy_certificates.txt

Proxy-Zertifikat enthaltenen Distinguished Name des Zertifikatsinhabers auf einen lokalen System-Benutzer abzubilden (s. Abschnitt 5.2.1).

Der vom Client übermittelte Job kann im Erfolgsfall mit den Zugriffsberechtigungen des angegebenen System-Benutzers ausgeführt werden. Der Client hat also einen zu diesem System-Benutzer gleichwertigen Zugang zu den Rechenressourcen. Die in der Abbildung 14 dargestellten Prozesse haben entsprechende Ausführungs- und Zugriffsrechte auf dem System. Zum Starten und Verwalten von Jobs auf dem Server dient die vom `Gatekeeper` erzeugte Komponente `job-manager`. Statusinformationen über einen gestarteten Job werden von den Komponenten `callback handler` und `reporter` behandelt.

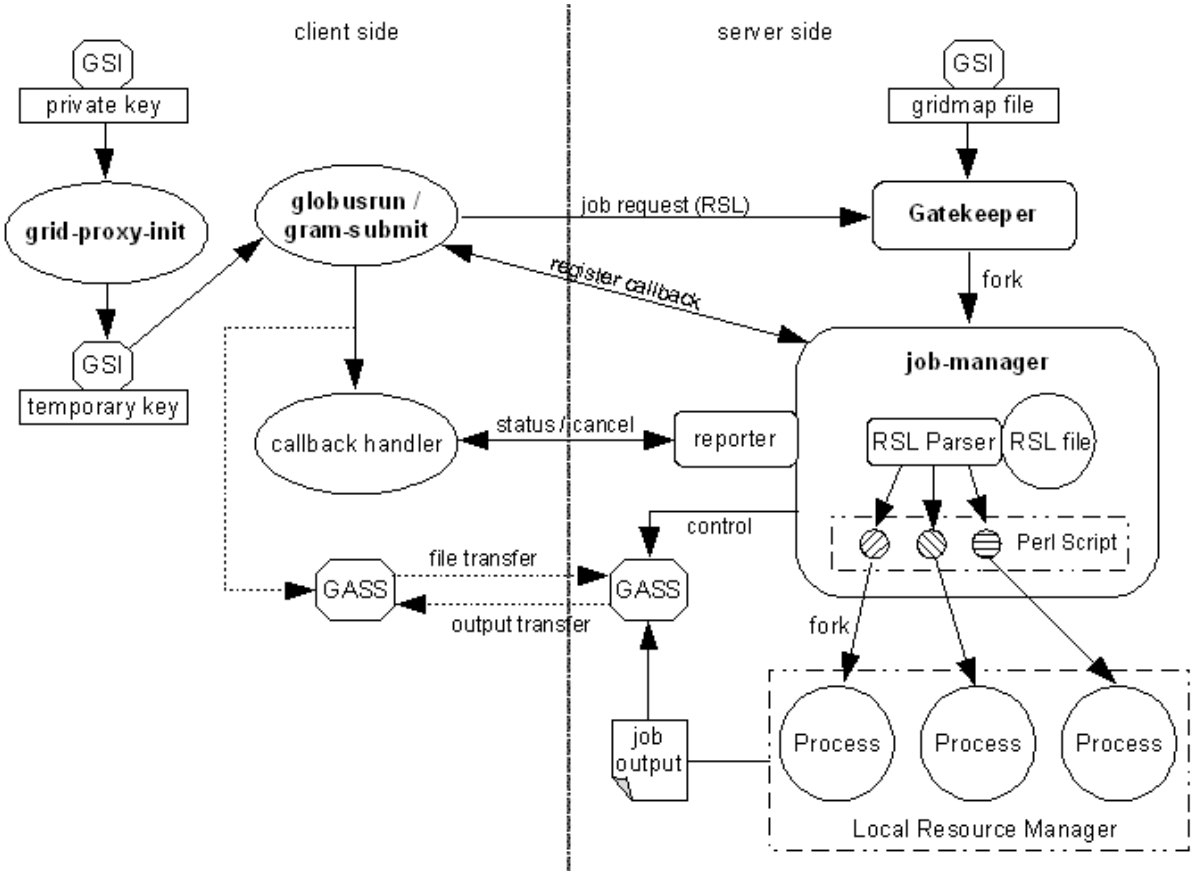

Abbildung 14: Funktionsweise von GRAM

Folgenden Komponenten sind im Zusammenhang mit GRAM relevant:

- Das **`globusrun` Kommando** bzw. die entsprechende API dient zum Absetzen von Jobs, die Grid-Ressourcen nutzen.
- Die **Resource Specification Language (RSL)** ist die Beschreibungssprache, mit der Job-Aufträge formuliert werden. Der Client beschreibt mit ihr Jobs, die vom Server ausgeführt werden sollen. RSL-Strings enthalten typischerweise die Kommandozeilen-Befehle und dazugehörige Parameter sowie Anweisungen, ob und ggf. wohin Standardausgaben umgeleitet werden sollen.
- Der **Gatekeeper-Daemon** mit der oben beschriebenen Funktion.
- Der **Job-manager** mit der im Prinzip oben beschriebenen Funktion (etwa neue Prozesse erzeugen, um Jobs zu starten). Dementsprechend gehört das Interpretieren der vom Client gesendeten RSL Strings auch zu seinen Aufgaben. Unter Verwendung von GASS (siehe unten) kann der Job-Manager Ausgaben der Jobs an den Client zurücksenden, falls der Client dies gefordert hat.

- **Dynamically-Updated Request Online Coallocator (DUROC)** ermöglicht es Benutzern, mehrere Jobs gleichzeitig an mehrere GRAMs mit einem Kommando zu senden. DUROC kümmert sich dabei um die Ausführung und das Verwalten der (auf mehrere Ressourcen) laufenden Jobs.

Optional: **Global Access to Secondary Storage (GASS)** dient zum Übertragen von Ausgaben bzw. Ausgabedateien vom Server (wo sie erzeugt wurden) zum Client.

Authentifizierung:

Der Gatekeeper ist diejenige Schnittstelle, die einen Benutzer/Client authentifiziert und die Jobs annimmt. Zur Authentifizierung wird GSI (siehe Kapitel 5.3) benutzt. Der Mechanismus stützt sich im Wesentlichen auf SSL/TSL (siehe Kapitel 5.1).

Autorisierung

Für die Ausführung eines Jobs ist eine lokale Zugangsberechtigung eines Benutzers erforderlich. Dazu wird auf Mechanismen der GSI zurückgegriffen: Auf Basis des Distinguished Names aus einem Proxy-Zertifikat und dem serverseitig vorhandenen Grid-Mapfile entscheidet der Gatekeeper ob der Client einen zu dem angegebenen System-Benutzer gleichwertigen Zugang zu den Rechenressourcen des Servers erhält.

6.1.3 Zusammenfassung

Wie in Abbildung 14 dargestellt bietet GT2 elementare Möglichkeiten für Authentifizierung und Autorisierung unter Verwendung der pre-WS-Konzepte Proxy-Zertifikate (hier als GridProxy bezeichnet) und Grid-Mapfile.

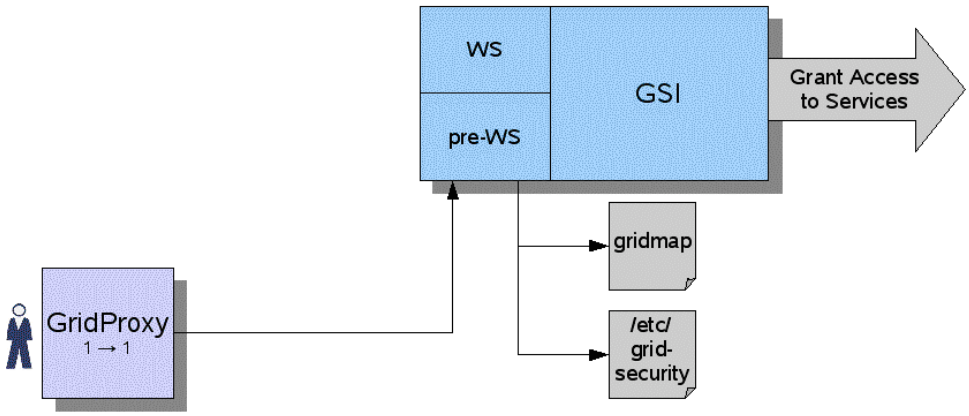

Abbildung 15: GT2 AAI

6.2 Globus Toolkit 4

Die frei verfügbare Software *Globus Toolkit* in der Version 4 (GT4) ist die konsequente Weiterentwicklung früherer Versionen. Der wesentliche Unterschied zu diesen früheren Versionen ist die durchgängige Unterstützung von Web Services als grundlegende Technologie. Dabei bleibt GT4 aber zu großen Teilen abwärts kompatibel.

Das GT4 enthält Software-Bestandteile für Sicherheit, Ressourcen-Management, Daten-Management, Kommunikation, Fehlererkennung und Portabilität. Die einzelnen Bestandteile wie Dienste, Schnittstellen und Protokolle können separat oder im Gesamtpaket verwendet werden, um darauf aufbauend Applikationen zu entwickeln und bereitzustellen.

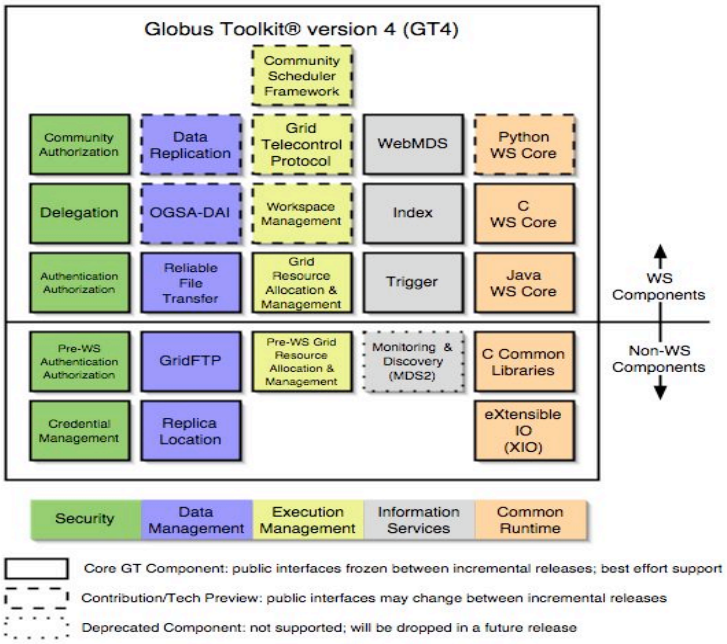

Abbildung 16 Überblick GT4 (Quelle: www.globus.org)

Abbildung 16 gibt einen Überblick über die in GT4 vorhandenen Komponenten. Die für die Sicherheit relevanten Komponenten sind grün eingefärbt (linke Spalte) und werden nachfolgend näher betrachtet. Prinzipiell teilt sich die Architektur von GT4 ein in Komponenten, die nicht mittels Web Services implementiert sind und in dieser Form auch schon in Vorgängerversionen vorhanden waren (non-WS bzw. pre-WS components) und in Komponenten, die nun als Web Services realisiert und auch auf diese zugeschnitten sind (WS components).

6.2.1 Management der Berechtigungsnachweise

Die Identität eines Benutzers wird im GT4 üblicherweise mittels eines gültigen und vom Ressourcenbetreiber anerkannten X.509-Zertifikats bewiesen. Als Alternative bietet GT4 für Web Services-Komponenten [Wel05] auch eine Authentifizierung über Benutzername und Passwort, z.B. bei der Verwendung von WS-Security oder JAAS[24]. Für das Management der Zertifikate eines Benutzers wird bei GT4 das MyProxy-Modul mitgeliefert. Für eine ganze Community kann der enthaltene Community Authorization Server (CAS) verwendet werden. Und die Anbindung weiterer Komponenten ist möglich. Prinzipiell muss aber bei der Authentifizierung und Autorisierung zwischen den pre-WS- und den WS-Komponenten unterschieden werden, da wichtige Erweiterungen nur für Web Services verwendet werden können.

6.2.2 MyProxy

MyProxy dient als Online-Depot für X.509-Zertifikate, das mittels Passwort oder anderen Mechanismen geschützt wird. Die gespeicherten Zertifikate können über das Netzwerk abgerufen und verwendet werden. So müssen der private Schlüssel und die Zertifikate nicht mehr auf die jeweilige vom Benutzer verwendete Maschine kopiert werden. Unterstützt werden Endbenutzer-Zertifikate und Proxy-Zertifikate. Eine genauere Beschreibung von MyProxy ist in Kapitel 5.2.3 zu finden.

[24] Der *Java Authentication and Authorization Service* (http://java.sun.com/products/jaas/) ist eine Java-basierte Implementierung des *Pluggable Authentication Module* (PAM) Frameworks, über das sich (lokale) Authentifizierungs- und Autorisierungstechnologien über Module anbinden lassen.

6.2.3 Authentifizierung und Autorisierung (pre-WS)

Die für Authentifizierung und Autorisierung zuständige Komponente in GT4 bietet Werkzeuge und Schnittstellen an, mit denen Authentifizierung durchgeführt, Autorisierung angewandt und Zertifikate verwaltet werden können. Dabei basiert die Authentifizierung auf *PKI-Technologie* (siehe Kapitel 3) und nutzt *Transport Layer Security* (TLS, siehe Kapitel 5.1). Delegation wird über X.509-Proxy-Zertifikate ermöglicht (siehe Kapitel 5.2.1).

Eine Autorisierung authentifizierter Benutzer ist auf zwei unterschiedliche Arten möglich:

- Durch die Nutzung einer *Zugriffskontrollliste* (z.B. Grid-Mapfile) ist es möglich, die entfernt arbeitenden Benutzer eines GT4-Knotens auf lokale System-Benutzer und deren Autorisierung abzubilden.
- Zweitens kann nach erfolgter Authentifizierung durch einen *Aufruf einer Entscheidungsinstanz* (callout) eine Zugriffskontrolle basierend auf der Identität eines Nutzers durchgeführt werden.

6.2.4 Authentifizierung und Autorisierung (WS)

Eine Authentifizierung in GT4 kann entweder über X.509-Zertifikate (siehe Kapitel 0) oder über Benutzername/Password-Kombinationen (siehe Kapitel 2.1.1.1) vorgenommen werden. Die Verwendung einer Benutzername/Password-Kombination setzt die Nutzung von WS-Security [ADH+02] und damit Message Level Security (siehe Kapitel 5.1.2) voraus. Für Authentifizierung und Autorisierung wird dann über JAAS eine lokal vorhandene Technologie als Entscheidungsinstanz angesprochen. In diesem Falle ist keine Delegation von Rechten möglich. TLS kann zusätzlich anonym verwendet werden; die Identität des Nutzers steckt dann in der Nachricht selbst. In der Standardkonfiguration von GT4 ist Message Level Security aus Performanzgründen nicht aktiviert, es wird alleine TLS verwendet.

Für die Autorisierung in GT4 ist das eingebaute *Authorization Framework* zuständig [BBF+05]. Es bietet umfassende Möglichkeiten zur Autorisierung für alle Ressourcen, Dienste, Container und Clients. Für alle im Web Services Container (hier: Tomcat der Apache Foundation) ablaufenden Einheiten (i.d.R. Web Services) lassen sich Ketten von Autorisierungsmodulen (PIP und PDP, siehe Kapitel 2.2.5 und Abbildung 17) hintereinander schalten, um eine Entscheidung für die Autorisierung herbeizuführen. Diese Module können frei gewählt werden, neue Module können jederzeit implementiert werden. Dazu wird von GT4 eine wohldefinierte Schnittstelle angeboten. Die einfachste Variante eines Moduls verwendet beispielsweise das Grid-Mapfile (siehe Kapitel 5.3.5), während komplexere Module SAML-Assertions (siehe Kapitel 5.6.1) dazu verwenden, alle notwendigen Daten einzuSAMLn. Die Kette der Module wird über XML-Dateien im GT4 definiert und muss vor dem Starten des Containers eingerichtet sein. Änderungen machen einen Neustart des Containers notwendig.

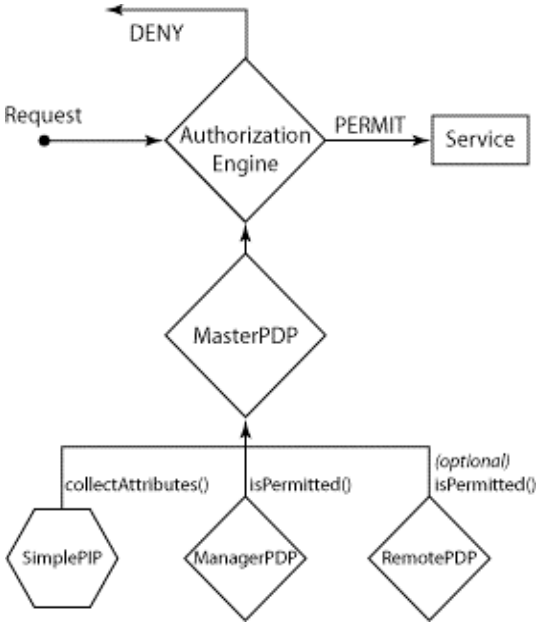

Abbildung 17 Beispiel einer Zusammenschaltung von PIP und PDP in GT4 (Quelle: [FrAn05])

An dieser Stelle lassen sich insbesondere externe Autorisierungsdienste wie PERMIS, VOMS oder Shibboleth anbinden [BBF+05]. Im Fall von Shibboleth stellt GridShib einen solchen PIP/PDP zur Verfügung. In GT4 eingebaute PDP sind:

- self: die Identität eines Clients muss mit der des Ressourcen-Besitzers übereinstimmen
- gridmap: eine Abbildung auf lokale System-Benutzer muss existieren
- identity: die Identität des Benutzers muss exakt übereinstimmen
- host: nur Clients, deren Hostname mit dem angegebenen übereinstimmt
- samlCallout: es wird ein mit OGSA-AuthZ kompatibler Dienst aufgerufen
- userName: ein Benutzer muss sich mit JAAS einloggen

6.2.5 Delegation

Unter Delegation wird in GT4 die Übertragung von Berechtigungen an Dritte verstanden. Zu diesem Zweck enthält GT4 einen Delegationsdienst. Durch diesen Dienst können Berechtigungen so geschaltet werden, dass mehrere parallele Instanzen eines weiteren Dienstes innerhalb derselben Ausführungsumgebung (hosting environment) darauf zurückgreifen können (z.B. mehrere parallele GRAM Job-Ausführungen). Auch die Erneuerung dieser Berechtigungen wird unterstützt.

Der Delegationsdienst akzeptiert Berechtigungsnachweise eines Benutzers und gewährt autorisierten Diensten, die im gleichen Container ablaufen, einen Zugriff auf diese. Der Benutzer erhält im Gegenzug eine Referenz auf die hinterlegten Berechtigungsnachweise und kann diese anderen Diensten übermitteln oder für eine Erneuerung nutzen. Wird eine Berechtigung erneuert, so erhalten alle interessierten Dienste eine entsprechende Benachrichtigung.

Die technische Realisierung basiert auf X.509-Proxy-Zertifikaten und WS-Trust.

6.2.6 Autorisierung einer Community

Der Community Authorization Service (CAS) [CCO+04] erlaubt es virtuellen Organisationen, Richtlinien (policies) für die Nutzung von Ressourcen auszudrücken, die über verschiedene Orte verteilt sind. Üblicherweise wird pro VO ein CAS-Server betrieben. Ein CAS-Server fügt SAML-Assertions in die Berechtigungsnachweise von VO-Nutzern ein und erlaubt so einen feingranularen Zugriff auf Ressourcen. CAS ist erweiterbar und für beliebige Dienste anwendbar; im aktuellen GT4 wird er nur vom GridFTP-Server (siehe Kapitel 5.5) unterstützt, um Zugriffsrechte auf Dateien und Verzeichnisse zu steuern (die zugehörige Attribute sind dabei fest kodiert und nur mit Aufwand änderbar).

Der allgemeine Ablauf ist wie folgt:

- Ein Verantwortlicher für eine VO betreibt einen CAS-Server und benutzt ein Zertifikat, welches den Dienst gegenüber GT4 ausweist.
- Ein Ressourcen-Anbieter gibt der VO Zugriff auf die gewünschten Ressourcen mittels geeigneter Sicherheitsmechanismen wie beispielsweise dem Grid-Mapfile.
- Die Verantwortlichen der VO verwalten die VO-Mitglieder im CAS und steuern die Zugriffsrechte der Benutzer über Policy-Einträge im CAS.
- Möchte ein Benutzer auf eine Ressource zugreifen, so kontaktiert er den CAS und erhält bei ausreichender Berechtigung ein Proxy-Zertifikat mit eingebetteten Rechten.
- Der Benutzer verwendet das ausgegebene Zertifikat, um die GT4-Ressource anzusprechen (z.B. mittels GridFTP). Die Ressource verwendet die Schnittmenge aus lokalen Sicherheitseinstellungen und der vom CAS-Server mitgelieferten Rechte, um den Zugriff des Nutzers auf die Ressource zu steuern.

6.2.7 Zusammenfassung

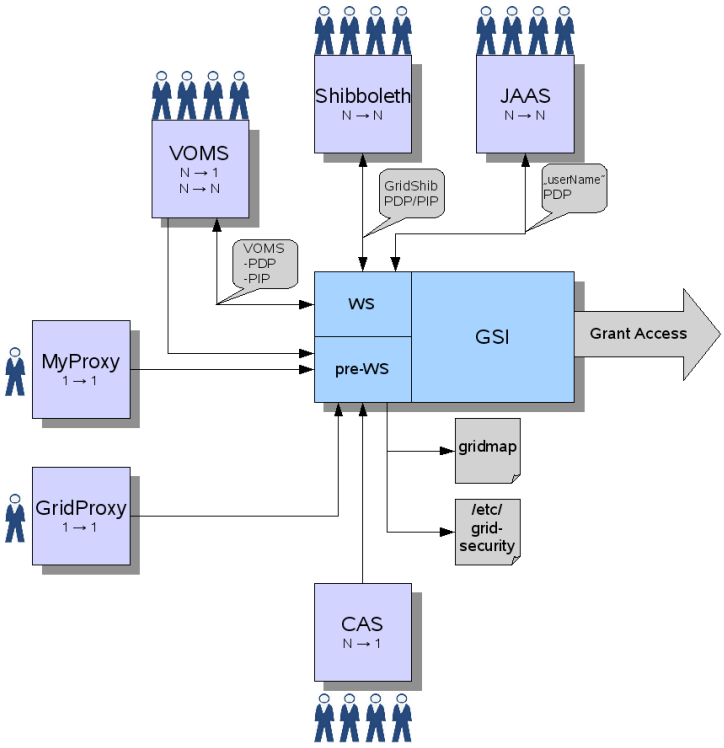

Abbildung 18 Authentifizierung und Autorisierung in GT4 im Überblick

GT4 bietet mannigfaltige Möglichkeiten für Authentifizierung und Autorisierung. Prinzipiell unterschieden werden muss zwischen pre-WS- und WS-Komponenten, die parallel betrieben und genutzt werden. Jede in GT4 angebotene Ressource, jeder Dienst und jeder Dienstcontainer kann

dabei separat konfiguriert werden. Die einen sind kompatibel mit früheren Versionen des Globus Toolkits, die anderen eröffnen neue Möglichkeiten durch das flexible Authorization Framework. Grid-Mapfile, MyProxy/GridProxy und VOMS verwenden pre-WS-Mechanismen für Authentifizierung und Autorisierung, während VOMS-PDP/PIP und GridShib im WS-Bereich angesiedelt sind und mittels SAML-Assertions und Callouts eine feingranulare Möglichkeit zur Autorisierung bieten.

6.3 LCG 2.6

Die Ursprünge der LCG-Software liegen im European Data Grid Projekt (EDG 2001-2004). Entwickelt wurde es auf Basis von Globus Toolkit Version 2 (GT2). Seit 2004 wird diese Middleware im Rahmen des EGEE-Projekts weiterentwickelt. Aktuell ist derzeit LCG Version 2.6, Version 2.7 ist in Entwicklung.

LCG wird im EGEE–Projekt von über 70 Partnern in mehr als 30 Ländern genutzt. Sie ist an über 150 Sites mit insgesamt mehr als 20000 CPUs installiert und wird in vielen Wissenschaftsdiziplinen verwendet, beispielsweise den Geowissenschaften, der Hochenergiephysik, der Bioinformatik sowie der Astrophysik.

Die Middleware besteht aus folgenden Komponenten:

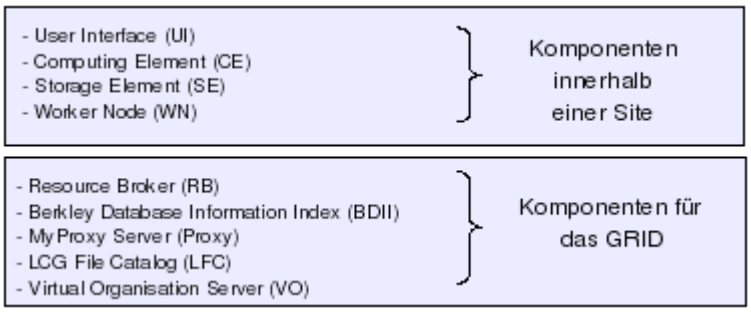

Abbildung 19: Komponenten von LCG

6.3.1 Die Mechanismen der Authentifizierung und Autorisierung in LCG

Abbildung 20 zeigt das Zusammenwirken der einzelnen Komponenten von LCG bezüglich der Authentifizierung und Autorisierung. Beginnend mit dem EEZ-Request des Benutzers an die CA, gefolgt von dem Eintritt in eine VO. Die Erstellung des PZs auf dem UI (für die Jobsubmission) unter Einbeziehung eines VOMS-Servers und die Ablage auf dem MyProxy-Server, sowie dem Vorgang der automatischen Verlängerung des PZs.

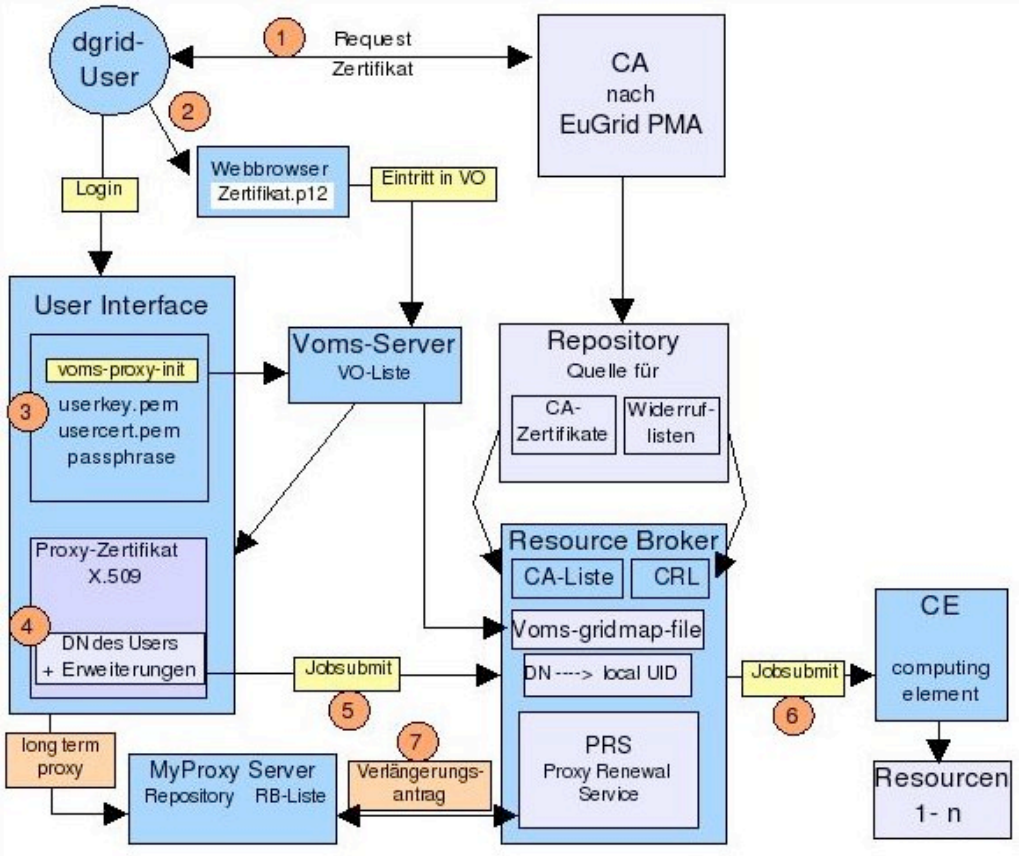

Abbildung 20: Übersicht Authentifizierung und Autorisierung in LCG 2

6.3.2 Das User Interface (UI)

Das User Interface ist die Komponente, über die der Benutzer Zugriff auf die Ressourcen des Grid erhält. Im UI hat der Benutzer alle Befehle zur Verfügung, die er benötigt um sich über das Grid zu informieren, auf das Grid zuzugreifen, seine abgeschickten Jobs zu überwachen und die Ergebnisse seiner Berechnungen entgegen zu nehmen. Folgende Voraussetzungen muss der Benutzer erfüllt haben:

- persönliches Zertifikat (EEZ) einer Certificate Authority (CA)
- Mitgliedschaft in einer Virtuellen Organisation (VO)

Autorisierung/Authentifizierung

Zunächst meldet sich der Benutzer mit seinem Benutzernamen und seinem Passwort am UI an. Um Grid-Services zu nutzen, führt der Benutzer eine Authentifizierung und Autorisierung in der Form eines Single Sign-On (SSO) durch. Dies geschieht durch die Erzeugung eines X.509-Proxy-Zertifikats.

6.3.3 Der Resource Broker (RB)

Der RB hat die Aufgabe, Ressourcen, die vom Benutzer benötigt werden, aufzufinden und zur Verfügung zu stellen. Der RB wurde auf der Basis des Storage Resorce Brokers (SRB) vom San Diego Supercomputer Center entwickelt und wird von vielen wissenschaftlichen Projekten eingesetzt.

Authentifizierung

Die Authentifizierung auf dem RB wird durchgeführt vom NS (Network Server) mit Funktionen aus der Globus Library.

Der Network Server (NS) bearbeitet eingehende Anfragen vom UI, er authentifiziert den Benutzer, kopiert die Input und Output Sandbox (Eingaben und Ausgaben des Benutzers) von oder zu dem UI. Er registriert die Benutzer, die ein long term proxy benutzen wollen beim Proxy Renewal Service, der

ebenfalls auf dem RB läuft. Der RB leitet die Anfragen bzw. Jobs weiter zum Workload Manager (WM).

Autorisierung

Alle Prozesse auf dem RB laufen unter dem gleichen Benutzer. Eine Ausnahme bildet der Dateitransfer mit GridFTP, das die Übertragungen der Input und Output Sandbox vornimmt. Hierzu wird ein Benutzer aus den Poolaccounts benutzt. GridFTP nimmt diese Zuordnung jedoch selbst anhand von Globus-Funktionen vor.

6.3.4 Das Computing Element (CE)

Das CE verwaltet die angeschlossenen Worker Nodes über ein Batch-System und stellt somit die Schnittstelle zu einem Cluster dar.

Authentifizierung

Auf dem CE läuft ein Gatekeeper, der die Benutzer authentifiziert und die Jobs annimmt. Zur Authentifizierung wird GSI (siehe Kapitel 5.3) benutzt. Der Mechanismus stützt sich im Wesentlichen auf SSL/TSL (siehe Kapitel 5.1).

Autorisierung

Die Autorisierung geschieht auf dem CE über die Mechanismen LCAS/LCMAPS (siehe Kapitel 5.4.3 bzw. 5.4.4) bzw. Grid-Mapfile durchgeführt.

6.3.5 Storage Element (SE)

Das SE stellt einen einheitlichen Zugriff auf große (Massen-)Speichersysteme zur Verfügung und ist der zentrale Speicherplatz für Grid-Dateien. Auf dem SE werden die Quelldaten für die Jobs abgelegt und es kann auch die Ergebnisdaten der Jobs entgegennehmen. Das klassische SE besteht im Wesentlichen nur aus einem GridFTP-Server und einem Speichermedium. Neuere Entwicklungen unterstützen über einen SRM (Storage Ressource Manager) den Zugriff auf Massenspeichersysteme wie DPM (Disk Pool Manager), Castor oder dCache. Mit diesen können auch Bandlaufwerke eingebunden werden.

Authentifizierung/Autorisierung

Die Authentifizierung auf dem SE wird wie beim RB und CE über X.509-Zertifikate durchgeführt.

Die Autorisierung wird ebenfalls wie beim CE über die Mechanismen LCAS/LCMAPS und das GridMapfile durchgeführt.

6.3.6 Worker Node (WN)

Ein WN ist der Rechner, auf dem die Jobs der Benutzer ausgeführt und die eigentlichen Berechnungen durchgeführt werden. Eine Reihe von WNs, die von einem CE gesteuert werden, bezeichnet man als einen einzelnen Cluster.

Autorisierung/Authentifizierung

Für die Ausführung eines Jobs ist bei einem WN keine lokale Zugangsberechtigung eines Benutzers erforderlich. Eine Authentifizierung bzw. Autorisierung des Benutzers muss somit auf einem WN nicht durchgeführt werden. Diese wird im Vorfeld auf dem CE durchgeführt.

6.3.7 LCG-File Catalog (LFC)

Mit dem LCG File Catalog werden die im Grid gespeicherten Dateien und ihre Replikas (Kopien) verwaltet und für Benutzer sowie Ressourcen auffindbar gemacht.

Der LFC stellt u.a. folgende Funktionen zur Verfügung

- einen hierarchischen Namensraum und Operationen auf diesem
- integrierte GSI- Authentifizierung und Autorisierung

- Access Control Listen (Unix Berechtigungen und POSIX ACLs)
- Aufbau von Sessions
- Checksummen für Dateien
- Unterstützung von Oracle- und MySQL-Datenbanken

Authentifizierung/Autorisierung

Die sichere Version des LFC bietet sowohl Kerberos 5 als auch GSI zur Authentifizierung an. Wird GSI verwendet, so wird die VO des Benutzers lokal über das Grid-Mapfile auf ein UID/GID Paar abgebildet, welches dann zur Autorisierung verwendet wird. Jeder Benutzer der VO erhält dabei die gleichen Rechte, d.h. jeder Benutzer kann die Einträge der anderen VO-Mitglieder verändern oder löschen.

Ebenfalls ist die VOMS-Unterstützung implementiert, dabei werden die VOMS-Rollen feingranular auf verschiedene Gruppen-IDs im LFC abgebildet. Wiederum werden die UID/GID Paare zur Autorisierung mittels der Datei-Eigentümer-Rechte verwendet, die im Katalog als System Metadaten auf dem Logical File Name (LFN) gespeichert werden.

6.3.8 Berkeley Database Information Index (BDII)

Der BDII fungiert als Informations-Cache in Verbindung mit MDS2 (Monitoring and Discovery Service). Er dient der Speicherung von Daten über die Art und den Zustand von Ressourcen des Grid. Die Datenbank des BDII wird auch vom RB abgefragt, um geeignete Ressourcen für eingehende Jobs zu finden.

Der Site-BDII sammelt Informationen über Abfragen des GRIS (Grid Resource Information Service), der auf den Komponenten CE und SE läuft. Die Informationen werden in einer LDAP-Datenbank nach dem GLUE-Schema (Grid Laboratory for a Uniform Environment) hinterlegt, die wiederum vom zentralen BDII höherer Ebene eingesammelt werden.

In jedem Grid existiert ein zentraler BDII, der redundant ausgelegt sein kann.

Authentifizierung/Autorisierung

Die LDAP-Anfragen können unauthentifiziert durchgeführt werden.

Künftig wird der BDII durch RGMA (Relational Grid Monitoring Architecture) ersetzt. In LCG 2.6 wird RGMA bereits für das Monitoring verwendet. RGMA verwaltet die Statusinformationen von Ressourcen in einer relationalen Datenbank. RGMA implementiert Sicherheitsmechanismen, die auf Authentifizierung durch X.509-Zertifikate basiert.

6.3.9 Zusammenfassung

Wie Abbildung 21 veranschaulicht, beruhen Authentifizierung und Autorisierung der LCG 2.6 Software im Wesentlichen auf Komponenten der GSI unter Einbeziehung von VO-/VOMS-Komponenten gekoppelt mit dem LCAS/LCMAPS-Mechanismus.

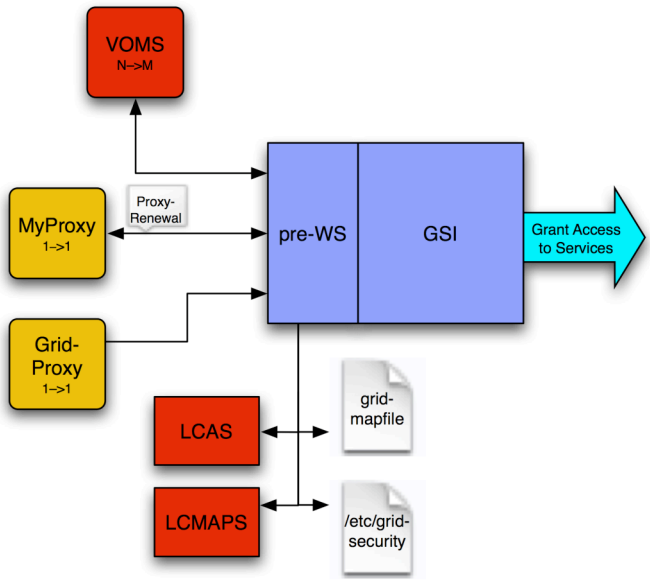

Abbildung 21 LCG2 AAI

6.4 gLite

Die gLite Middleware wird im Rahmen des von der EU geförderten EGEE-Projektes aus bestehenden Grid-Middlewares weiterentwickelt, insbesondere aus LCG2. Eine wichtige Entwicklung hierbei ist in der Verwendung von Web-Service-Technologien zu sehen, die zunehmend in den Komponenten eingesetzt werden.

6.4.1 User Interface (UI)

Das User Interface (UI) stellt die Schnittstelle zwischen dem Benutzer und dem Grid dar. Der Benutzer verfügt über einen lokalen (UNIX-)Account auf dem UI. Über verschiedene Programme kann der Benutzer auf Ressourcen des Grids zugreifen, um Jobs abzugeben oder Daten zu transferieren.

Authentifizierung

Die Authentifizierung zur Nutzung dieses Accounts erfolgt über die herkömmlichen Methoden wie Username/Password (siehe Kapitel 2.1.1.1), Preshared Keys, RSA-Authentication, Kerberos und anderen.

Das Nutzerzertifikat und der zugehörige Schlüssel können lokal gespeichert oder auf einem MyProxy-Server hinterlegt werden. Zur Nutzung der durch das Grid angebotenen Dienste muss sich der Benutzer mittels seiner Credentials ein Proxy-Zertifikat generieren oder vom verwendeten MyProxy-Server abrufen. Dieses Proxy-Zertifikat wird bei der Jobabgabe an den Workload Management Service (WMS) gesendet.

Bei der Erstellung des Proxy-Zertifikates werden Informationen zur VO-Mitgliedschaft in das Proxy-Zertifikat eingebunden, welche vom VOMS-Server (s. 5.4.2) bezogen und vom WMS (s. 6.4.4) ausgewertet werden.

Autorisierung

Berechtigungen des Nutzers auf dem UI werden anhand der Dateirechte für Benutzer, Gruppen und Andere und der Mitgliedschaft zu verschieden Gruppen geregelt.

6.4.2 MyProxy-Server

Der MyProxy-Dienst (s. Abschnitt 5.2) erfüllt in gLite zwei Aufgaben. Zum einen dient er zur sicheren Verwahrung von X.509-Benutzer-Zertifikaten und längerfristig gültigen X.509-Proxy-Zertifikaten, von denen der Benutzer Proxy-Zertifikate ableiten und zur Abgabe von Grid-Jobs verwenden kann. Die

zweite Funktionalität ist die Unterstützung des Proxy-Renewal, also der Erneuerung von Proxy-Zertifikaten durch Grid-Komponenten während der Laufzeit eines Jobs.

6.4.3 Virtual Organization Membership Service (VOMS)

gLite verwendet den Virtual Organization Membership Service (VOMS) wie in Abschnitt 5.4.2 dargestellt. Dabei ist es möglich und üblich in einer gLite basierenden Infrastruktur mehrere VOs bzw. VOMS-Server zu unterstützen. Der VOMS ermöglicht das Betreiben mehrerer VOMS-Instanzen auf einem Server, wodurch es möglich ist, mehrere VOs auf einem System zu verwalten. VOMS unterstützt jedoch keine verteilte Verwaltung einer VO auf mehreren Servern.

6.4.4 Workload Management Service (WMS)

Der Workload Management Service (WMS) ist die Komponente in gLite, die Anforderungen des Nutzers mit den verfügbaren Ressourcen korreliert und anschließend den Job an die ermittelten Komponenten übergibt.

Authentifizierung

Benutzer müssen anhand ihrer X.509-Proxy-Zertifikate von den Komponenten WMProxy oder Network Server des WMS authentifiziert werden, bevor sie ihre Jobs und Daten an den Workload Manager übergeben dürfen.

Um den Abbruch von laufenden Jobs durch Proxy-Zertifikate, deren Gültigkeitsdauer abgelaufen ist, zu verhindern, kann der WMS das ablaufende Proxy-Zertifikat durch ein neues ersetzen. Dieser Vorgang wird als Proxy-Renewal (s. Abbildung 22) bezeichnet. Hierzu authentifiziert sich der WMS mit seinem X.509-Host-Zertifikat gegenüber dem MyProxy-Server und fordert mit dem noch gültigen Proxy-Zertifikat des Nutzers ein neues Proxy-Zertifikat an. Das neue Proxy-Zertifikat wird vom WMS zur Authentifizierung gegenüber dem VOMS-Server genutzt, um die VO-Attribute des ursprünglichen Nutzers zu beziehen.

Die Verbindung zwischen dem WMS und anderen Komponenten der gLite Middleware wird durch die Verwendung von X.509-Host-Zertifikaten authentifiziert und durch die Nutzung von SSL/TLS gesichert.

Autorisierung

Der WMS ist die erste Komponente der gLite Middleware, die Benutzer auf Pool-Accounts abbildet. Das Mapping findet durch das Grid-Mapfile statt. Die Einträge in diesem Grid-Mapfile werden regelmäßig durch Anfragen an den VOMS-Server auf den aktuellen Stand gebracht.

Durch den Prozess des Matchmaking, bei dem Nutzeranforderungen mit vorhandenen Ressourcen korreliert werden, werden zugleich Berechtigungen zur Nutzung einer Ressource anhand der VO-Mitgliedschaft des Nutzers geprüft.

Für den oben beschriebenen Prozess des Proxy-Renewal findet die Autorisierung des Klienten durch eine *Access Control List* (ACL) statt.

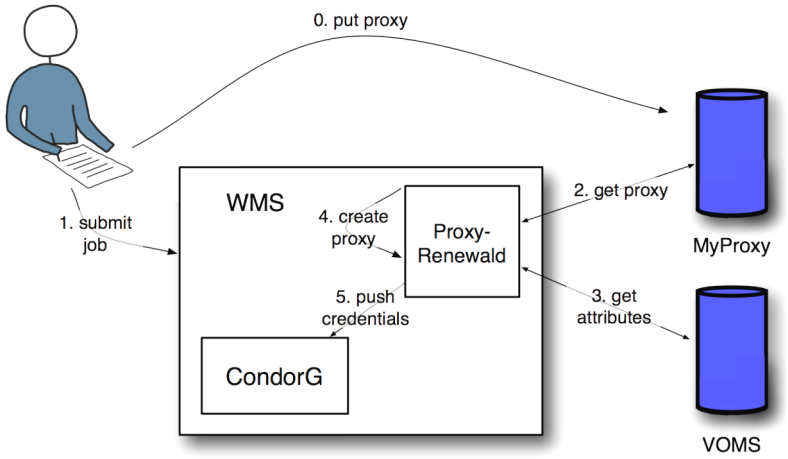

Abbildung 22: Proxy-Renewal in gLite-Konzept

6.4.5 Computing Element (CE)

Das Computing Element ist die Schnittstelle zu den Worker Nodes. Die Jobsubmission vom WMS an das CE geschieht durch das lokale Batchsystem.

Authentifizierung

Die Authentifizierung findet durch die X.509-Proxy-Zertifikate des Nutzers statt.

Autorisierung

Zur Autorisierung wird auf dem CE LCAS und LCMAPS (s. Abschntt 5.4.3 und 5.4.4) eingesetzt.

6.4.6 Catalog / IO-Service

Die gLite Middleware unterscheidet bei den Dateidiensten zwischen Katalog- und IO-Diensten.

Der Katalogdienst ist für das Auffinden von Dateien und die Verwaltung von Replikas zuständig. Replikas sind dabei Dateien, die Kopien einer Ursprungsdatei darstellen und die aus Gründen der besseren Ressourcennutzung an verschiedenen Stellen im Grid vorgehalten werden. Wichtig hierbei ist insbesondere die Sicherung der Konsistenz der verschiedenen Replikas.

Der IO-Service bietet dem Benutzer die Möglichkeit, Dateien von und zum *Storage Element* (SE) zu transferieren.

Authentifizierung

Im Catalog-Service wird die Authentifizierung von Benutzern und anderen Diensten, wie z.B. dem IO-Server über X.509(-Proxy)-Zertifikate realisiert.

Autorisierung

Die Autorisierungskomponente des Catalog-Services verwendet die VOMS-Attribute im Proxy-Zertifikat des zugreifenden Benutzers, sowie klassische *Access Control Lists* (ACL) zur Entscheidung über die Zugriffsrechte auf Dateien und Verzeichnisse. Dabei werden für die Autorisierungsentscheidung der *Distinguished Name* im Subject des Benutzer-Zertifikats oder dessen VOMS-Attribute verwendet.

Im IO-Server erfolgt die Autorisierung über den *File Authorization Service* (FAS) des Catalog-Services.

6.4.7 DataGrid Accounting Service (DGAS)

Das *Distributed Grid Accounting System* (DGAS) entstand im Rahmen des EU Datagrid Projektes als *DataGrid Accounting System* und wird seit Anfang 2004 im Rahmen des EGEE Projektes

weiterentwickelt. DGAS dient dem Erfassen des Ressourcenverbrauches von Jobs, sowie dem Accounting und Billing.

Authentifizierung

Die Authentifizierung in DGAS erfolgt wie bei anderen gLite Komponenten über GSI mittels X.509-Zertifikaten und SSL/TLS. Beim DGAS-Server authentifizieren sich sowohl Benutzer, die Jobs autorisieren wollen, als auch andere Dienste, die Informationen über Benutzerjobs auf dem DGAS-Server ablegen wollen.

Autorisierung

Die Autorisation für einen Job erfolgt durch den Benutzer, um Missbrauch zu verhindern, wie z.B. die Erschleichung von Diensten auf Kosten des Nutzers. In gLite autorisiert der Benutzer Accounting und Payment-Transaktionen bei Abgabe des Jobs an das System. Die Autorisierung erfolgt über die *Distinguished Names* in den Zertifikaten der Benutzer und Hosts.

6.4.8 Logging und Bookkeeping (L&B)

Statusinformationen eines Jobs werden im Logging- und Bookkeeping-Server gespeichert. Dabei wird zwischen kurzlebigen Informationen (Bookkeeping) und langlebigen Informationen (Logging) unterschieden.

Die Bookkeeping-Komponente hält aktuelle Informationen, die den Status eines Jobs betreffen vor, während der Logging-Service die Nachverfolgung eines Jobs auch nach dessen Beendigung ermöglicht.

Authentifizierung

Auf die vom Logging- und Bookkeeping-Server angebotenen Informationen darf der durch ein Proxy-Zertifikat authentifizierte Benutzer zugreifen. Hosts, die Meldungen über den Status eine Jobs an den L&B-Server melden, werden über ihre X.509-Host-Zertifikate authentifiziert.

Autorisierung

Zusätzliche Benutzer der L&B-Informationen können durch weitere Einträge in einer Access Control List bestimmt werden. Diese ACL kann der Jobs abgebende Benutzer durch Event-Meldungen verändern.

Der Workload Management Service und das Computing Element sind berechtigt, Statusinformationen eines Job an den Logging und Bookkeeping-Server zu melden. Die Autorisierung erfolgt über den DN im Subject ihrer X.509 Host-Zertifikate.

6.4.9 Zusammenfassung

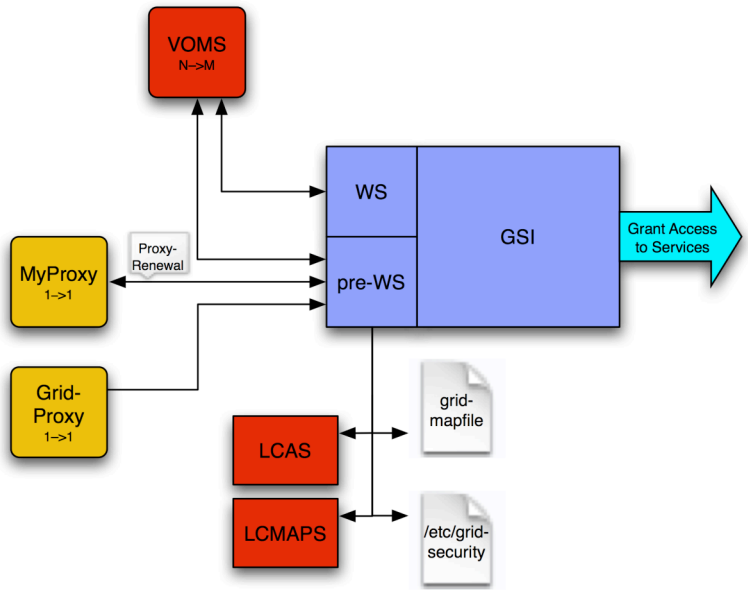

Abbildung 23: gLite AAI

Wie beim Wechsel von Globus-Toolkit Version 2 auf Globus-Toolkit Version 4 ist der Übergang auf Web-Service basierende Dienste auch bei gLite ein Schwerpunkt der Entwicklung. Dieser Paradigmenwechsel ist noch nicht abgeschlossen, so dass weiterhin mehrere Schnittstellen unterstützt werden müssen. Die Nutzung des VOMS-Dienstes als zentralen PDP innerhalb einer VO ist in gLite noch nicht in allen Komponenten vollständig realisiert, so dass auch hier noch weitere Entwicklungen notwendig und zu erwarten sind.

6.5 UNICORE

Mit UNICORE soll der Zugriff auf Rechenressourcen vereinheitlicht werden; bereits der Name – UNICORE ist eine Abkürzung für UNiform Interface to COmputing REsources – weist auf diese Absicht hin.

Das UNICORE-Paket bietet einen sogenannten „vertikal integrierten“ Softwarestack an. Das bedeutet, dass von der Kommunikation auf Netzwerkebene bis hin zur Benutzerschnittstelle alle Ebenen der Kommunikation in einem einzigen Programm zusammengefasst sind. Es ist mithin nicht nötig, mehrere Bibliotheken oder Programme zu installieren, um letztlich mit Hilfe des Benutzerinterfaces einen Rechenjob abschicken zu können, wie dies bei anderen Middlewares der Fall ist. Vielmehr ist es hinreichend, ein einziges Programm – den UNICORE-Client – zu installieren.

Durch diesen monolithischen Ansatz geht natürlich Flexibilität verloren, gleichzeitig wachsen Umfang und Mächtigkeit des Clients. In der Tat bietet UNICORE eine ausgefeilte graphische Benutzeroberfläche, die Rechenprozesse in Form von Graphen darstellen und auf diese Weise Abhängigkeiten von Jobs untereinander verstehen kann. Es existieren viele Typen von Tasks, aus denen der Benutzer per Mausklick, aber wahlweise auch über eine textuelle Eingabe, seinen Rechenjob zusammensetzen kann. Gleichzeitig führt der UNICORE-Client eine Liste von Ressourcen, auf die der Benutzer zugreifen darf; einzelne Tasks können dann wieder per Mausklick den Ressourcen zugewiesen werden. Auf diese Weise kann ein Benutzer leicht festlegen, auf welchen Rechnern seine Jobs tatsächlich gerechnet werden sollen, und die einzelnen Schritte seines Gesamtjobs sogar auf verschiedene Ressourcen verteilen.

6.5.1 Grundlagen

6.5.1.1 Authentifizierung und Autorisierung

Authentifizierung und Autorisierung erfolgen bei UNICORE mit Hilfe von X.509-Zertifikaten.

Es gibt vier Softwareinstanzen, die an Authentifizierung und Autorisierung beteiligt sind. Es handelt sich hierbei um den Client, das Gateway, den Network Job Supervisor (NJS) und die UNICORE User Database (UUDB). Zudem sind für das Verständnis noch die Begriffe UNICORE-Site (Usite) und Virtual Site (Vsite) wichtig.

Client: Beim UNICORE-Client handelt es sich um das auf dem Arbeitsplatzrechner eines UNICORE-Anwenders installierte Softwarepaket.

UNICORE-Site (Usite): Bei einer Usite handelt es sich um eine organisatorische Einheit, nicht um eine konkrete Ressource. Eine Usite enthält eine oder mehrere Virtual Sites und verwaltet diese.

Virtual Site (Vsite): Bei einer Vsite handelt es sich – im Gegensatz zu einer Usite – um eine konkrete Ressource, beispielsweise ein Rechencluster. Eine Vsite ist Teil einer UNICORE-Site.

Gateway: Ein Gateway ist das für eine gegebene UNICORE-Site (Usite) zuständige Portalsystem. Sämtliche Zugriffe von außen, insbesondere von Anwendern, durchlaufen das Gateway. Am Gateway erfolgt die Authentifizierung.

Network Job Supervisor (NJS): Der Network Job Supervisor ist das für eine gegebene Virtual Site (Vsite) zuständige Portalsystem. Hier findet die Autorisierung statt. Der NJS ist außerdem das den eigentlichen Ressourcen am nächsten stehende System, mit dem ein Anwender direkt kommunizieren kann; er nimmt Rechenjobs von Benutzern und anderen NJSs entgegen.

UNICORE User Database (UUDB): Die UNICORE User Database einer Usite hält Informationen über die UNICORE-Anwender, die auf dieser Usite angehörenden Vsites zugelassen sind. In jeder Usite gibt es genau eine UUDB. Sie dient unter anderem der Authentifizierung von Benutzern und NJSs.

Zwischen diesen Instanzen bestehen drei relevante Kommunikationspfade:

- Kommunikation zwischen Client und NJS. Dies tritt dann auf, wenn ein Anwender direkt auf eine Ressource zugreift.
- Kommunikation zwischen zwei NJSs. Diese Konstellation tritt bei so genannten Multi-Site Jobs auf, also bei Jobs, die auf Ressourcen zugreifen, die über mehrere Vsites oder Usites hinweg verteilt sind.
- Kommunikation zwischen NJS und UUDB. Dies kommt immer dann vor, wenn ein NJS die Autorisierung eines Clients oder eines anderen NJSs prüft.

6.5.1.2 Zukünftige Entwicklung von UNICORE

Die aktuelle Entwicklung von UNICORE strebt an, die Schnittstellen zwischen UNICORE-Instanzen als Webservices zu implementieren. Die nächste Entwicklungsstufe von UNICORE heisst daher auch UNICORE/GS: UNICORE/Grid Services. Ziel ist es, die als Grid-Basisdienste identifizierten primitiven Operationen, die mehr oder weniger explizit auch in UNICORE vorhanden sind, in Form von modularen Blöcken zur Verfügung zu stellen, so dass sie beispielsweise als Web Services oder als Java-Bean-API für Benutzer verfügbar gemacht werden können.

Die treibende Idee hinter dieser Entwicklung ist die voranschreitende Standardisierung im Bereich der Protokolle im Grid-Computing-Umfeld. Die Entwicklung von UNICORE/GS scheint gut voranzukommen; die Entwickler betonen allerdings, dass in 2006 keinesfalls mit einer produktionstauglichen Version zu rechnen ist.

6.5.2 Der UNICORE Client

Der UNICORE-Client umfasst sämtliche Ebenen des für UNICORE benötigten Software-Stacks. Sowohl die unteren Schichten – beispielsweise die Kommunikationsmechanismen – als auch die

oberen Schichten – beispielsweise das Benutzerinterface – sind in einem einzigen Programm zusammengefasst. Der Anwender hinterlegt seine X.509-Zertifikate sowie die dazugehörigen passwortgeschützten privaten Schlüssel im UNICORE-Client. Bei jedem Starten des Clients wird er aufgefordert, seine privaten Schlüssel durch Eingabe der Kennwörter zu dechiffrieren. Der Client hat dann direkten Zugang zu den privaten Schlüsseln des Anwenders, solange er ausgeführt wird; auf diese Weise kann sich der Client im Namen des Benutzers authentifizieren, ohne dass dieser jedes mal erneut seine Kennwörter eingeben muss. Daher sind bei UNICORE keinerlei Proxy-Zertifikate vorgesehen; Single Sign-On wird bereits standardmäßig unterstützt.

Weiterhin muss der Benutzer geeignete Zertifikate im UNICORE-Client hinterlegen, die es erlauben, die Gegenstellen, mit denen kommuniziert werden soll – also die Gateways und die NJSs der entsprechenden Usites beziehungsweise Vsites –, zu authentifizieren. Auch die bei Public Key Infrastructures übliche Möglichkeit, Zertifikate durch Certificate Revocation Lists (CRLs) zurückzuziehen, kann durch entsprechende Einstellungen vom UNICORE-Client genutzt werden. Ebenso ist es möglich, nach erfolgter Authentifizierung eine gesicherte SSL-Verbindung zum Kommunikationspartner herzustellen.

6.5.3 Das UNICORE Gateway

Das UNICORE-Gateway ist für die Authentifizierung der Absender eingehender Verbindungen zuständig. Diese Aufgabe wird mit Hilfe gewöhnlicher X.509-Zertifikate bewältigt; kann der Absender einer eingehenden Verbindung ein aus Sicht des Gateways gültiges Zertifikat präsentieren, so gilt er als authentifiziert. Hierbei ist – ebenso wie beim Client – auch der Einsatz von CRLs möglich.

Das Gateway ist – dem Namen entsprechend – die einzige Verbindung zwischen der Außenwelt und einer Usite. Insbesondere muss sämtliche Kommunikation zu den Network Job Supervisors, die als Portale zu Vsites innerhalb der umschließenden Usite angesiedelt sind, das Gateway passieren.

6.5.4 Der UNICORE Network Job Supervisor (NJS)

Der Network Job Supervisor ist einerseits die Gegenstelle einer Vsite, die eingehenden Jobs eines Benutzers von seinem Client entgegennimmt, andererseits aber auch die Stelle im UNICORE-Schema, die die Autorisierung der Anwender übernimmt. Da alle Verbindungen von außen das Gateway der Usite passiert haben müssen, sind aus Sicht des NJS die Gegenstellen jeglicher Außenkommunikation authentifiziert. Geprüft werden muss daher nur noch, ob die gegebene Gegenstelle berechtigt ist, auf das vom NJS verwaltete System zuzugreifen.

Dies geschieht durch eine Anfrage bei der UNICORE User Database. Ist die entsprechende Anfrage positiv, so ist die Gegenstelle aus Sicht des NJS authentifiziert und autorisiert und somit legitimiert, auf die Ressourcen zuzugreifen.

6.5.5 Die UNICORE User Database (UUDB)

Die UNICORE User Database ist die zentrale Stelle einer Usite, an der Daten über die in den verschiedenen Vsites der Usite zugelassenen Benutzer gehalten werden. Insbesondere enthält die UUDB eine Abbildung von Zertifikaten auf lokale Accounts. Die UUDB ordnet nicht nur Distinguished Names oder Common Names, sondern ganze Zertifikate lokalen Accounts zu. Auf diese Weise wird auf der einen Seite ermöglicht, unter ein- und demselben Common Name mit Hilfe mehrerer unterschiedlicher Zertifikate unterschiedliche Zugriffsberechtigungsstufen oder Testaccounts zu vergeben; auf der anderen Seite zieht dies ebenso einen vergleichsweise großen Arbeitsaufwand bezüglich der Pflege der UUDB beziehungsweise der darin enthaltenen Zertifikate nach sich, denn es kann nicht pauschal einem Benutzer aufgrund seines Common Names Zugang gewährt werden. Läuft ein Zertifikat aus, muss es in der UUDB erneuert werden.

Die Zuordnung von Zertifikaten auf lokale Accounts in der UUDB ist überdies keine Eins-zu-Eins-Abbildung. Vielmehr ist es sowohl möglich, mehrere Zertifikate auf einen Account abzubilden, als auch ein Zertifikat mehreren Accounts zuzuordnen.

6.5.6 Zusammenfassung

Authentifikation und Autorisation sind im UNICORE-Umfeld sehr einfach implementiert. Es existiert so gut wie kein Spielraum für verschiedene Strukturen, vielmehr ist der schematische Aufbau der Sicherheitsbeziehungen der verschiedenen Instanzen untereinander immer gleich (siehe Abbildung 24). Ein Client authentifiziert sich stets zunächst beim UNICORE-Gateway. Nach erfolgter Authentifizierung leitet das UNICORE-Gateway Client-Anfragen dann weiter an den entsprechenden Network Job Supervisor, der zum Zwecke der Autorisierung des Clients Informationen von der UNICORE User Database einholt. Nach erfolgter Autorisierung reicht der NJS die Client-Anfrage schließlich an die tatsächlichen Ressourcen weiter.

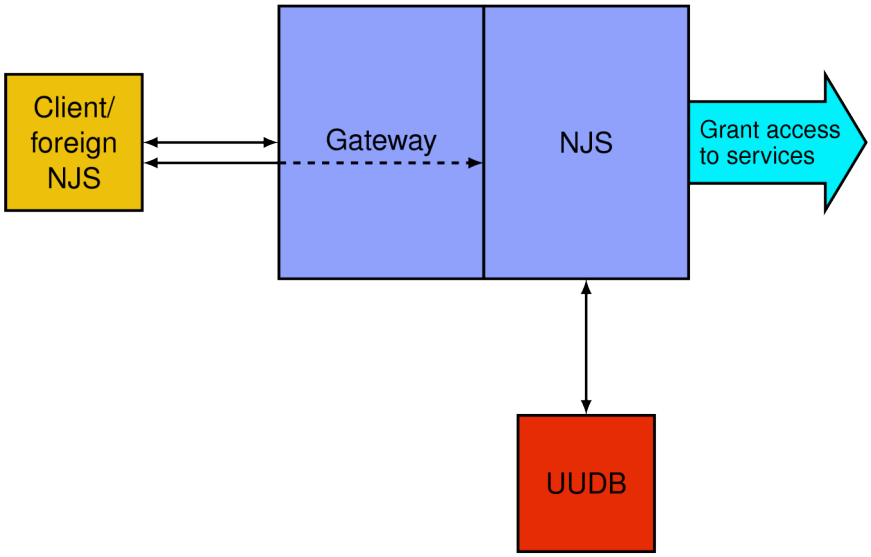

Abbildung 24: UNICORE AAI

7 Zusammenfassung

Die vorgelegte Analyse untersucht die Infrastrukturen für Authentifizierung und Autorisierung in fünf verschiedenen Grid Middlewares. Es zeigt sich, dass die Grid Middlewares Globus Toolkit 2, Globus Toolkit 4, LCG und gLite auf ähnlichen Mechanismen beruhen, was angesichts des gemeinsamen Ursprungs im Globus Toolkit 2 (GT2) zu erwarten ist. Dem gegenüber enthält die Grid Middleware UNICORE, abgesehen von der Nutzung von X.509-Zertifikaten, eigenständige Ansätze zur Realisierung einer AAI.

7.1 Ergebnisse

Für die Authentifizierung von Benutzern und Ressourcen lassen sich in den GT2-basierten Grid Middlewares zwei übergreifende Verfahren identifizieren:

- Nutzung von X.509-Zertifikaten mit zugrunde liegender Public Key Infrastructure,
- Single Sign-On und Delegation von Rechten anhand von Proxy-Zertifikaten.

Beide Verfahren werden in allen GT2-basierten Grid Middlewares durch die Grid Security Infrastucture (GSI) ermöglicht. Zur Wahrung der weltweiten Akzeptanz der im D-Grid verwendeten Zertifikate ist darauf zu achten, dass die ausstellenden CAs nach EUGridPMA-konformen Policies arbeiten.

Eine rudimentäre, nutzerbasierte Autorisierung wird in den GT2-basierten Grid Middlewares über das so genannte Grid-Mapfile realisiert. Die hierfür erforderlichen Funktionen werden ebenfalls in GSI implementiert.

Durch die Einführung von Web-Services mit Globus Toolkit 4 ergeben sich neue Möglichkeiten zur Erweiterung der bisherigen Ansätze. Die Vereinheitlichung der Kommunikation über SOAP und HTTP sowie die Nutzung von SAML bietet standardisierte Schnittstellen und Protokolle, über die neue Dienste zur Authentifizierung und Autorisierung eingebunden werden können.

Die Verwendung von SAML erlaubt eine einheitliche Syntax zur Formulierung von Policies und Assertions, so dass deutlich verbesserte Konzepte zur Autorisierung umgesetzt werden können. Nutzerübergreifende Ansätze wie VO- oder rollenbasierte Modelle werden möglich.

Neben den verfügbaren Mechanismen werden in der vorliegenden Analyse auch die Dienste zur Bereitstellung und Umsetzung von AAIs betrachtet. Zur Verwaltung der Mitglieder in VOs ist in Globus Toolkit 4, LCG sowie gLite die Nutzung von VOMS möglich. Neben der Verwaltung der Accounts werden über VOMS auch die Rollen der Benutzer festgelegt und deren Attribute beglaubigt, d.h. signiert. Zu berücksichtigen ist, dass VOMS als zentrale Instanz innerhalb einer VO bereitgestellt werden muss und nicht verteilt betrieben werden kann.

Als zweiter wesentlicher Dienst, der den genannten Middlewares gemeinsam ist, dient MyProxy dem Erneuern von Proxy-Zertifikaten sowie bei Bedarf dem Ablegen von privaten Schlüsseln. In der letztgenannten Funktion übernimmt MyProxy die Aufgabe eines Credential Wallet. Im Gegensatz zu VOMS ist die Anzahl der MyProxy-Server innerhalb einer VO nicht eingeschränkt.

Neben VOMS ist als Alternative AAI Shibboleth zu berücksichtigen, welches zunehmend im Hochschulbereich zur einheitlichen Verwaltung und Rechtevergabe von Accounts für Mitarbeiter und Studierende eingesetzt wird. Mit der derzeitigen Entwicklung von GridShib wird eine Kopplung von Shibboleth an das Globus Toolkit 4 angestrebt.

7.2 Weiteres Vorgehen

Ziel der vorgelegten Analyse ist die Vorbereitung eines Entwurfs für eine einheitliche AA-Infrastruktur im D-Grid. Aufgrund der oben dargelegten Erkenntnisse und der Anforderungen aus den Communities ist es für die weiteren Betrachtungen möglich, eine Einschränkung auf Komponenten der Middlewares Globus Toolkit 4 und den gemeinsamen Nachfolger von LCG und gLite (vermutlich gLite 3.0) vorzunehmen.

Verglichen mit dem Globus Toolkit 4 ist die Einbindung von Web-Services in LCG bzw. gLite noch nicht weit fortgeschritten. Daraus folgt, dass in einer übergreifenden AAI im D-Grid sowohl VOMS als auch MyProxy über herkömmliche Schnittstellen (pre Web-Services) und über Web-Services zur Verfügung gestellt werden müssen.

Zusätzlich ist UNICORE als eigenständige Lösung gesondert zu betrachten.

Bedingt durch die Verankerung von Communities in internationalen Projekten, die auf vorgegebene Grid Middleware aufsetzen (z.B. HEP in EGEE), ist eine Unterstützung von VOMS im D-Grid unerlässlich. Daneben formieren sich Communities wie C3 und TextGrid ohne vorgegebene Bindung an etablierte AAIs. Diese Communities verfügen über bestehende dezentrale Nutzerverzeichnisse und fordern daher als Alternative zu VOMS eine Unterstützung von Shibboleth bzw. GridShib im D-Grid.

Unabhängig von der Diskussion über VOMS und Shibboleth ist zu bedenken, dass über die vorhandenen Mechanismen hinausgehende Sicherheitsanforderungen an AAIs im D-Grid bestehen. Besondere Ansprüche werden hierbei aus den Communities der Medizin und den Ingenieurwissenschaften gestellt. Hier werden zu Beginn des nächsten Projektabschnitts intensive Gespräche stattfinden, um die Anforderungen und mögliche Lösungen sowie notwendige Entwicklungen zu identifizieren.

8 Verzeichnis der Abkürzungen

A

AA	Attribute Authority
AAA	Authentication, Authorisation, Accounting
AAI	Authentication and Authorisation Infrastructure
ABAC	Attribute-Based Access Control
ACL	Access Control List
ACS	Assertion Consumer Service
AFS	Andrew File System
API	Application Programming Interface
ARP	Attribute Release Policy
AuthZ	Authorization
AuthN	Authentication

B

BDII	Berkeley Database Index

C

CA	Certification Authority
CAS	Community Authorisation Server
CE	Computing Element
CLI	Command-Line Interface
CN	Common Name
CRL	Certificate Revocation List

D

DAC	Discretionary Access Control
DGAS	DATAGrid Accounting System
DLAM	Digital Library Access Control Model
DN	Distinguished Name
DPM	Disk Pool Manager

E

EDG	European Data Grid
EEZ	End-Entity-Zertifikat
EGEE	Enabling Grid for ESciencE

F

FAS	File Authorisation Service
FQAN	Fully-Qualified Attribute Name
FTP	File Transfer Protocol

G

GASS	Global Access to Secondary Storage
GLUE	Grid Laboratory a Uniform Environment
GRAM	Grid Resource Allocation Manager
GSI	Grid Security Infrastructure
GSIFTP	um GSI-Funktionalität erweitertes FTP
GSS-API	Generic Security Service Application Programming Interface
GT	Globus Toolkit

H

http	Hypertext Transport Protocol
HS	Handle Service

I

IETF	Internet Engineering Task Force
IdM	Identity Management
IdP	Identity Provider

J

JAAS	Java Authentication and Authorization Service
JDL	Job Description Language

K

L

LCAS	Local Center Authorisation Service
LCG	Large Hadron Collider Computing Grid
LCMAPS	Local Credential Mapping Service
LDAP	Lightweight Directory Access Protocol
LFC	LCG File Catalogue
LFN	Logical File Name
LRC	Local Replica Catalog

M

MAC	Mandatory Access Control
MAC	Message Authenticity Code

N

NJS	Network Job Supervisor

O

OGSI	Open Grid Services Infrastructure
OU	Organisational Unit

P

PAM	Pluggable Authentication Modules
PBS	Portable Batch System
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIN	Personal Identification Number
PIP	Policy Information Point
PKCS	Public Key Cryptographic Standard
PKI	Public Key Infrastructure
PRP	Policy Retrieval Point
PRS	Proxy Renewal Service
PZ	Proxy-Zertifikat

Q

QoP	Quality of Protection

R

RA	Registration Authority
RB	Resource Broker
RBAC	Role-Based Access Control
RFIO	Remote File Input/Output Protocol
RGMA	Relational Grid Monitoring Architecture
RLS	Replica Location Service
ROC	Regional Operation Center

S

SAML	Security Assertion Markup Language
SASL	Simple Authentication and Security Layer
SP	Service Provider
SE	Storage Element
SRB	Storage Resource Broker
SRM	Storage Resource Management
SSH	Secure SHell
SSL	Secure Socket Layer
SSO	Single Sign-On

T

TLS	Transport Layer Security

U

UNICORE	Uniform Interface to Computing Resources
UI	User Interface
UUDB	Unicore User DataBase

V

VO	Virtual Organisation
VOMS	Virtual Organization Membership Service

W

WAYF	Where Are You From
WMS	Workload Mangement Service
WN	Worker Node
WS	Web Services

X

XML	eXtensible Markup Language

Y

Z

9 Literatur

[ABM04] M. Ahsant, J. Basney, O. Mulmo. Grid Delegation Protocol, Workshop on Grid Security Experiences, 2004. http://www.ncsa.uiuc.edu/~jbasney/Grid-Delegation-Protocol.pdf

[ACC+04] R. Alfieri, R. Cecchini, V. Ciaschini, F. Spataro, L. dell'Agnello, Á. Frohner, K. Lörentey, From gridmap-file to VOMS: managing Authorization in a Grid environment, 2004, http://grid-auth.infn.it/docs/voms-FGCS.pdf

[ADH+02] B. Atkinson et al. Web Services Security (WS-Security). April 2002. ftp://www6.software.ibm.com/software/developer/library/ws-secure.pdf

[Bas05] J. Basney. GFD-E.054 – MyProxy Protocol, Global Grid Forum, GridForum.org, November 2005.

[BBF+05] Tom Barton, Jim Basney, Tim Freeman, Tom Scavo, Frank Siebenlist, Von Welch, Rachana Ananthakrishnan, Bill Baker, and Kate Keahey. Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, Gridshib, and MyProxy. In 5th Annual PKI R&D Workshop, April 2006.

[BHW05] J. Basney, M. Humphrey, V. Welch. The MyProxy Online Credential Repository. Software: Practice and Experience, Volume 35, Issue 9, July 2005, pp. 801-816.

[BNO+04] J. Basney, W. Nejdl, D. Olmedilla, V. Welch, M. Winslett. Negotiating Trust on the Grid. 2nd Workshop on Semantics in P2P and Grid Computing at the Thirteenth International World Wide Web Conference, New York, Mai 2004.

[BSXH05] J. Basney, Z. Sun, D. Xin, Y. Huang, SACRED for Java, v1.2 April 2005, http://sacred.sf.net/

[BWE+00] R. Butler, V. Welch, D, Engert, I. Foster, S. Tuecke, J. Volmer, C. Kesselman. A National-Scale Authentication Infrastructure. *Computer* 33, 12 (Dec. 2000), 60-66. DOI= http://dx.doi.org/10.1109/2.889094.

[BYBS03] J. Basney, W. Yurcik, R. Bonilla, A. Slagell. The Credential Wallet: A Classification of Credential Repositories Highlighting MyProxy. 31st Research Conference on Communication, Information and Internet Policy (TPRC 2003), Arlington, Virginia, September 19-21, 2003.

[CCO+03] S. Cannon, S. Chan, D. Olson, C.Tull. Using CAS to Manage Role-Based VO Sub-Groups. 2003 Conference for Computing in High Energy and Nuclear Physics (CHEP03, März 2003)

[Con06] Condor Team. Condor-G v6.7 Manual. University of Wisconsin-Madison, Februar 2006.

[DCP05] Certification Policy and Certification Practice Statement des DFN-Vereins für Grid-Zertifikate, http://www.dfn.de/pki/grid

[DGAK2+04] D-Grid – AK2 Middleware und Services, Anhang Bestandsaufnahme, 2004, http://www-grid.desy.de/d-grid/ak2/DGrid_AK2_Anhang_Bestandsaufnahme.pdf

[DiAl99] T. Dierks, C. Allen. The TLS Protocol Version 1.0, RFC 2246, 1999.

[EGP06] European Grid Policy Management Authority, http://eugridpma.org/

[Eng04] D. Engert. GSS-API Extensions, Proc. 16th IETF, 2004. http://www3.ietf.org/proceedings/04aug/229.htm

[ESP06] Evaluation of Shibboleth and PKI for Grids project (eSP-grid). http://wiki.oucs.ox.ac.uk/esp-grid/FrontPage

[FBA+03] Luis Ferreira, Viktors Berstis, Jonathan Armstrong, Mike Kendzierski, Andreas Neukoetter, MasanobuTakagi, Richard Bing-Wo, Adeeb Amir, Ryo Murakawa, Olegario Hernandez, James Magowan, Norbert Bieberstein. Introduction to Grid Computing with Globus, IBM Redbooks, 2003. (http://www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf)

[FKTT98] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. 1998. A security architecture for computational grids. In *Proceedings of the 5th ACM Conference on Computer and Communications Security* (San Francisco, California, United States, November 02–05, 1998). CCS '98. ACM Press, New York, NY, 83-92. http://doi.acm.org/10.1145/288090.288111

[FrAn05] T. Freeman, R. Ananthakrishnan. Authorization processing for Globus Toolkit Java Web services. IBM Developerworks Grid Library, 25. Oktober 2005

[GCP05] Certification Policy and Certfication Practice Statement der Zertifizierungsstelle GridKa-CA des Forschungszentrum Karlsruhe, http://grid.fzk.de/ca/gridka-cps.pdf

[GJN04] D. Gustafson, M. Just, M. Nystrom, SACRED – Securely Available Credentials – Credential Server Framework, IETF RFC 3860, April 2004.

[Gri06] GridShib Project. http://gridshib.globus.org/

[Gro06] Grouper Project. http://middleware.internet2.edu/dir/groups/grouper/

[Haz06] K. Hazelton (Ed.). EduPerson Object Class Specification. Internet2 Middleware Architecture Committee for Education, Directory Working Group (MACE-Dir), DRAFT Revision, February 2006 http://www.educause.edu/eduperson/

[IGTF05] International Grid Trust Federation, Grid Policy Management Authority, Working to Establish WorlWide Trust for Grids, http://www.gridpma.org/

[ITU05] ITU-T Recommendation X.509 (08/2005): Information Technology – Open Systems Interconnection – The Directory: Authentication Framework, 2005.

[ITU97] ITU-T Recommendation X.509 (1997 E): Information Technology – Open Systems Interconnection – The Directory: Authentication Framework, June 1997.

[JFB+03] Bart Jacob, Luis Ferreira, Norbert Bieberstein, Candice Gilzean, Jean-Yves Girard, Roman Strachowski, Seong (Steve) Yu. Enabling Applications for Grid Computing with Globus, IBM Redbooks, 2003. (http://www.redbooks.ibm.com/redbooks/pdfs/sg246936.pdf)

[JTE01] K. Jackson, S. Tuecke, D. Engert. TLS Delegation Protocol, Internet Draft, draft-ietf-tls-delegation-01.txt, 2001.

[Kac+04] Peter Kacsuk (MTA SZTAKI), The LHC Grid, 2004, http://www.lpds.sztaki.hu/pvmmpi/download/tutorials/kacsuk_LCG_tutorial.ppt

[KoBa05] D. Kouril, J. Basney. A Credential Renewal Service for Long-Running Jobs. Grid 2005 – 6th IEEE/ACM International Workshop on Grid Computing, Seattle, WA, November 13-14, 2005.

[Korn04] H. Kornmayer. Das Globus-Toolkit, Version 2. Linux-Magazin 06/2004.

[LBK04] M. Lorch, J. Basney, D. Kafura. A Hardware-secured Credential Repository for Grid PKIs. 4th IEEE/ACM International Symposium on Cluster Computing and the Grid, Chicago, Illinois, April 19-22, 2004.

[LCG-2] User Guide Document identifier:CERN-LCG-GDEIS-454439 http://edg-wp2.web.cern.ch/edg-wp2/security/voms/edg-voms-credential.pdf

[LCGUG+05] Antonio Delgado Peris, Patricia Méndez Lorenzo, Flavia Donno, Andrea Sciabà, Simone Campana, Roberto Santinelli, LCG-2 User Guide, Document Identifier: CERN-LCG-GDEIS-454439, 2005. https://edms.cern.ch/file/454439//LCG-2-UserGuide.pdf

[LCGMW+04] Simone Campana, Maarten Litmaath, Andrea Sciabà, LCG-2 Middleware Overview, Document Identifier: CERN-LCG-GDEIS-498079, 2004, http://www.grid.org.tr/servisler/dokumanlar/LCG-mw.pdf

[LCWS+04] Installation of the (gLite-)Gatekeeper+LCAS+LCMAPS and the Workspace Service, http://www.nikhef.nl/grid/lcaslcmaps/installation_notes/INSTALL_WITH_WORKSPACE_SERVICE

[Lib06] Liberty Alliance Project. http://www.projectliberty.org/

[Lin00] J. Linn. Generic Security Service Application Program Interface Version 2, Update 1, RFC 2743, 2000.

[MinR05] Profile for Traditional X.509 Public Key Certification Authorities with secured infrastructure Version 4.0 http://eugridpma.org/guidelines/

[MWTE04] S. Meder, V. Welch, S. Tuecke, D. Engert. GSS-API Extensions, Grid Security Infrastructure (GSI) WG, GFD-E.024 (Experimental), February 2001, revised June 2004 http://www.ggf.org/documents/GFD.24.pdf

[Mye97] J. Myers, Simple Authentication and Security Layer (SASL), IETF RFC 2222, Oktober 1997.

[Nad05] Anthony Nadalin (Editor). Web Services Trust Language (WS-Trust). Februar 2005. ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf

[Nad05a] Anthony Nadalin (Editor). Web Services Secure Conversation Language (WS-SecureConversation). Februar 2005. http://www-128.ibm.com/developerworks/web services/library/specification/ws-secon/

[NTW01] J. Novotny, S. Tuecke, V. Welch. An Online Credential Repository for the Grid: MyProxy. Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001, pp. 104-111.

[PAP+05] Pool Accounts Patch for Globus, http://www.gridsite.org/gridmapdir/

[PWF+02] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke. A Community Authorization Service for Group Collaboration. In *Proceedings of the 3rd international Workshop on Policies For Distributed Systems and Networks (Policy'02)* (June 05–07, 2002). POLICY. IEEE Computer Society, Washington, DC, 50.

[ScCa01] T. Scavo, S. Cantor. SAML Metadata Extension for a Standalone Attribute Requester, Committee Draft, 2001

[Sch04] J. Schlimmer (Editor). Web Services Policy Framework (WS-Policy). September 2004. ftp://www6.software.ibm.com/software/developer/library/ws-policy.pdf

[Shi06a] Shibboleth Project. http://shibboleth.internet2.edu/index.html.

[Shi06b] Shibboleth Technical Introduction. http://shibboleth.internet2.edu/shib-tech-intro.html

[Sig06] Signet Project. http://middleware.internet2.edu/signet/

[Sun00] Sun Microsystems. GSS-API Programming Guide, 2000.

[TW+04] S. Tuecke, V. Welch et al. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, IETF RFC 3820, Juni 2004.

[TWE+04] S. Tuecke, V. Welch, D. Engert, L. Pearlman, M. Thompson. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, RFC 3820, 2004.

[VOMS+04] Ákos Frohner, Vincenzo Ciaschini, VOMS Credential Format, 2004, http://edg-wp2.web.cern.ch/edg-wp2/security/voms/edg-voms-credential.pdf

[WBK+05] von Welch, T. Barton, K. Keahey, F. Siebenlist. Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Collaboration. 4th Annual PKI R&D Workshop: Multiple Paths to Trust. April 19-21, 2005, NIST, Gaithersburg MD

[Wel04] V. Welch. Grid Security Infrastructure Message Specification. (Version 2, vom 9.02.2004).

[Wel05] V. Welch. Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective. (Version 4 vom 12.09.2005). http://www.globus.org/toolkit/docs/4.0/security/GT4-GSI-Overview.pdf

[WFK+04] V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, F. Siebenlist. X.509 Proxy Certificates for Dynamic Delegation. PKI'04 (April 2004).

[WFK+04] V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, F. Siebenlist. X.509 Proxy Certificates for Dynamic Delegation. PKI'04 (April 2004).

[WSF+03] Von Welch, Frank Siebenlist, Ian Foster, John Bresnahan, Karl Czajkowski, Jarek Gawor, Carl Kesselman, Sam Meder, Laura Pearlman, Steven Tuecke. Security for Grid Services. 12th IEEE International Symposium on High Performance Distributed Computing (HPDC-12 '03), 2003.