Attributbasierte Autorisierung im Grid Computing: Vertrauenswürdige Architekturen und sichere Implementierung

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades

Doktor-Ingenieur
(abgekürzt: Dr.-Ing.)

genehmigte Dissertation

von
M. Sc. Ralf Gröper
geboren am 30.06.1978 in Hamburg

2011

Erster Referent: Prof. Dr. Matthew Smith

Zweiter Referent: Prof. Dr. Kurt Schneider

Tag der Promotion: 27.09.2011

Danksagung

Mein besonderer Dank gilt Herrn Prof. Smith für das mir entgegengebrachte Vertrauen und die Unterstützung beim Erstellen dieser Arbeit. Darüber hinaus danke ich Herrn Dr. Grimm, von dem ich zu Beginn der Arbeit betreut wurde und Frau Prof. von Voigt, die als Leiterin des Lehrstuhls immer eine offenes Ohr für mich hatte.

Herrn Prof. Schneider danke ich für die Übernahme des Koreferats.

Meinen aktuellen und ehemaligen Kollegen am Lehrgebiet Rechnernetze bzw. der Distributed Computing & Security Group danke ich herzlich für die hilfreichen und anregenden Diskussionen zum Themenkomplex dieser Arbeit sowie die stets persönliche und vertrauensvolle Arbeitsatmosphäre.

Meiner Familie und meiner Freundin Janina Schirmer danke ich ganz besonders für die Unterstützung und den Rückhalt während der Erstellung dieser Arbeit.

Hannover, den 25.03.2011
Ralf Gröper

Zusammenfassung

Das Grid Computing hat sich inzwischen als Standard für organisationsübergreifendes wissenschaftliches Arbeiten mit IT-Infrastrukturen etabliert. Das Ziel des Grid Computing ist es, Rechen- und Speicherressourcen über Organisationsgrenzen hinweg gemeinsam nutzen zu können. Dies beinhaltet neben gemeinsamen Schnittstellen für den Zugriff auf diese Ressourcen auch Authentifizierungs- und Autorisierungsfunktionen, die den sicheren organisationsübergreifenden Einsatz der Ressourcen erlauben. Bisher wird allerdings im Bereich der Autorisierung noch nicht das volle Potential ausgeschöpft, das durch die Verwendung von Nutzerattributen aus unterschiedlichen Quellen möglich wäre.

Ausgangspunkt der in dieser Arbeit vorgestellten Forschungen sind die neuartigen Anforderungen, die sich für Autorisierungsinfrastrukturen in Grid-Umgebungen ergeben. Aktuelle Grid-Infrastrukturen verwenden vorwiegend die wenig flexible und für große Nutzerzahlen nicht skalierende identitätsbasierte Autorisierung, da die Voraussetzungen für feingranulare attributbasierte Autorisierungen in diesem Kontext noch nicht gegeben sind. Die besondere Herausforderung liegt darin, dass die Attribute aus mehreren Quellen, beispielsweise dem Nutzermanagement der Grid-Infrastruktur und dem Identitätsmanagement der Heimateinrichtung des Nutzers, stammen und dass diese Quellen keine organisatorische Beziehung zu den geschützten Ressourcen haben.

In dieser Arbeit werden Probleme identifiziert, die in einer solchen Infrastruktur zum Verlust von Vertrauen in die Korrektheit der auf Grid Ressourcen zur Verfügung stehenden Attribute führen. Der Vertrauensverlust kann dadurch zustande kommen, dass nicht eindeutig und explizit geprüft werden kann, ob vorhandene Nutzerattribute auch denjenigen Nutzer beschreiben, der den Zugriff initiiert hat. Weiterhin müssen die Attribute eindeutig einer für sie zuständigen Attribute Authority zugeordnet werden können. Es muss sichergestellt sein, dass genau diese Attribute Authority diese Attribute ausgestellt hat und keine dritte Instanz.

Die Identifizierung und Lösung der Vertrauensprobleme ist die zentrale Herausforderung dieser Arbeit. Kein existierender Ansatz bietet bisher die notwendigen Funktionen und löst gleichzeitig die aufgezeigten Probleme. Basierend auf der Anforderung nach attributbasierter Autorisierung werden, unter Beachtung der identifizierten Probleme, Lösungen konzipiert und analysiert. Hierbei wird besonderes Augenmerk auf die Vertrauenswürdigkeit der vorgestellten Lösungen gelegt. Weiterhin

wird überprüft, ob die vorgestellten Lösungen zu existierenden Grid-Infrastrukturen kompatibel sind und somit bereits erfolgreich eingeführte und betriebene Komponenten weiter genutzt werden können. Es wird eine Lösung ausgewählt, welche die Anforderungen vollständig umsetzt und dargelegt, wie diese Lösung konkret in einer existierenden Grid-Infrastruktur umgesetzt werden kann. Für die Verwendung der verifizierten Attribute in der eigentlichen Autorisierungsentscheidung auf den Grid-Ressourcen wird eine Erweiterung der attributbasierten Autorisierung entwickelt, die die Herkunft der Attribute aus mehreren unabhängigen Quellen explizit berücksichtigt.

Für viele wissenschaftliche Communities ist die Verwendung des Grids attraktiv, da es die Kollaboration mit Wissenschaftlern aus anderen Einrichtungen vereinfacht und gleichzeitig Rechen- und Speicherressourcen bereitstellt, die lokal häufig nicht ausreichend vorhanden sind. Die Anpassung der existierenden Software aus diesen Communities an die Schnittstellen des Grids, insbesondere an die Grid Sicherheitsfunktionen, stellt allerdings zu hohe Anforderungen an die domänenspezifischen Softwareentwickler. Daher ist eine Abstraktion dieser Schnittstellen notwendig. Ein solche Abstraktion wird in dieser Arbeit definiert und eine Java-basierte Implementierung vorgestellt.

Insgesamt werden in dieser Arbeit die Voraussetzungen geschaffen, die notwendig sind, um die Umstellung von identitätsbasierter Autorisierung auf attributbasierte Autorisierung vollziehen zu können. Diese Umstellung ist notwendig, da die identitätsbasierte Autorisierung zum einen nicht flexibel genug ist, um alle Anwendungsfälle im Grid Computing umzusetzen und zum anderen, da sie für die wachsende Nutzerzahl im Grid nicht ausreichend skalierbar ist. Darüber hinaus wird die Voraussetzung für domänenspezifische Softwareentwickler geschaffen, ihre Software mit niedrigem Aufwand erfolgreich an die speziellen Erfordernisse im Grid anzupassen.

Schlagwörter: Grid Computing, Attributbasierte Autorisierung, Verteilte Systeme

Abstract

Grid Computing evolved into an accepted standard for inter-organizational scientific work with IT Infrastructures. It is the aim of Grid Computing to share compute- and storage resources across organizational boundaries. This includes, besides common interfaces for accessing these resources, authentication and authorization mechanisms that allow the inter-organizational application of these resources. Regarding authorization, the full potential resulting from the use of user attributes from multiple sources for authorization is not used until now.

Starting point of this thesis are novel requirements that unfold for authorization infrastructures in Grid environments. Current Grid infrastructures use identity-based authorization, which is not very flexile and does not scale well for big numbers of users. The preconditions for fine-grained attribute-based authorization are not met yet in this context. The challenge lies in the fact that attributes from multiple sources, e. g. the user management of the Grid infrastructure and the identity management system of the user's home organization, have to be used. These sources do not share an organizational relationship to the resources that need to be protected.

In this thesis, problems are being identified that can lead to a loss of trust in the correctness of the attributes available on Grid resources in such an infrastructure. This loss of trust can arise because it cannot be verified unambiguously and explicitly whether available attributes describe in fact the user that initiated a request. Additionally, it must be possible to check which attribute authority issued a specific attribute. It must be made sure that this attribute authority issued these attributes and not a third party.

Identification and finding solutions for these trust problems is the main challenge of this thesis. No approach currently available implements the necessary features and solves all identified trust issues. Based on the requirement for attribute-based authorization solutions are being designed and analyzed, taking all trust issues into account. Special interest is being laid on the trustworthiness of these solutions. Additionally, it is taken into account whether the solutions are compatible to existing Grid infrastructures and whether already used components can still be used. One solution that implements all requirements is chosen. Concepts on how to implement that solution into an existing Grid infrastructure are presented. For using the verified attributes in an authorization decision, an extension of attribute-based access control is being developed which explicitly accounts for attributes from multiple sources.

Using the Grid is attractive for many scientific communities because it facilitates collaboration with scientists from other organizations and supplies compute- and storage resources that are often not available locally. Adapting existing software from these communities to the interfaces of the Grid and especially to the Grid security concepts requires too much from domain specific software developers due to the high complexity of these interfaces and concepts. Thus, an abstraction that encapsulates this complexity is necessary. Such an abstraction is being defined and a Java-based implementation presented in this thesis.

Altogether, in this thesis the preconditions for the conversion from identity-based authorization to attribute-based authorization with multiple attribute authorities are being developed. This conversion is necessary as identity-based authorization is on the one hand not flexible enough to implement all requirements in Grid computing and, on the other hand, does not scale for growing user numbers in the Grid. Additionally, the preconditions for domain-specific software developers are being met to enable them to adapt their software successfully to the Grid.

Key Words: Grid Computing, Attribute Based Authorization, Distributed Systems

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AA	Attribute Authority
AAA	Authentication, Authorization and Accounting
AAI	Authentifizierungs- und Autorierungsinfrastruktur
ABAC	Attribute Based Access Control
ABMAC	Attribute Based Multipolicy Access Control
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BPEL	Business Process Execution Language
C	Country
CA	Certification Authority
CN	Common Name
CPU	Central Processing Unit
CSR	Certificate Signing Request
DFN	Deutsches Forschungsnetzwerk
DFN-AAI	Authentifizierungs- und Autorierungsinfrastruktur des Deutschen Forschungsnetzwerkes
DFN-PKI	Public Key Infrastructure des Deutschen Forschungsnetzwerkes
DGI	D-Grid Integrationsprojekt
DN	Distinguished Name
DS	Discovery Service
ePPN	eduPersonPrincipalName
FQDN	Fully Qualified Domain Name
GAT	Grid Application Toolkit
GDI-Grid	Geodaten-Infrastrukturgrid
GIS	Geoinformationssystem
GRAM	Grid Resource Allocation Manager

GRRS	Grid Resource Registration Service
GSI	Grid Security Infrastructure
GT	Globus Toolkit
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identification
IdP	Identity Provider
IT	Informationstechnik
MAABAC	Multiple Attribute Authority Based Access Control
O	Organization
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
OGSA-DAI	Open Grid Services Architecture Data Access and Integration
OU	Organizational Unit
OWS	OpenGIS Web Service
PAP	Policy Administraton Point
PBS	Portable Batch System
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
PKCS12	Public-Key Cryptography Standard 12
PKI	Public Key Infrastructure
PMA	Policy Management Authority
RA	Registration Authority
RBAC	Role Based Access Control
RFC	Request for comments
RFT	Reliable File Transfer Service
SAGA	Simple API for Grid Applications
SAML	Security Assertion Markup Language
SLC	Short Lived Certificate
SLCS	Short Lived Certificate Service
sn	surname

SOAP	ursprünglich: Simple Object Access Protocol, heute als Eigenname genutzt
SP	Service Provider
SSH	Secure Shell
SSO	Single Sign-On
TLS	Transport Layer Security
UID	User Identification
UNICORE	Uniform Interface to Computing Resources
URL	Uniform Resource Locator
USA	United States of America
UUDB	UNICORE User Database
VASH	VOMS Attributes from Shibboleth
VM	Virtual Machine
VO	Virtuelle Organisation
VOMRS	Virtual Organization Management Registration Service
VOMS	Virtual Organization Membership Service
VPN	Virtual Private Network
WAYF	Where Are You From
WML	Web Map Service
WPS	Web Processing Service
WSDL	Web Services Description Language
WSRF	Web Services Resource Framework
XACML	eXtensible Access Control Markup Language
XML	Extensible Markup Language

Kapitel 1

Einleitung

1.1 Einführung und Motivation

In den vergangenen Jahrzehnten sind IT-Infrastrukturen für alle erdenklichen Zwecke aufgebaut worden. Insbesondere im kommerziellen Sektor und im Forschungsbereich sind sehr große und komplexe Systeme entstanden, die häufig schützenswerte Ressourcen enthalten. Dies betrifft entweder Daten, die vertraulich bleiben müssen oder teure Ressourcen, die nicht durch unrechtmäßige Nutzung verschwendet werden dürfen. Deshalb haben sich parallel zu den eigentlichen Fachinfrastrukturen Sicherheitsinfrastrukturen entwickelt, die diesen in ihrer Unterschiedlichkeit und Inkompatibilität untereinander in nichts nachstehen. Dies stellte in der Vergangenheit aber nur in Ausnahmesituationen ein Problem dar, weil die einzelnen Systeme unabhängige Insellösungen darstellten. Nur beispielsweise bei der Fusion zweier Industrieunternehmen mussten ihre inkompatiblen IT-Infrastrukturen aneinander angepasst und verschmolzen werden.

Durch die allgegenwärtige Vernetzung nahezu aller IT Systeme durch das Internet hat sich dies grundlegend geändert: IT-Systeme unabhängiger Organisationen werden mittels geeigneter Schnittstellen verbunden und sind so in der Lage, gemeinsame Ziele der beteiligten Organisationen zu verfolgen. Dies stellt grundlegend neuartige Anforderungen an die Sicherheitsinfrastrukturen und auch an die Entwickler solcher vernetzter Systeme. Der entscheidende Faktor, warum bestehende Ansätze zur Absicherung von IT Systemen nicht unverändert übernommen werden können, liegt in der föderierten Struktur dieser neuartigen Kollaborationen begründet. So kollaborieren Organisationen miteinander, die keine direkte juristische Beziehung zu einander haben. Daher reicht auf der technischen Ebene implizites, durch Verträge gesichertes Vertrauen zwischen den Partnern für die Autorisierung nicht mehr aus. Für Autorisierungsentscheidungen werden Aussagen über die anfragenden Entitäten, z. B. den Benutzer[1], benötigt. Ein genaues Verständnis, welche Autorität warum welche Aussage über wen trifft und treffen darf, und wie die Validität einer solchen Aussage sichergestellt werden kann, ist Kernfrage dieser Arbeit.

[1] In diesem Text wird der Einfachheit halber nur die männliche Form verwendet. Die weibliche Form ist selbstverständlich immer mit eingeschlossen.

1.2 Ziele

Ziel der Arbeit ist es, erforderliche Konzepte und Lösungen für die Anforderungen moderner föderierter IT-Infrastrukturen im Bereich Sicherheit zu finden und zu modellieren. Es müssen Ansätze erforscht werden, die notwendig sind, um die identifizierten Anforderungen umzusetzen. Im Kontext dieser Arbeit wird die Umsetzbarkeit anhand des Paradigmas des Grid Computing dargestellt, die gewonnenen Erkenntnisse können aber generisch auf organisationsübergreifende IT-Infrastrukturen angewandt werden. Ein zusätzlicher Aspekt ist die Konzeption der Überführung der erforschten Aspekte in den realen Betrieb einer Grid-Infrastruktur. Durch Entwicklung und Implementierung einer Abstraktionsschicht für Softwareentwickler, die ihre Produkte an das Grid und seine komplexen Schnittstellen und Sicherheitsanforderungen anpassen möchten, wird sichergestellt, dass diese Software im Grid zuverlässig und sicher funktioniert.

1.3 Aufbau der Arbeit

Ausgehend von den Grundlagen des Grid Computing und der verteilten Autorisierung in Kapitel 2 werden in Kapitel 3 Aspekte analysiert, die bei Autorisierungsentscheidungen mit Attributen aus mehreren Quellen beachtet werden müssen, um die Vertrauenswürdigkeit der Entscheidung sicherzustellen. Im Anschluss werden Fehlerquellen im Umgang mit Grid-Infrastrukturen mit komplexen Authentifizierungs- und Autorisierungsinfrastrukturen analysiert.

Darauf aufbauend wird in Kapitel 4 ein erweitertes Konzept der attributbasierten Autorisierung in verteilten Umgebungen mit mehreren Attribute Authorities erarbeitet. Um die erkannten Fehlerquellen durch unsachgemäßen Umgang durch Softwareentwicklern mit den komplexen Schnittstellen des Grids zu umgehen wird eine Abstraktionsschicht definiert, die die Verwendung des Grids durch Entwickler vereinfacht und potentielle Fehlerquellen vor ihnen verbirgt.

Anhand einer real existierenden Grid-Infrastruktur werden in Kapitel 5 Lösungen für die Umsetzung des entwickelten Konzepts für die vertrauenswürdige Bereitstellung von Attributen aus mehreren Quellen auf Grid-Ressourcen präsentiert. Darüberhinaus wird eine Java-basierte Implementierung der definierten Abstraktionsschicht für Softwareentwickler vorgestellt.

Abschließend wird in Kapitel 6 das Erreichte kritisch bewertet und abschließend in Kapitel 7 Fazit und Ausblick präsentiert.

Kapitel 2

Grundlagen der Sicherheit im Grid Computing

2.1 Grid Computing und verteilte Systeme

2.1.1 Warum Grid Computing?

Das Paradigma des Grid Computing ist Anfang der 1990er Jahre entstanden. Seitdem wurden mehrere, sich ergänzende Definitionen des Begriffes geprägt, wobei es von dem konkreten Anwendungsfall abhängt, welche dieser Definitionen, auf eine gegebene Grid-Infrastruktur zutrifft.

Der Begriff Grid Computing leitet sich vom englischen *Power Grid*, also dem Stromnetz, ab. Darauf begründet sich auch der Slogan „Rechenleistung aus der Steckdose“. Es geht darum, dass Rechenleistung, Speicherkapazität und andere Computerressourcen nicht mehr in erster Linie lokal im Rechner durch CPU und Festplatten bereitgestellt werden, sondern, wie elektrischer Strom, aus der Steckdose bezogen werden. Hierbei soll der Nutzer sich nicht um die Details der Bereitstellung kümmern müssen, so wie er als Stromverbraucher auch keinerlei Kenntnisse über Kraftwerkstechnik benötigt. Diese Vision wurde zuerst 1965 von V. A. Vyssotsky et. al. publiziert [59], allerdings noch nicht unter dem Begriff „Grid Computing“.

Spezifischere Definitionen des Begriffes wurden ab dem Ende der 1990er Jahre geprägt. Ian Foster und Carl Kesselmann definierten den Begriff 1998 in dem Buch „The Grid: Blueprint for a New Computing Infrastructure.“ wie folgt:

> „A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.“ [39]

In dieser frühen Definition wird das Konzept des Grid Computing noch auf den entfernten Zugriff auf Hochleistungsressourcen beschränkt. Dieser Zugriff soll zuverlässig, konsistent, allgegenwärtig und preiswert sein.

Diese Definition wurde 2003 in der zweiten Auflage des Buchs verfeinert zu

> „The sharing that we are concerned with is not primarily file exchange but rather direct access to computers, software, data, and other resources, as is required by a range of collaborative problem-solving and resource-brokering strategies emerging in industry, science, and engineering. This sharing is, necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs.

> A set of individuals and/or institutions defined by such sharing rules form what we call a virtual organization.“ [25]

In dieser Definition werden mehrere Aspekte eingeführt, die für diese Arbeit besondere Bedeutung haben. So wird eine Autorisierungsinfrastruktur gefordert, die überwacht, wer berechtigt ist, gemeinsame Ressourcen einzubringen und wer berechtigt ist, diese zu benutzen. Eine Menge von Individuen und/oder Organisationen, deren Zusammenarbeit über solche Regeln definiert ist, wird *Virtuelle Organisation (VO)* (siehe 2.1.3) genannt.

Später (2002) stellt Ian Foster in „What is the Grid? A Three Point Checklist“ drei Bedingungen auf, die erfüllt sein müssen, damit eine Infrastruktur dem Grid Paradigma entspricht:

1. „Computing resources are not administered centrally.
2. Open standards are used.
3. Nontrivial quality of service is achieved.“ [23]

Es darf also keine zentrale Administration der Ressourcen erfolgen. Sie sind nicht nur räumlich, sondern auch organisatorisch getrennt. Weiterhin müssen offene Standards eingesetzt werden. Dies stellt sicher, dass eine Infrastruktur keine Insellösung ist, sondern potentiell beliebig erweitert werden kann. Dies beinhaltet die Vision, dass es in Zukunft keine getrennten Grid-Infrastrukturen mehr geben wird, sondern dass alle Ressourcen ein weltweites Grid darstellen. Das heißt nicht, dass jeder Nutzer beliebig auf jede Ressource zugreifen *darf*, sondern dass er es, unter der Voraussetzung einer gültigen Autorisierung, *kann*. Zuletzt wird eine nichttriviale Dienstgüte gefordert. Dies bedeutet, dass das Grid Mechanismen benötigt, um durch das Zusammenspiel der Ressourcen Dienste anbieten zu können, die auf getrennten Ressourcen ohne Gridverbund nicht möglich wären. Die Nützlichkeit des Gridverbundes muss deutlich größer sein, als die Summe der Nützlichkeiten der einzelnen Ressourcen.

Das D-Grid

In Deutschland wird der Aufbau einer nationalen Grid-Infrastruktur, dem *D-Grid*, durch das Bundesministerium für Bildung und Forschung gefördert. Im D-Grid arbeiten wissenschaftliche Communities aus allen Bereichen mit dem D-Grid Integrationsprojekt (DGI) zusammen, um eine Infrastruktur zu schaffen, die den Bedürfnissen aller beteiligen Communities entspricht. Da einige dieser Communities bereits

vor Beginn des D-Grid Projektes mit Gridtechnologien gearbeitet haben und diese nicht aufgeben konnten, mussten Wege gefunden werden, diese Technologien zu einer gemeinsamen Infrastruktur zu kombinieren. Basis dessen sind die Grid Middlewares *Globus Toolkit*, *UNICORE* und *gLite*. Neben der anforderungsgetriebenen Konzeption der Sicherheitsinfrastruktur stellt die Unterstützung dieser untereinander bisher nicht kompatiblen Middlewares die zentrale Herausforderung dieser Arbeit dar. Durch die Verwendung der im Grid Computing geforderten offenen Standards wird sichergestellt, dass zukünftige Middlewares ebenfalls unterstützt werden.

2.1.2 Warum Sicherheit im Grid?

In Grid-Infrastrukturen für das wissenschaftliche Rechnen werden schützenswerte Ressourcen über das Internet vernetzt und online zugreifbar. Drei Grundtypen von Ressourcen, die eine Autorisierungsinfrastruktur notwendig machen, lassen sich unterscheiden:

Daten: Daten können aus unterschiedlichen Gründen schützenswert sein. Zum einen können Daten Lizenzbestimmungen unterliegen, die eine Weitergabe nicht erlauben. Hierunter fallen häufig z. B. Geodaten, die von einem Dienstleister eingekauft wurden. Andere Daten können schützenswert sein, da insbesondere in kompetitiven Bereichen der Wissenschaft, aber auch in der Wirtschaft, die Mitbewerber einen Vorteil erlangen würden, wenn sie die Daten eines Gridteilnehmers einsehen könnten. Da das Grid von mehreren Marktteilnehmern gemeinsam genutzt wird, muss dies über eine Sicherheitsinfrastruktur verhindert werden.

Lizenzpflichtige Programme: Lizenzpflichtige Software unterliegt häufig einer genauen Kontrolle, wie viele Instanzen des Programms auf wie vielen Rechnern, CPUs oder von wie vielen Nutzern gleichzeitig betrieben werden. Dies stellt in Grids eine besondere Herausforderung dar, da viele Lizenzmodelle und technische Implementierungen von Lizenzmanagementsystemen bisher nicht an die Voraussetzungen von Grid-Systemen angepasst sind. Hinzu kommt, dass die Lizenzen sowohl von den Ressourcenbetreibern als auch von den Nutzern der Ressourcen beschafft werden können. Auch diese beiden Fälle erfordern unterschiedliche Funktionen in den Lizenzmanagementsystemen. Daher kann die im Grid verwendete Sicherheitsinfrastruktur auf diesen Anwendungsfall

vorbereitet werden, aber ihn nicht ohne Anpassungen bei den Lizenzgebern umsetzen.

Teure Ressourcen: Hochleistungsressourcen wie Speichersysteme mit hunderten von Terabyte oder Clustersysteme mit zehntausenden von CPUs sind sehr teuer in der Anschaffung und im Unterhalt. Um diese Kosten zu decken muss sichergestellt werden, dass nur berechtigte Nutzer diese Ressourcen einsetzen können und die Ressourcen nicht durch unberechtigte Nutzer blockiert werden.

Es ist notwendig, diese schützenswerten Ressourcen durch wirksame technische und organisatorische Mechanismen gegen unerlaubte Zugriffe zu sichern. Hierbei dürfen die Vorteile des Grid Computing allerdings nicht eingeschränkt werden. Nach der oben genannten Definition des Grid Computing müssen also auch die Sicherheitsmechanismen einer Grid-Infrastruktur die genannten Voraussetzungen erfüllen und eine nicht-zentrale Administration ermöglichen, auf offenen Standards basieren und nichttriviale Dienstgüten ermöglichen.

2.1.3 Virtuelle Organisation

Die Virtuelle Organisation, meist kurz nur VO genannt, ist eines der zentralen Konzepte im Grid Computing. IT-Ressourcen und Personen sind in der realen Welt realen Organisationen zugeordnet. Im Grid werden sowohl die IT-Ressourcen als auch die Personen neu organisiert, so dass sie an gemeinsamen Zielen arbeiten können. Es entsteht eine neue Form der Organisation, die sich von den beteiligen realen Organisation in mehreren Aspekten unterscheidet.

Es gibt sehr viele Definitionen für Virtuelle Organisationen. Eine allgemeine Definition findet sich in Schiffers 2007:

> „Eine Virtuelle Organisation ist eine zeitlich begrenzte koordinierte Kooperation von Elementen in Form von Individuen, Gruppen von Individuen, Organisationseinheiten oder ganzer Organisationen, die Teile ihrer physischen oder logischen Ressourcen oder Dienste auf diesen, ihre Kenntnisse und Fähigkeiten sowie Teile ihrer Informationsbasis in Form virtueller Ressourcen und Dienste über eine Grid-Infrastruktur derart zur Verfügung stellen, dass die gemeinsam vereinbarten Ziele unter Berücksichtigung lokaler und globaler Policies erreicht werden können.“ [52]

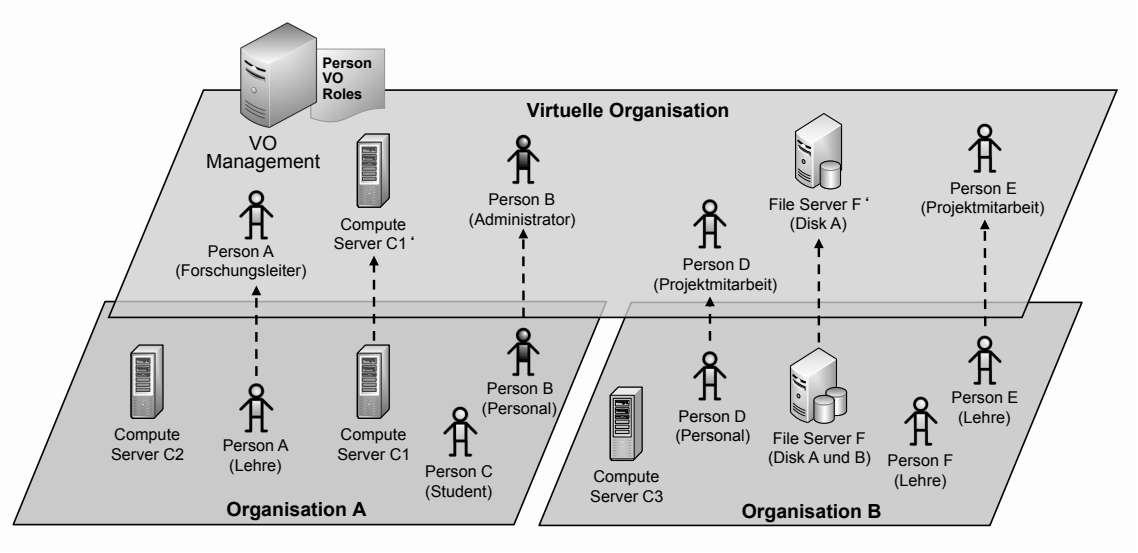

Abbildung 2.1: Aufbau einer Virtuellen Organisation

Diese Definition spart explizit den Aspekt einer Eigentümerschaft der in der VO organisierten Ressourcen aus. Sie bleiben Eigentum der beteiligten realen Organisationen, werden aber vollständig oder teilweise zur Nutzung durch die Virtuelle Organisation freigegeben.

Die Komponenten einer Virtuellen Organisation und deren Zusammenhang sind in Abb. 2.1 dargestellt. Teilmengen der Personen und Ressourcen mehrerer realer (nicht virtueller) Organisationen A und B werden zusammengefasst, um an einem gemeinsamen Ziel zu arbeiten. Während die Personen der einen realen Organisation als solche keinen Zugriff auf die Ressourcen einer anderen realen Organisation haben, wird diese Unterscheidung durch die Bildung der Virtuellen Organisation aufgehoben: Personen und Ressourcen befinden sich zwar weiterhin in unterschiedlichen realen Organisationen, aber in einer gemeinsamen Virtuellen Organisation. Durch die Rechte innerhalb der Virtuellen Organisation ist der Zugriff auf die Ressourcen nun möglich. So sind die Personen A und B der realen Organisation A auch Mitglieder der Virtuellen Organisation. In der Virtuellen Organisation bekleiden sie andere Rollen als in ihrer realen Organisation. Der Computeserver C1 aus Organisation A wird der Virtuellen Organisation als Computeserver C1' zur Verfügung gestellt. Dieser kann sich vom realen Computeserver C1 unterscheiden, indem beispielsweise nur ein Teil der vorhandenen Nodes oder CPUs für die Verwendung in der Virtu-

ellen Organisation bereitgestellt wird. Person C und Computeserver C2 werden der Virtuellen Organisation nicht zur Verfügung gestellt. In der realen Organisation B stellt sich dies analog dar.

Die Mitgliedschaften der Personen und ihre Attribute in der Virtuellen Organisation werden von einem VO Managementsystem verwaltet. Attribute können beispielsweise Nutzern Rollen zuordnen, die unabhängig von ihren Rollen in der realen Organisation sind.

Virtuelle Organisationen sind eine der beiden Hauptquellen für Nutzerattribute und -rechte im Grid Computing. Weitere Quellen von Nutzerattributen (weniger von Nutzerrechten) sind die Heimatorganisationen der VO-Mitglieder. Die Heimatorganisation der VO-Mitglieder entspricht der realen Organisation, der sie angehören. In dieser Arbeit werden die VO-Managementsysteme als Quellen von Attributen gesehen, die für die Autorisierung auf Grid-Ressourcen verwendet werden können. Der innere Aufbau und die Abläufe des verwendeten VO-Managementsystems sind von nachrangiger Bedeutung.

2.2 Authentifizierung im Grid Computing

Zugriffsschutz in IT-Systemen wird meist durch zwei aufeinander aufbauende Schritte realisiert. Zuerst wird bei der Authentifizierung festgestellt, wer einen Zugriff initiiert. Anschliessend wird in der Autorisierung überprüft, ob dieser Zugriff erlaubt ist. Im Folgenden wird zuerst die Authentifizierung mittels Public Key Infrastrukturen erläutert. Anschliessend werden die Grundlagen der Autorisierung in verteilten Systemen vorgestellt.

2.2.1 Authentifizierung mittels Public Key Infrastrukturen

Authentifizierung ist die Überprüfung einer vorgegeben Behauptung von einem oder über einen Kommunikationspartner. In der IT-Sicherheit betrifft dies üblicherweise die vorgegebene Identität des Kommunikationspartners. Wird die vorgegebene Identität verifiziert, ist die Authentifizierung erfolgreich. Andernfalls ist sie gescheitert. Die Authentifizierung ist zwingende Voraussetzung für den Großteil der Autorisierungsentscheidungen im Grid Computing (siehe Kapitel 2.4).

Die Grundlage der Authentifizierung im Grid bildet die asymmetrische Kryptographie sowie die vertrauenswürdige Verteilung von öffentlichen kryptographischen

Schlüsseln nach dem X.509 Public Key Infrastructure (PKI) Standard [36]. Mit Hilfe der an Zertifikate gebundenen öffentlichen Schlüssel und der zugehörigen privaten Schlüssel ist es möglich, Nachrichten zu *verschlüsseln*, um die Vertraulichkeit von übertragenen Nachrichten sicherzustellen. Weiterhin können Nachrichten *signiert* werden, um ihre Authentizität zu sichern.

In einer PKI werden von einer vertrauenswürdigen dritten Instanz, einer Certification Authority (CA), überprüfte Identitäten an öffentliche kryptographische Schlüssel gebunden. Eine solche Verbindung aus Identität und öffentlichem Schlüssel wird *Zertifikat* genannt. Die Aufgabe einer CA ist es sicherzustellen, dass alle von ihr ausgegebenen Zertifikate vertrauenswürdig sind. Zwei Aspekte sind besonders wichtig, um dies zu gewährleisten:

Zum einen muss die CA sicherstellen, dass die Überprüfung der Antragsteller den Richtlinien entspricht. Damit wird sichergestellt, dass sich niemand ein Zertifikat auf den Namen einer anderen Person ausstellen lassen kann. Dies geschieht in der Regel durch die Ernennung von Registration Authorities (RA). Diese sind vor Ort bei den an einer PKI teilnehmenden Einrichtungen angesiedelt und nehmen Anträge entgegen. Gemäß den Policies der CA kann hier beispielsweise ein persönliches Erscheinen und Vorweisen eines amtlichen Ausweisdokuments gefordert sein.

Der zweite Aspekt, den eine CA sicherstellen muss, um ihre Vertrauenswürdigkeit zu gewährleisten, ist der unbedingte Schutz ihres privaten Schlüssels. Dieser zum CA-Zertifikat gehörende private Schlüssel wird genutzt, um die ausgestellten Zertifikate zu signieren. Wird dieser Schlüssel kompromittiert, kann er missbräuchlich verwendet werden, um unautorisierte Zertifikate im Namen der CA auszustellen. Auf Ressourcen, die anhand der Zertifikate authentifizieren, kann dieser Missbraucht nicht festgestellt werden. Um diesen privaten Schlüssel zu schützen wird er nach den Policies der CA beispielsweise auf einem Rechner verwaltet, der nicht am öffentlichen Internet angeschlossen ist und über spezialisierte Hardware verfügt, welche den Schlüssel sicher vorhält.

In der aktuellen Grid-Infrastruktur in Europa gibt es in der Regel eine CA pro Land. In Deutschland gibt es mit der GridKA CA am Karlsruher Institut für Technologie und der Grid CA im Rahmen der DFN-PKI am DFN Verein zwei Grid CAs. Da auch der länderübergreifende Einsatz im Grid Computing vorgesehen ist, müssen sich diese CAs auf miteinander kompatible Sicherheitsniveaus und Policies einigen. Dies geschieht in den Gremien der Policy Management Authorities (PMA). Es gibt

für jeden Kontinent eine zuständige PMA. In Europa ist dies die EUGridPMA [18].

In Abb. 2.2 ist der Inhalt eines X.509 von der Grid CA der DFN-PKI ausgestellten Zertifikats dargestellt. Wichtige Bestandteile des Zertifikats sind:

Seriennummer des Zertifikats: Sie dient der eindeutigen Identifizierung eines Zertifikats, beispielsweise um es zurückzurufen, wenn der private Schlüssel kompromittiert wird (Zeile 4).

Distinguished Name des Ausstellers: Dies ist der Distinguished Name desjenigen Zertifikats, mit dessen privaten Schlüssel das betractete Zertifikat signiert wurde (Zeile 6). Der Eintrag dient dazu, die Zertifikatskette bis zum CA-Zertifikat zurückverfolgen zu können. Außerdem kann eine Ressource anhand des so identifizierten ausstellenden Zertifikats die Gültigkeit der Signatur prüfen.

Gültigkeitszeitraum des Zertifikats: Jedes Zertifikat hat eine Gültigkeitsdauer, definiert durch das Startdatum und das Enddatum (Zeilen 7-9). Ausserhalb dieses Zeitraums darf das Zertifikat nicht akzeptiert werden.

Distinguished Name des Subjekts: Das Subjekt bezeichnet eindeutig den Eigentümer des Zertifikats. Der Distinguished Name des Subjekts besteht aus mehreren Komponenten, welche Land (C), Organisation (O), Organisationseinheit (OU) und den Common Name des Subjektes (CN) beschreiben (Zeile 10).

Öffentlicher kryptographische Schlüssel: Dieser wird verwendet, um mit dem zum Zertifikat gehörenden privaten Schlüssel verschlüsselte Nachrichten entschlüsseln zu können (Zeilen 11-18, gekürzt dargestellt).

Kritische Zertifikatserweiterungen: Diese müssen von allen Komponenten, die das Zertifikat verwenden, berücksichtigt werden. Ist einer Komponente eine vorhandene kritische Erweiterung unbekannt, muss das Zertifikat abgelehnt werden. Von besonderer Bedeutung ist in der Grid Security Infrastructure die kritische Erweiterung mit dem Inhalt `CA:FALSE` (Zeilen 20-23).

Unkritische Zertifikatserweiterungen: Diese Erweiterungen beinhalten zusätzliche Informationen, die von Komponenten, welche das Zertifikat verwenden,

```
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 259237689 (0xf73a739)
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=DE, O=DFN-Verein, OU=DFN-PKI, CN=DFN-Verein PCA Grid - G01
    Validity
      Not Before: Jan 18 09:28:09 2010 GMT
      Not After : Feb 17 09:28:09 2011 GMT
    Subject: C=DE, O=GridGermany, OU=Leibniz Universitaet Hannover, OU=RRZN, CN=Ralf Groeper
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
        RSA Public Key: (2048 bit)
        Modulus (2048 bit):
          00:e0:e5:61:96:00:e6:d2:b0:97:ac:aa:40:a1:b7:
          [...]
          cc:d7
        Exponent: 65537 (0x10001)
    X509v3 extensions:
      X509v3 Basic Constraints: critical
        CA:FALSE
      X509v3 Key Usage: critical
        Digital Signature, Key Encipherment, Data Encipherment
      X509v3 Extended Key Usage:
        TLS Web Client Authentication, E-mail Protection
      X509v3 Subject Key Identifier:
        48:B5:D1:69:59:32:CD:5C:85:01:42:1C:5C:B8:9C:F8:51:D5:BD:F8
      X509v3 Authority Key Identifier:
        keyid:96:EC:DC:AD:9A:C3:FE:50:A3:3C:22:E5:3D:C2:C5:FF:CA:D9:22:C6
      X509v3 Subject Alternative Name:
         email:groeper@rvs.uni-hannover.de
      X509v3 Certificate Policies:
         Policy: 1.3.6.1.4.1.22177.300.1.1.3.1.4
         Policy: 1.2.840.113612.5.2.2.1
      X509v3 CRL Distribution Points:
         URI:http://cdp1.pca.dfn.de/dfn-pki/certification/x509/grid/g1/data/crls/root-ca-crl.crl
         URI:http://cdp2.pca.dfn.de/dfn-pki/certification/x509/grid/g1/data/crls/root-ca-crl.crl
      Authority Information Access:
         CA Issuers - URI:http://cdp1.pca.dfn.de/[...]/root-ca-cert.crt
         CA Issuers - URI:http://cdp2.pca.dfn.de/[...]/root-ca-cert.crt
  Signature Algorithm: sha1WithRSAEncryption
    b2:1d:68:d2:64:5c:9a:67:0e:83:79:42:c3:d8:5d:4b:43:8d:
    [...]
    7f:59:59:15
```

Abbildung 2.2: Inhalt eines X.509 Zertifikats

berücksichtigt werden könne, aber nicht müssen. Insbesondere muss eine unkritische Erweiterung den Komponenten nicht bekannt sein. (Zeilen 24-40, tw. gekürzt dargestellt)

Signatur des Ausstellers: Diese stellt sicher, dass alle vorher genannten Informationen in genau dieser Form vom Aussteller garantiert werden. Sind die Inhalte des Zertifikats nachträglich verändert worden, ist die Signatur ungültig und alle Ressourcen können feststellen, dass dieses Zertifikat nicht vertrauenswürdig ist (Zeilen 41-44, gekürzt dargestellt).

Die Authentifizierung in einer PKI geschieht durch den Nachweis des Besitzes des zu einem Zertifikat gehörenden privaten kryptographischen Schlüssels. Dies geschieht dadurch, dass die Ressource einen beliebigen, zufälligen Wert an den Anfragenden übermittelt. Dieser verschlüsselt den übertragenen Wert mit dem nur ihm bekannten privaten Schlüssel und überträgt das Kryptogramm wieder an die Ressource. Diese entschlüsselt das Kryptogramm mittels des öffentlichen Schlüssels aus dem Zertifikat. Wenn der entschlüsselte Wert mit dem ursprünglich gewählten Wert übereinstimmt ist die Identität sichergestellt. Somit ist bewiesen, dass die Person die im Zertifikat hinterlegte Identität besitzt. Daher ist es in einer PKI unabdingbar, dass private Schlüssel geheim gehalten werden und ausschließlich dem Zertifikatsbesitzer bekannt sind.

Das gleiche Verfahren kann gleichzeitig andersherum ablaufen, damit der Anfragende auch zweifelsfrei sicher sein kann, mit der gewünschten Ressource verbunden zu sein. In diesem Fall spricht man von *Mutual Authentication*. Um dies zu ermöglichen, gibt es nicht nur Nutzerzertifikate sondern auch Hostzertifikate. Diese enthalten als Common Name nicht den Namen eines Nutzers sondern den Fully Qualified Domain Name (FQDN, z. B. `gramd1.d-grid.uni-hannover.de`) der Grid-Ressource.

2.2.2 Die Grid Security Infrastructure

Der Einsatz einer Authentifizierungs- und Autorisierungsinfrastruktur ist im Grid Computing besonderen Anforderungen unterworfen:

- Single Sign-On (SSO) für Benutzer
- Delegation von Rechten an Grid-Ressourcen

Die im Grid Computing eingesetzte Grid Security Infrastructure setzt diese Anfoderungen unter Verwendung einer X.509 Public Key Infrastructure um, die allerdings um im Folgenden erläuterte Funktionen erweitert ist.

Single Sign-On für Benutzer ermöglicht die Bearbeitung eines Jobs im Grid, welcher nach dem Konzept des „Fire and Forget“ abgegeben wurde. Der Nutzer authentifiziert sich einmal im Grid-System und initiiert daraufhin einen Grid Job. Dieser durchläuft dann ohne weitere Nutzerinteraktion mehrere parallele und konsekutive Schritte. Da bei jedem neuen Schritt neue Ressourcen eingebunden werden, muss jedes Mal eine neue Authentifizierung stattfinden. Da weitere Nutzerinteraktionen (d. h. i. d. R. die Eingabe eines Passwortes) hier unerwünscht und schwierig umzusetzen wären, muss durch ein geeignetes Single Sign-On sichergestellt sein, dass der Nutzerlogin bei Abschicken des Jobs persistent erhalten bleibt. Hierzu dient in der Grid Security Infrastructure (GSI) ein *User Proxy* [26], also ein Stellvertreter, welcher den Nutzer im Grid repräsentiert. Der Aufbau von Proxycredentials ist in Kapitel 2.4.1 beschrieben.

Zur Authentifizierung eines Nutzers über ein Proxycredential auf einer Grid-Ressource müssen mehrere Voraussetzungen erfüllt sein: Die ausstellende CA des Nutzerzertifikats muss auf der Grid-Ressource als vertrauenswürdige CA konfiguriert sein. Anschließend wird die Zertifikatskette aus dem Proxycredential validiert. Das heißt, die Zertifikatskette muss vom CA Root-Zertifikat bis zum letzten Proxyzertifikat vollständig vorhanden sein und alle Signaturen müssen gültig sein. Da auch hierfür kein privater Schlüssel vom anfragenden System an das Zielsystem übertragen wird, muss das anfragende System noch den Besitz des zum letzten Proxyzertifikat gehörenden privaten Schlüssels nachweisen. Hierzu wird ein dem regulären SSL-Handshake [15] entsprechendes Verfahren eingesetzt.

Ein Dienst, der die Bereitstellung von Proxycredentials auf Grid-Ressourcen ermöglicht, ist MyProxy [46]. Dies ist beispielsweise dann sinnvoll, wenn ein Grid-Credential abläuft, während ein Job noch aktiv ist. Die Grid-Ressource kann sich von MyProxy ein neues Credential abholen, wenn sie hierfür autorisiert ist. Ein anderer Anwendungsfall ist der Einsatz von Clientwerkzeugen, die die GSI nicht unterstützen. Es kann ein Grid-Credential auf dem MyProxy Service hinterlegt werden und ein Grid-Dienst kann mittels der vom Client erhaltenen Logindaten dieses Proxycredential beziehen und somit im Namen des Nutzers im Grid agieren.

Die Umsetzung der Delegation von Nutzerrechten in der Grid Security Infrastructure wird in Kapitel 2.4.1 beschrieben.

2.3 Autorisierung in verteilten Systemen

Nach der Authentifizierung eines Nutzers muss eine Ressource im Grid entscheiden, ob die angefragte Aktion von diesem Subjekt ausgeführt werden darf. Dieser Vorgang ist die *Autorisierung*. Für diese Entscheidung sind drei Informationen notwendig:

- Informationen zum *Subjekt*, welches die Anfrage ausführt
- Informationen zum *Objekt*, auf welches zugegriffen werden soll
- und Informationen, welche die Zugriffsregeln enthalten, so genannte *Policies*

Das Subjekt einer Anfrage ist die Entität, welche die Anfrage initiiert. Eine Anfrage kann direkt vom Subjekt ausgehen, oder von dritten Subjekten, die im Namen oder mit den Rechten im Auftrag des ursprünglichen Subjekts agieren.

Das Objekt einer Anfrage ist die Entität, auf die zugegriffen werden soll. Dies kann eine abstrakte Ressource wie „Rechenzeit“ oder eine konkrete Ressource wie eine spezifische Datei sein. Unterschiedliche Zugriffsarten (z. B. Lesen und Schreiben) können als Bestandteil des Objektes aufgefasst werden. So kann eine Datei A als Objekt „Lesen von Datei A“ und als Objekt „Schreiben von Datei A“ auftreten.

Policies beinhalten die Regeln, welche die erlaubten Zugriffe spezifizieren. Sie bestehen aus einem Subjekt, einem Objekt sowie dem Autorisierungsstatus dieser Kombination (z. B. „erlauben“ oder „verbieten“).

2.3.1 Architekturen

Autorisierungsarchitekturen lassen sich nach RFC 2904 [58] in drei Grundtypen unterteilen. Diese werden abhängig von der verwendeten Zusammenschaltung der verschiedenen Komponenten einer derartigen Architektur Push, Pull und Agent genannt. In dieser modellhaften Betrachtung werden folgende Komponenten unterschieden:

Das Subjekt einer Anfrage, d. h. ein Nutzer oder ein IT-System. Das Subjekt initiiert die Anfrage und muss die notwendige Autorisierung besitzen.

Die angefragte Ressource wird durch die Authentifizierungs- und Autorisierungsinfrastruktur vor unerlaubten Zugriffen geschützt.

Der AAA Service beinhaltet Komponenten, die für die Authentifizierung und Autorisierung notwendig sind. AAA steht für Authentication, Authorization und Accounting.

Auf diese Komponenten verteilen sich logische Subsysteme, die spezielle Funktionen während des Autorisierungsvorgangs übernehmen:

Policy Enforcement Point (PEP): Am PEP werden Zugriffsentscheidungen durchgesetzt. Dies bedeutet also entweder, dass ein Zugriff gestattet und damit ausgeführt, oder dass er abgelehnt wird. Die eigentliche Zugriffsentscheidung findet hier nicht statt.

Policy Decision Point (PDP): Der PDP ist die Komponente, die, basierend auf den zur Verfügung stehenden Regeln (Policies, siehe PAP) und den Attributen (siehe PIP), eine Zugriffsentscheidung trifft. Diese lautet in der Regel „Allow“ oder „Deny“.

Policy Information Point (PIP): Der PIP stellt dem PDP Informationen bereit, anhand derer er die Zugriffsentscheidung berechnet. Dies sind in der Attributbasierten Autorisierung die Attribute. Diese kann ein PIP aus dem Environment beziehen (z. B. die Uhrzeit oder den Hostnamen). Attribute zum Subjekt bezieht ein PIP beispielsweise aus mit der Anfrage erhaltenen Attribute Assertions.

Policy Administration Point (PAP): Am PAP werden die Policies administriert und verwaltet.

Die Verteilung dieser konzeptionellen Elemente auf die tatsächlichen Subsysteme einer Autorisierungsinfrastruktur hängt vom jeweiligen Anwendungsfall ab.

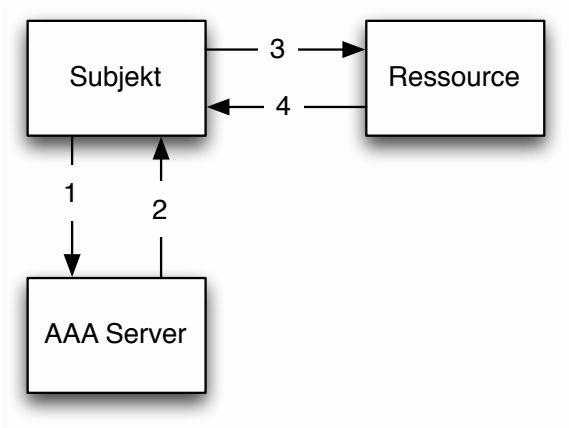

Abbildung 2.3: Das Push-Verfahren

Das Push-Verfahren

Beim Push-Verfahren wird die Autorisierungsinformation vom Subjekt der Anfrage beim Autorisierungsdienst abgefragt. Dieser stellt ein Token aus, das der Nutzer bei der angefragten Ressource vorzeigt. Diese gewährt den Zugriff wenn das Token valide ist. Die Schritte sind in Abb. 2.3 dargestellt. In Schritt eins und zwei fragt das Subjekt beim AAA Server ein Token ab, welches die Autorisierungsentscheidung enthält. Dies wird mit der Anfrage zur Ressource übertragen (push, Schritt 3) und das Subjekt erhält die Antwort (Schritt 4).

Das Pull-Verfahren

Beim Pull-Verfahren fragt die angefragte Ressource beim Autorisierungsdienst nach, ob eine Anfrage legitim ist. Stimmt dieser zu, wird der Zugriff gewährt. Das Vorgehen ist in Abb. 2.4 dargestellt. Im ersten Schritt fragt das Subjekt eine Ressource an. Die Ressource kontaktiert den AAA-Server und bezieht von ihm eine Autorisierungsentscheidung (pull, Schritte 2 und 3). Im vierten Schritt wird das Ergebnis an das Subjekt zurückgeliefert.

Das Agent-Verfahren

Beim Agent-Verfahren kommuniziert das Subjekt nur mit dem Autorisierungsdienst. Wenn eine Anfrage legitim ist, führt der Autorisierungsdienst diese auf der Ressource aus und leitet das Ergebnis an das Subjekt weiter. Das Vorgehen ist in Abb. 2.5

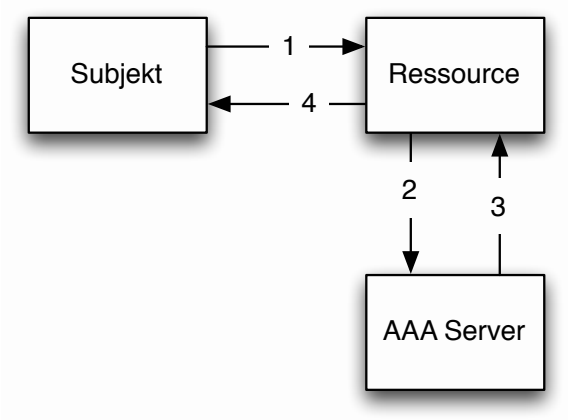

Abbildung 2.4: Das Pull-Verfahren

Abbildung 2.5: Das Agent-Verfahren

dargestellt. Zuerst kontaktiert das Subjekt den AAA-Server und teilt ihm mit, auf welche Ressource er zugreifen möchte. Wenn der AAA-Service diesen Zugriff autorisiert, wird die Ressource angefragt (Schritte 2 und 3). Das Ergebnis der Anfrage wird vom AAA-Server in Schritt 5 an das Subjekt übermittelt.

Übermittlung von Attributen

Die in RFC 2904 definierten Zugriffsmuster sind auf die Verbreitung von Autorisierungsentscheidungen zugeschnitten. Das heißt, der Policy Decision Point befindet sich im AAA Service. In dieser Arbeit ist die genaue Position der PDPs nicht relevant. Es wird angenommen, dass sie eng mit der Ressource gekoppelt sind oder geeignete Wege eingesetzt werden, um entfernte PDPs einzubinden (z. B. über XACML [44]). Der Schwerpunkt dieser Arbeit liegt in der vertrauenswürdigen Bereitstellung von Attributen zur Autorisierung. Die Attribute können analog über die drei hier vorgestellten Systeme übertragen werden. Die Attribute Authority nimmt hierfür den Platz des AAA-Services ein. Beim Push-Verfahren werden die Attribute vom Subjekt einer Anfrage von der Attribute Authority abgerufen und anschließend mit

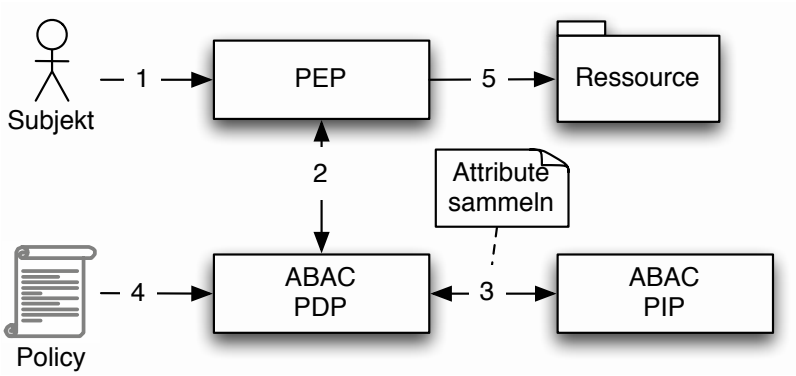

Abbildung 2.6: ABAC Autorisierungsarchitektur

der Anfrage an die Ressource verschickt. Das Pull-Verfahren sieht vor, dass eine Anfrage keine Nutzerattribute enthält, sondern die Ressource nach der Authentifizierung des Subjekts Attribute bei der Attribute Authority abruft. Beim Agent-Verfahren übermittelt das Subjekt die Anfrage an die Attribute Authority. Diese erweitert die Anfrage um Nutzerattribute und kontaktiert die gewünschte Ressource.

2.3.2 Attributbasierte Autorisierung

Die attributbasierte Autorisierung (Attribute Based Access Control, ABAC [60]) ist eine verallgemeinerte Form der rollenbasierten Autorisierung (Role Based Access Control, RBAC [51]). Hierbei werden auf geeignete Art und Weise Attribute für den Zugriff auf eine Ressource zur Verfügung gestellt. Anhand dieser Attribute kann nun entschieden werden, ob der Zugriff rechtmäßig ist oder nicht. Der prinzipielle Aufbau ist in Abb. 2.6 dargestellt.

Ein Subjekt, also ein Benutzer oder eine andere Komponente der Infrastruktur, versucht auf eine Ressource zuzugreifen (Schritt 1). Ist diese Ressource zugriffsbeschränkt, wird dieser Zugriff von einem Policy Enforcement Point (PEP) abgefangen und erst nach einer Autorisierungsentscheidung entweder verworfen oder ausgeführt. Hierzu ruft der Policy Enforcement Point einen Policy Decision Point auf (Schritt 2), der die eigentliche Entscheidung fällt. Dazu benötigt der Policy Decision Point zum einen Informationen (d. h. Attribute) zu dem anfragenden Subjekt, der angefragten Ressource und dem Environment. Diese erhält der Policy Decision Point von einem oder mehreren Policy Information Points (PIP) (Schritt 3). Zum anderen benötigt

der Policy Decision Point die Policy, welche die Zugriffsregeln enthält (Schritt 4). Anhand der Regeln und der Attribute berechnet der Policy Decision Point die Autorisierungsentscheidung und liefert diese an den Polciy Enforcement Point. Dieser setzt die Entscheidung um und erlaubt bei positiver Entscheidung den Zugriff auf die Ressource (Schritt 5).

Grundlage jeder Zugriffsentscheidung in ABAC sind die Subjekte S, die Ressourcen R und die Environments E. Diesen sind jeweils Attribute zugeordnet:

- Die Attribute des Subjekts: $SA_k (1 \leq k \leq K)$
- Die Attribute der Ressource: $RA_l (1 \leq l \leq L)$
- Die Attribute der Environments: $EA_m (1 \leq m \leq M)$

Die Zuweisung der Attribute an Subjekte s, Ressourcen r und Environments e geschieht durch folgende Relationen:

- $ATTR(s) \subseteq SA_1 \times SA_2 \times \ldots \times SA_K$
- $ATTR(r) \subseteq SR_1 \times SR_2 \times \ldots \times SR_L$
- $ATTR(e) \subseteq SE_1 \times SE_2 \times \ldots \times SE_M$

Eine Zugriffsentscheidung wird durch eine Policyregel gesteuert. Diese Regel ist eine boolesche Funktion $can\_access()$ über die Attribute des Subjekts, der Ressource und des Environments (Formel 2.1).

$$Rule : can\_access(s, r, e) \leftarrow f(ATTR(s), ATTR(r), ATTR(e)) \tag{2.1}$$

Eine konkrete einfache Policyregel hat in ABAC die Form der Formel 2.2.

$$\begin{aligned} &R1 : can\_access(s, r, e) = \\ &[s.Role = \text{`}Manager\text{'}] \wedge [r.Name = \text{`}ApprovePurchase\text{'}] \end{aligned} \tag{2.2}$$

In diesem Fall darf jedes Subjekt mit dem Attribut $Role = \text{`}Manager\text{'}$ auf Ressourcen mit dem Attribut $Name = \text{`}ApprovePurchase\text{'}$ zugreifen.

Eine weitere Regel, die die Vorteile von ABAC gegenüber RBAC zeigt, ist in Formel 2.3 gegeben. Sie legt fest, dass jedes Subjekt, repräsentiert durch das Attribut „Name“, auf eine Ressource zugreifen kann, wenn sie ein Attribut „Owner“ besitzt, welches den gleichen Wert hat. Die Angabe eines expliziten Wertes ist somit nicht in allen ABAC-Regeln notwendig.

$$R2 : can\_access(s, r, e) = [s.Name == r.Owner] \tag{2.3}$$

Dies ist flexibler und mit weniger Regeln umzusetzen als bei Role Based Access Control. Dort muss eine solche Regel für jede Ressource geschrieben werden, da die Objekte in RBAC keine entsprechenden Attribute haben.

Die Attribute in einer ABAC-Infrastruktur werden von einer *Attribute Authority* verwaltet. Sie stellt die von ihr verwalteten Attribute auf geeignete Art und Weise in Form von *Attribute Assertions* zur Verfügung, welche eine Menge von Attributen einer Entität enthält. Diese Attribute Assertion wird im PIP einer Ressource ausgewertet und die enthaltenen Attribute dem PDP zur Verfügung gestellt.

2.3.3 Kombinierungsalgorithmen für Autorisierungspolicies

In einer Autorisierungspolicy können mehrere Regeln auf eine Anfrage zutreffen. Es ist also ein Algorithmus notwendig, der aus der Anzahl der zutreffenden Regeln eine einzige macht, die dann angibt, ob ein Zugriff erlaubt oder verboten ist. Drei Algorithmen stehen hier zur Auswahl und es ist vom Anwendungsfall abhängig, welcher zu wählen ist.

First Applicable

Hierbei werden die Regeln der Reihe nach auf Anwendbarkeit auf den zu überprüfenden Zugriff getestet. Sobald eine passende Regel gefunden wird, wird sie angewendet und nicht nach weiteren passenden Regeln gesucht. Dies ist besonders bei großen Policymengen der performanteste Ansatz, da nicht immer alle Regeln untersucht werden müssen.

Der Nachteil ist, dass die Reihenfolge der Regeln relevant ist. So muss beispielsweise in einem Szenario, in dem eine Gruppe autorisiert ist, auf ein Objekt zuzugreifen, aber einzelne Gruppenmitglieder auf einer Blacklist stehen, sichergestellt sein, dass die Blacklist vor der Gruppenregel in der Policy steht und somit auch zuerst evaluiert wird.

Deny Overrides

Der Deny Overrides Algorithmus evaluiert alle Regeln, und prüft, ob mindestens eine Regel vorhanden ist, die den Zugriff erlaubt und dass keine einzige Regel den

Zugriff verbietet. Wenn keine passende Regel oder eine verbietende Regel gefunden wird, wird die Anfrage abgelehnt.

In diesem Algorithmus ist die Reihenfolge der Regeln irrelevant. Dafür müssen aber für jede akzeptierte Anfrage alle Regeln durchsucht werden, da die letzte Regel in der Policy die Entscheidung noch ändern kann. Wird bereits vorher eine verbietende Regel gefunden, kann die weitere Evaluation der Regeln abgebrochen werden.

Allow Overrides

Der Allow Overrides Algorithmus ist der inverse Ansatz zum zuvor ausgeführten Deny Overrides Algorithmus. Hierbei wird jede Anfrage zugelassen, für die mindestens eine Regel existiert, die den Zugriff erlaubt. Trifft keine Regel zu oder existieren nur ablehnende Regeln zu der aktuellen Anfrage, wird der Zugriff abgelehnt.

In diesem Ansatz ist die Reihenfolge der Regeln irrelevant. Dafür muss für jede abgelehnte Anfrage die gesamte Policymenge evaluiert werden. Sobald eine akzeptierende Regel gefunden ist, kann die weitere Evaluation abgebrochen werden und die Anfrage wird akzeptiert.

2.4 Autorisierungstechnologien im Grid

Aufbauend auf den Grundlagen der Autorisierung in verteilten Systemen werden im Folgenden die spezifischen Ausprägungen dieser Konzepte im Grid Computing erläutert.

2.4.1 Delegation von Rechten

Im Grid Computing ist die Delegation der Rechte vom Nutzer an Gridkomponenten notwendig, da Gridkomponenten mit den Rechten des Nutzers auf andere Komponenten zugreifen müssen. Das einfachste Beispiel hierfür ist der Zugriff auf Eingabedaten für einen Compute Job. Da diese auf einer Speicherressource im Grid liegen, muss die Computeressource, die mit diesen Daten arbeiten soll, diese mit den Rechten des Nutzers abrufen können.

Sowohl der Single Sign-On Mechanismus als auch die Delegation von Nutzerrechten ist mit einer herkömmlichen X.509 Public Key Infrastruktur nicht möglich. Der Nachweis der Identität des Nutzers erfolgt dabei nämlich über den Nachweis

des Besitzes des zu einem Zertifikat gehörenden privaten Schlüssels. Dieser kann jedoch nicht einfach einer Gridkomponente übergeben werden, da die Weitergabe des privaten Schlüssels automatisch dazu führt, dass das Zertifikat als unzuverlässig angesehen werden muss. Es wird somit von der Zertifizierungsstelle zurückgezogen und kann nicht mehr verwendet werden. Daher wurde der X.509 PKI Standard zur *Grid Security Infrastructure* (GSI) [27] erweitert.

Um die Delegation zu realisieren werden in der GSI X.509 Proxyzertifikate eingesetzt, die in RFC 3820 [56] spezifiziert sind. Proxyzertifikate werden von X.509 Zertifikaten abgeleitet, wie reguläre End Entity Zertfikate (d. h. Nutzer- oder Hostzertifikate) von d CA-Zertifikaten abgeleitet werden. Dies geschieht dadurch, dass der Distinguished Name des ausstellende Zertifikat als `Issuer` in das neu ausgestellte Zertifikat aufgenommen wird und das ausgestellte Zertifikat mit dem privaten Schlüssel der CA signiert wird. Um ein Zertifikat als CA-Zertifikat zu kennzeichnen, besitzt es eine kritische Erweiterung mit dem Inhalt `CA = true`. Nur so gekennzeichnete Zertifikate dürfen in einer X.509 PKI verwendet werden, um neue Zertifikate auszustellen.

Nutzerzertifikate sind durch die kritische Zertifikatserweiterung `CA = false` als ungeeignet für die Ausstellung neuer abgeleiteter Zertifikate gekennzeichnet. Die Verwendung von Proxyzertifikaten hebt diese Beschränkung unter bestimmten Voraussetzungen auf.

Bei der Erstellung eines Proxyzertifikates wird auf dem Rechner des Zertifikatbesitzers ein neues Schlüsselpaar erzeugt. Der öffentliche Schlüssel wird mit dem um einen `CN = <Zufallszahl>` erweiterten Distinguished Name des Nutzers sowie einer für Proxyzertifikate typischen kritischen Zertifikatserweiterung mit Hilfe des dem Nutzerzertifikat zugehörigen privaten Schlüssels signiert und ergibt somit ein Proxyzertifikat. Die kritische Zertifikatserweiterung kann weitere Einschränkungen enthalten wie die Angabe, dass von diesem Proxyzertifikat kein weiteres abgeleitet werden darf. Das neue Proxyzertifikat hat einen Distinguished Name der Form `/C=DE/O=GridGermany/OU=Leibniz Universitaet Hannover/OU=RRZN/CN=Ralf Groeper/CN=846285572`. Bei einem Proxyzertifikat muss somit der Distinguished Name immer dem Distinguished Name des Issuers erweitert um einen Common Name mit Zufallszahl entsprechen.

Das Proxyzertfikat entspricht technisch einem regulären X.509 Zertifikat, wel-

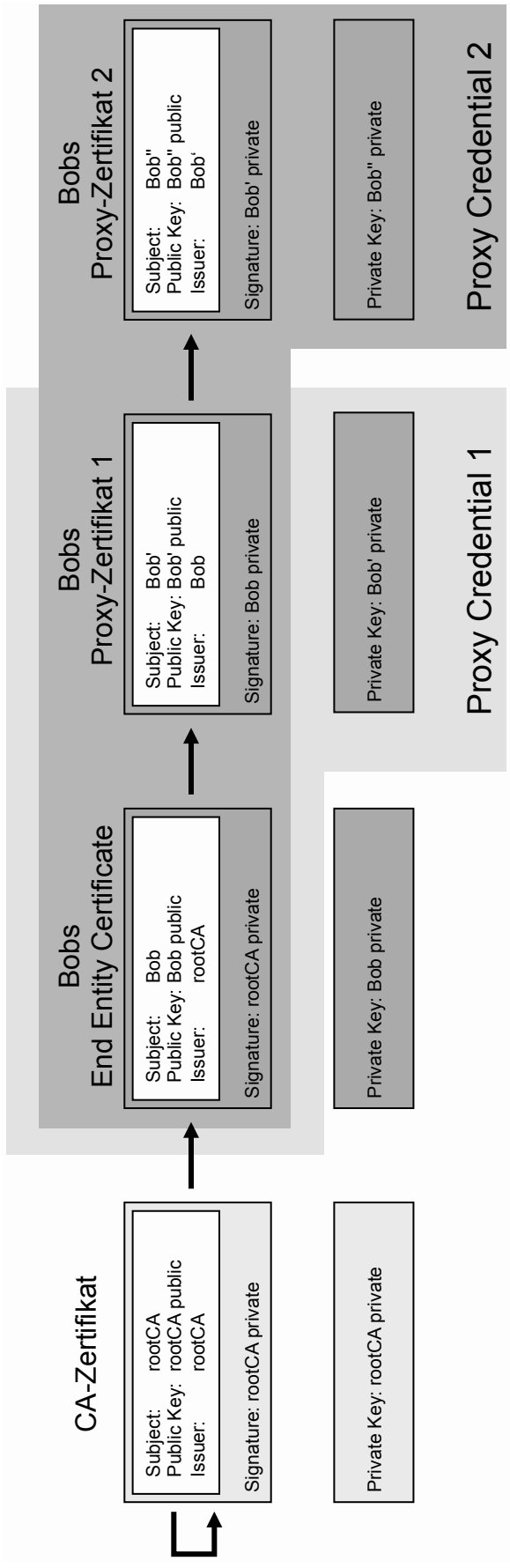

Abbildung 2.7: Bestandteile von Proxycredentials

ches allerdings von Komponenten, die die GSI nicht unterstützen, abgelehnt werden muss, da es von einem Zertifikat (dem Nutzerzertifikat) abgeleitet ist, welches `CA = false` enthält. GSI-kompatible Ressourcen akzeptieren das Zertifikat, wenn oben genannte Bedingungen erfüllt sind. Der Nutzer hat nun ein Credential, welches über einen eigenen privaten Schlüssel verfügt, der aber nicht den privaten Schlüssel seines Nutzerzertifikates kompromittiert. Dieses kann im Grid eingesetzt werden. Bei jeder weiteren Delegation wird dieser Vorgang wiederholt, so dass die Zertifikatskette jedes Mal um eins länger wird und gemeinsam mit dem privaten Schlüssel des letzten Proxyzertifikates das Proxycredential bildet.

In Abb. 2.7 ist eine solche Zertifikatskette für zwei Proxyzertifikate dargestellt. Das CA-Zertifikat ist nicht Bestandteil der Proxycredentials, da es als Wurzel des Vertrauens auf den Grid-Ressourcen vorinstalliert sein muss. Auf der ersten beteiligten Grid-Ressource liegt das Proxycredential 1 vor. Es besteht aus Bobs Nutzerzertifikat, Bobs Proxyzertifikat 1 und dem privaten Schlüssel, der zu dem öffentlichen Schlüssel in Proxyzertifikat 1 gehört. Proxyzertifikat 1 ist mit dem privaten Schlüssel signiert, der zu Bobs Nutzerzertifikat gehört.

Wenn eine weitere Grid-Ressource im Namen des Nutzers angesprochen wird, liegt auf dieser nach erfolgter Delegation das Proxycredential 2 vor. Dieses besteht aus Bobs Nutzerzertifikat, dem Proxyzertifikat 1 und dem Proxyzertifikat 2. Das Proxyzertifikat 2 ist mit dem privaten Schlüssel signiert, welcher zu dem Proxyzertifikat 1 gehört und auf der vorhergehenden Grid-Ressource liegt. Hinzu kommt noch der private Schlüssel, der zu Proxyzertifikat 2 gehört.

Wenn das Ableiten eines Proxycredentials nicht lokal auf einem Rechner geschieht, sondern die Nutzerrechte über ein Proxycredential an eine entfernte Grid-Ressource delegiert werden sollen, ist dies möglich, ohne den privaten Schlüssel des Proxycredentials (oder irgendeinen anderen privaten Schlüssel) zu übertragen. Dazu wird auf dem Zielsystem ein Schlüsselpaar generiert. Der öffentliche Schlüssel wird als Certificate Signing Request (CSR) an das delegierende System gesendet. Dieses ergänzt die Metadaten (Distingushed Name usw.) und signiert diesen CSR. Somit wird aus dem Certificate Signing Request ein Zertifikat. Dieses wird – zusammen mit der gesamten Ableitungskette aus Zertifikaten und Proxyzertifikaten – zurück an das Zielsystem gesendet. Dieses verfügt somit über ein gültiges Proxycredential (d.h. eine Zertifikatskette mit einem neuen Proxyzertifikat am Ende sowie dem zu diesem Zertifikat gehörenden privaten Schlüssel), ohne dass ein privater Schlüssel über ein

Netzwerk übertragen wurde.

Durch diesen Mechanismus wird das Single Sign-On realisiert, da der Nutzer keine Authentifizierungsinformationen eingeben muss, wenn eine Grid-Ressource in seinem Namen auf weitere Ressourcen zugreift. Gleichzeitig werden die Rechte des Nutzers in Form seiner Identität in das Grid delegiert, da jede Ressource, die über ein Proxy-credential des Nutzers verfügt, mit dessen Rechten auf weitere Grid-Ressourcen zugreifen kann.

2.4.2 Identitätsbasierte Autorisierung

Der einfachste Fall der Autorisierung im Grid ist gleichzeitig der in aktuellen Grid-Infrastrukturen gebräuchlichste. Hierbei wird der Nutzer anhand seiner Identität, repräsentiert durch seinen X.509 Distinguished Name (DN), autorisiert. Die Policy besteht somit aus einer Liste aller Distinguished Names, die zu den berechtigten Nutzern gehören. Dieser Ansatz skaliert schlecht für sehr große Nutzermengen, da auf jeder Grid-Ressource alle Nutzer über ihren Distinguished Name bekannt sein müssen, unabhängig davon, ob sie tatsächlich jemals auf diese Ressource zugreifen. Der DN neuer Nutzer muss auf allen Grid-Ressourcen bekanntgemacht werden, auf denen der Nutzer Zugriff hat. Nutzergruppen oder Rollen können nicht berücksichtigt werden.

2.4.3 Autorisierung auf Basis Virtueller Organisationen

Das Management Virtueller Organisationen beinhaltet Informationen zu Nutzern, die für die Autorisierung auf Grid-Ressourcen relevant sind. Dies gilt insbesondere für Ressourcen, die von einem Ressourcenprovider für eine VO bereitgestellt werden oder Ressourcen, die der VO gehören. Es kann sich hierbei im bereitgestellte Hardwareressourcen handeln, häufiger sind aber Ressourcen wie Daten und Softwarelizenzen Eigentum von Virtuellen Organisationen. Wenn der Zugriff auf diese Ressourcen nur Mitgliedern der Virtuellen Organisation gestattet ist, ist die reine Mitgliedschaft in der Virtuellen Organisation die Information, die vorgelegt werden muss, um Zugriff zu erhalten. Innerhalb der Virtuellen Organisation gibt es noch feingranularere Unterscheidungen nach Rollen und Eigenschaften der VO-Mitglieder. Beispielsweise kann ein VO-Mitglied mit der Rolle „VO Software Manager“ weitergehende Schreibrechte auf VO-Daten und Software haben, als ein reguläres Mitglied.

Diese Informationen sind relevant für die Autorisierung auf Basis Virtueller Organisationen und das VO-Manamgementsystem tritt als Attribute Authority auf, welche diese Informationen in Form von Attributen zur Verfügung stellt.

2.4.4 Attribute Certificates

Attribute Certificates nach RFC 3281 [19] dienen der Übermittlung von Attributen von einer Attribute Authority an eine Ressource, die mit diesen Attributen Autorisierungsentscheidungen treffen muss. Die Kodierung dieser Attribute Certificates ist angelehnt an X.509 Zertifikate. Sie enthalten somit eine Menge von ASN.1-kodierten Informationen, welche durch eine abschließende Signatur gegen Veränderung geschützt sind. Außerdem kann durch die Signatur zweifelsfrei nachgewiesen werden, dass die Attribute von der angegebenen Attribute Authority ausgegeben wurden und nicht von einer dritten Stelle. Der Inhalt eines an ein Proxyzertifikat gebundenes Attribute Certificates ist in Abb. 3.4 dargestellt.

2.5 SAML und Shibboleth

2.5.1 Die Security Assertion Markup Language

Wie bereits erläutert stellt der Austausch von Authentifizierungs- und Autorisierungsinformationen stellt in föderierten Systemen eine besondere Herausforderung dar. Beteiligte Systeme müssen auf sichere und vertrauenswürdige Weise Informationen austauschen können. Um diesen Ansatz zu ermöglichen wurde die Security Assertion Markup Language (SAML) [48] von der *Organization for the Advancement of Structured Information Standards* (OASIS) [47] spezifiziert. SAML ist ein XML-basiertes Format, welches durch kryptographische Signaturen und Verschlüsselung sicherstellt, dass bei dem Austausch der Informationen die Integrität und Vertraulichkeit gewahrt sind.

In Abb. 2.8 ist eine SAML Assertion dargestellt, die Attribute Statements enthält. In dieser Arbeit werden SAML Assertions mit Attribute Statements kurz *SAML Attribute Assertions* genannt. Die relevanten Teile sind:

- Zeile 13: ID der Assertion
- Zeilen 16-42: Die Signatur mit Metadaten, darunter:

  - Zeile 20: gesamte Assertion mit dieser ID (aus Zeile 13) ist signiert
  - Zeilen 32-36: Wert der Signatur
  - Zeilen 37-41: Zertifikat des Issuers, also der ausstellenden Attribute Authority

- Zeilen 43-56: Identität des Subjekts, für den die Assertion gültig ist (vgl. Kapitel 3.4.3)

- Zeilen 59-70: Das eigentliche Attribut, in diesem Fall mit zwei verschiedenen Werten

Im Rahmen dieser Arbeit können Attribute Certificates und SAML Attribute Assertions als funktional äquivalent angesehen werden. Allerdings ist SAML der flexiblere Standard und auch außerhalb der Grid-Welt verbreitet und ermöglicht so in Zukunft die Interoperabilität der in dieser Arbeit erstellten Infrastruktur mit anderen Infrastrukturen, beispielsweise wenn die europäischen Gridverbünde zusammenwachsen werden. Daher werden SAML Attribute Assertions in dieser Arbeit bevorzugt eingesetzt.

2.5.2 Shibboleth

Shibboleth [35] ist ein Softwarepaket für den Aufbau föderierter Authentifizierungs- und Autorisierungsinfrastrukturen (AAI) basierend auf SAML. Viele internationale Projekte im Bereich der Authentifizierungs- und Autorisierungsinfrastrukturen verwenden Shibboleth, insbesondere die Deutsche DFN-AAI [14]. Shibboleth unterstützt organisationsübergreifendes Single Sign On (SSO) für webbasierte Systeme. Das heißt, es setzt HTTP-Server als Server bzw. Webbrowser als Clients voraus. In einer Service Orientierten Architektur, z. B. basierend auf SOAP oder auf WSRF-basierten Grid-Diensten, kann es nicht unmittelbar eingesetzt werden.

Die drei Hauptkomponenten von Shibboleth sind:

Identity Provider (IdP): Die Heimatorganisationen der Nutzer stellen jeweils einen IdP bereit. Dieser IdP ist dafür zuständig, die Metadaten der Nutzer dieser Einrichtung den Service Providern innerhalb der Föderation zur Verfügung zu stellen.

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"+
  ID="_22040814-27f5-44aa-9534-165d2aebfc0e" InResponseTo="81b1743c-abc3-4c65-855e-
7afb2e0a9148"
  IssueInstant="2009-06-10T17:08:20.321Z" Version="2.0">
   <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
       CN=host/dgridvoms.fzk.de,OU=FZK,O=GermanGrid
    </saml:Issuer>
   <Status>
     <StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
   </Status>
   <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" ID="_6205c368-ccd3-4374-
       911a-dc9944c1d54d" IssueInstant="2009-06-10T17:08:20.320Z" Version="2.0">
     <saml:Issuer>CN=host/dgrid-voms.fzk.de,OU=FZK,O=GermanGrid</saml:Issuer>
     <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
       <ds:SignedInfo>
         <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
         <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
         <ds:Reference URI="#_6205c368-ccd3-4374-911a-dc9944c1d54d">
           <ds:Transforms>
             <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
             <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
               <ec:InclusiveNamespaces xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
                       PrefixList="ds saml xs"/>
             </ds:Transform>
           </ds:Transforms>
           <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
           <ds:DigestValue>8N4xc0EnjMS5txKxWtnNyLGNI70=</ds:DigestValue>
         </ds:Reference>
       </ds:SignedInfo>
       <ds:SignatureValue>
         i/3eyz8Pj3xoNPw9itnm9vYeRWca6GE/QLXL8ufA15bWScHJAMO1Aq6b4o0Q0OHhMfsJuZ0V+M7c
         ...gekürzt...
         Pj8RwOPQ5W/DcqmSVxpBhhy7qO//sQqIGFsZmg==
       </ds:SignatureValue>
       <ds:KeyInfo>
          <ds:X509Data>
           <ds:X509Certificate>...gekürzt... </ds:X509Certificate>
          </ds:X509Data>
       </ds:KeyInfo>
     </ds:Signature>
     <saml:Subject>
       <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
              format:X509SubjectName">CN=Benjamin Henne,
              OU=RRZN,OU=Leibniz Universitaet Hannover,O=GridGermany,C=DE</saml:NameID>
       <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
         <saml:SubjectConfirmationData>
           <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
             <ds:X509Data>
                <ds:X509Certificate>...gekürzt... </ds:X509Certificate>
             </ds:X509Data>
           </ds:KeyInfo>
         </saml:SubjectConfirmationData>
       </saml:SubjectConfirmation>
     </saml:Subject>
     <saml:Conditions NotBefore="2009-06-10T17:08:20.320Z" NotOnOrAfter="2009-06-
        11T05:08:20.320Z"/>
     <saml:AttributeStatement>
       <saml:Attribute Name="http://voms.forge.cnaf.infn.it/fqan"
              NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
         format:unspecified">
         <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-
         instance xsi:type="xs:string">/education/rrzn</saml:AttributeValue>
         <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-
           instance" xsi:type="xs:string">/education</saml:AttributeValue>
       </saml:Attribute>
     </saml:AttributeStatement>
   </saml:Assertion>
</Response>
```

Abbildung 2.8: Eine signierte SAML Assertion mit Attribute Statements

Service Provider (SP): Die Service Provider-Komponente wird beliebigen, webbasierte Ressourcen installiert, die die Zugangskontrolle über Shibboleth-Mechanismen regeln sollen. Ein Nutzer, der auf eine solche Ressource zugreifen möchte, wird von dem Service Provider an seinen Identity Provider weitergeleitet, bei dem er sich authentifiziert. Der Identity Provider sichert nun dem Service Provider über kryptographisch gesicherte SAML Assertions zu, dass der Nutzer sich erfolgreich authentifiziert hat und stellt dem Service Provider optional weitere Nutzerattribute zur Verfügung. Anhand dieser Informationen fällt der Service Provider die Zugriffsentscheidung und erlaubt ggf. den Zugriff auf die Ressource.

Where Are You From (WAYF) bzw. Discovery Service (DS): WAYF (Shibboleth 1.x) bzw. DS (Shibboleth 2.x) sind föderationsweite Dienste, die eine Liste aller Identity Provider zur Verfügung stellen. Da ein Service Provider bei einem Nutzerzugriff nicht wissen kann, welcher Identity Provider für diesen Nutzer zuständig ist, leitet der Service Provider alle Nutzer an den Discovery Service weiter, von wo aus der Nutzer anhand seiner Auswahl an den zuständigen Identity Provider weitergeleitet wird.

Die Verwendung von Shibboleth setzt voraus, dass im Webbrowser des Benutzers Java Script und Cookies aktiviert sind. Wenn Java Script deaktiviert ist, müssen einige der automatisch ablaufenden Schritte manuell durch einen Mausklick ausgeführt werden. Ohne Cookies funktioniert Shibboleth nicht, es reicht aber, die Cookies nur für die aktuelle Session zu behalten. Sind Cookies auch über die Session hinaus erlaubt, wird beim nächsten Ausführen eines Logins über Shibboleth die Auswahl des Heimat Identity Providers automatisiert, d. h. der Umweg über den WAYF oder DS entfällt.

Neben den Protokollen und Schnittstellen müssen in einer Föderation auch die Namen und die Semantiken der Attribute, welche innerhalb der Föderation verwendet werden, standardisiert werden. Dies ist notwendig, damit die Service Provider die von den Identity Providern bereitgestellten Attribute einsetzen können. Die Autorisierungspolicies der Service Provider basieren auf diesen Attributen. Als vordefinierter Satz von Attributen hat sich das vom Internet2 Projekt in den USA spezifizierte eduPerson Schema [17] etabliert. Dies wird auch in der DFN-AAI, der deutschen Shibbolethföderation für Forschungs- und Lehreinrichtungen, eingesetzt

und wird daher auch in dieser Arbeit verwendet.

2.5.3 Einsatz von Shibboleth im Grid Computing

Ein von vielen Nutzern aktueller Grid-Infrastrukturen monierter Nachteil ist, dass die Einarbeitungszeit und der organisatorische Overhead bis zum ersten erfolgreichen Gridjob zu hoch sein. So müssen potentielle Nutzer zuerst ein X.509 Zertifikat erhalten und aufgeklärt werden, wie sie den dazu gehörigen privaten Schlüssel absichern müssen. Die Beantragung eines X.509 Zertifikates ist damit verbunden, persönlich unter Vorlage eines Ausweisdokumentes bei einer Registration Authority vorzusprechen. Anschließend muss der Nutzer in eine Virtuelle Organisation aufgenommen werden, wobei einige der Schritte wiederholt werden müssen.

Durch den föderierten Ansatz ist Shibboleth ein idealer Kandidat, um in Grid-Umgebungen eingesetzt zu werden und oben genannte Nachteile zu verringern. Viele Einrichtungen treten derzeit Shibboleth Föderationen bei und machen somit ihr vorhandenes Identity Management von außen erreichbar. Dies kann genutzt werden, um nicht diese bereits erhobenen Daten für die Verwendung im Grid erneut erheben zu müssen.

Virtuelle Organisationen im Grid sind üblicherweise aus organisatorisch und geographisch getrennten Personen und Ressourcen zusammengesetzt. Der Einsatz einer existierenden, nicht notwendigerweise Grid-spezifischen, Shibboleth Föderation, vereinfacht die Verwaltung von Virtuellen Organisationen, da die Metadaten der Nutzer nicht mehr neu erhoben, gepflegt und verifiziert werden müssen. Da es technisch möglich ist, durch die Identity Provider beliebige Attribute-Value-Paare (z. B. VO-Zugehörigkeit, Rolle in der VO) zu verwalten, kann das VO-Management theoretisch sogar vollständig innerhalb einer Shibboleth Föderation durchgeführt werden. In einer solchen Infrastruktur nehmen die Grid-Ressourcen die Rolle von Shibboleth Service Providern ein.

Wenn eine existierende, nicht gridspezifische Shibboleth Föderation eingesetzt wird, ist es üblicherweise nicht möglich, Grid-spezifische Attribute in dieser Föderation zu verwalten. Es ist also ein weiterer, dedizierter Dienst notwendig, um die gridspezifischen Attribute zu verwalten, während die Stammdaten der Nutzer aus der Shibboleth Föderation verwendet werden können. Dies ist mit den verfügbaren Lösungen für das VO-Management nicht möglich, es gibt allerdings bereits einige Projekte, die sich mit diesem Problem befassen und in Kapitel 3.5 analysiert werden.

Kapitel 3

Analyse von attributbasierten Autorisierungsfunktionen im Grid Computing

3.1 Übersicht

Basierend auf den Erfahrungen in der D-Grid Infrastruktur hat sich gezeigt, dass im Grid Bedarf an attributbasierten Autorisierungsfunktionen besteht [29]. Die bisherige identitätsbasierte Autorisierung skaliert schlecht für viele Nutzer und Ressourcen und die feingranulare Vergabe von Rechten im Grid ist praktisch unmöglich.

Ein häufiger Anwendungsfall ist die Ausweitung der Zugriffsrechte basierend auf einer Rolle, die ein Nutzer in seiner Virtuellen Organisation inne hat. Als Softwareadministrator muss er beispielsweise im VO-weiten Softwareverzeichnis ein Update der VO-Software installieren können. Reguläre Nutzer der VO dürfen diese Software nur ausführen, aber nicht verändern. Dies kann durch attributbasierte Autorisierung sehr einfach abgebildet werden. Bei identitätsbasierter Autorisierung ist dies nur mit sehr hohem administrativen Aufwand möglich und Nutzer mit der entsprechenden Rolle können ausschließlich als VO-Softwareadministratoren auf das Grid zugreifen. Zur Vermeidung von unbeabsichtigten Veränderungen an der VO-Software ist dies aber nur sinnvoll, wenn auch administrative Funktionen ausgeführt werden müssen. Bei regulärer Nutzung des Grids möchten diese Nutzer nur mit den normalen Nutzerrechten auf das Grid zugreifen. Dies kann mit identitätsbasierter Autorisierung nicht realisiert werden.

Durch die organisatorische Verteilung und lose Kopplung der beteiligten Partner in einem Gridverbund stellt ein solches Szenario neuartige Anforderungen an die Authentifizierungs- und Autorisierungsinfrastruktur, welche diese Funktionen bietet. Die formalen Grundlagen einer solchen Infrastruktur, entstehende Vertrauensprobleme und entsprechende Lösungen werden in diesem Kapitel diskutiert. Anschließend werden existierende Softwarelösungen für die attributbasierte Autorisierung im Grid Computing analysiert und, basierend auf den Erkenntnissen der Modellierung, bewertet. Darauf aufbauend wird eine Gesamtarchitektur entwickelt, welche die Anforderungen nach attributbasierter Autorisierung mit mehreren Attribute Authorities unterstützt, ohne durch die organisatorische Verteilung aller beteiligten Komponenten die vorher identifizierten Vertrauensprobleme zu verursachen. Der Fokus liegt hierbei auf der vertrauensvollen Übermittlung von Attributen von den Attribute Authorities an die entsprechenden Grid-Ressourcen, die mittels dieser Attribute eine Autorisierungsentscheidung durchführen sowie auf den Aspekten der eigentlichen Autorisierung, welche von der Verwendung mehrerer Attribute Authorities betrof-

fen sind. Weitere Aspekte der eigentlichen Autorisierung auf den Ressourcen sowie die Erstellung und Pflege der Autorisierungspolicies sind nicht Gegenstand dieser Arbeit.

Die Entwicklung domänenspezifischer Gridsoftware setzt den korrekten Einsatz der Funktionen einer Grid-Infrastruktur in den Clientprogrammen und Fachdiensten voraus. Es hat sich gezeigt, dass insbesondere die korrekte Einbindung der Authentifizierungs- und Autorisierungsfunktionen Softwarentwickler ohne Erfahrung mit den Konzepten des Grid Computing verleitet, fehlerhafte Software zu entwickeln oder dass sie Probleme damit haben, aus ihrer Software heraus überhaupt Grid-Dienste mit korrekter Behandlung der Sicherheitsfunktionen anzusprechen. Dies tritt bereits im vergleichsweise simplen Fall der identitätsbasierten Autorisierung auf. Eine Authentifizierungs- und Autorisierungsinfrastruktur, die die in dieser Arbeit behandelten attributbasierten Autorisierungsfunktionen unterstützt, wird diese Hürde noch erhöhen. Daher wird ein Konzept erarbeitet, welches ein vereinfachtes Interface beinhaltet, dessen Ziel es ist, die Komplexität des Grids, insbesondere der Authentifizierungs- und Autorisierungsinfrastrukturen, zu verbergen. Zusätzlich dient das Interface dazu, die Eigenschaften einer spezifischen Grid-Infrastruktur wie dem D-Grid vor dem Entwickler zu verbergen. So können beispielsweise die URLs zentraler Dienste vorkonfiguriert werden und müssen nicht vom Entwickler der domänenspezifischen Software nachgeschlagen und korrekt eingesetzt werden.

3.2 Quellen von Attributen für die Autorisierung im Grid Computing

In verteilten Infrastrukturen mit attributbasierter Autorisierung kann es beliebig viele Attribute Authorities geben, die diese Attribute verwalten und Attribute Assertions mit diesen Attributen ausstellen. Im Grid Computing haben sich zwei Arten von Quellen für Attribute etabliert. Die eine ist die Heimatorganisation des Nutzers, die seine *Campusattribute* ausstellt, die andere ist die Virtuelle Organisation, in welcher der Nutzer Mitglied ist. Diese stellt *VO-Attribute* aus.

1. Campusattribute sind Nutzerattribute, die von ihrer Heimatorganisation verwaltet werden. Sie identifizieren und beschreiben den Nutzer, z. B. durch seinen Namen, seine Nationalität oder auch seine Telefonnummer. Für die Autori-

sierung relevante Informationen sind seine Institutszugehörigkeit, sein Status (bspw. Professor oder Mitarbeiter) oder das aktuelle Hören einer bestimmten Lehrveranstaltung. Campusattribute werden zunehmend durch eine deutschlandweite Shibboleth Föderation universitätsübergreifend zur Verfügung gestellt (DFN-AAI).

2. VO-Attribute beschreiben die Mitgliedschaft, Rollen und zugesicherten Fähigkeiten innerhalb einer Virtuellen Organisation. Diese Attribute werden vom VO-Manager in einem VO Management System verwaltet. Häufig verwendete Systeme sind VOMS [55] und VOMRS [20].

Da das Management Virtueller Organisationen im Grid geschieht und die Softwarewerkzeuge direkt für den Einsatz im Grid entwickelt werden, ist es hier leichter, Anforderungen komplexer gridbasierter Authentifizierungs- und Autorisierungsinfrastrukturen umzusetzen. Die Werkzeuge für das Identitätsmanagement bei den Heimatorganisationen sind dagegen nicht Grid-spezifisch und müssen die Anforderungen vieler verschiedener Nutzer dieser Systeme berücksichtigen oder zwischen unvereinbaren Anforderungen abwägen. Dort ist es somit ungleich schwerer, gridspezifische Anforderungen umzusetzen. Es wird daher in dieser Arbeit davon ausgegangen, dass die Shibboleth Föderation für das Grid genutzt werden kann, dass es aber nicht möglich ist, die Funktionsweise dieser zu verändern, beispielsweise durch die Einführung gridspezifischer Attribute oder zusätzlicher Softwarekomponenten bei den Identity Providern.

3.3 Stakeholder

3.3.1 Anwender

Die Anwender akademischer Grid-Infrastrukturen sind Forscher aus unterschiedlichen wissenschaftlichen Communities. Sie sehen die Grid-Infrastruktur als Service, den sie verwenden, um ihre fachlichen Ziele zu erreichen. Der innere Aufbau und die Komplexität der Infrastruktur ist für sie nicht relevant. Sie interagieren mit dem Grid nur über ihre Clientwerkzeuge. Aus Sicht dieser Anwender ist es erstrebenswert, das Grid soweit zu abstrahieren, dass sie in ihren Clientwerkzeugen nicht unterscheiden können, ob bspw. eine Berechnung lokal oder im Grid erfolgt. Sie sind nur an

der schnellstmöglichen Verfügbarkeit der Ergebnisse der Berechnung interessiert. Die Sicherheit der verwendeten Infrastruktur und der in das Grid eingebrachten Daten wird vorausgesetzt, aber die Bereitschaft für diese Sicherheit Komfortverluste in Kauf zu nehmen, ist sehr begrenzt.

3.3.2 Ressourcenanbieter

Wissenschaftlich genutzte Grid-Infrastrukturen werden hauptsächlich durch Universitätsrechenzentren bereitgestellt. So wurden im Rahmen der Förderung des D-Grid hierfür dedizierte Ressourcen wie Computecluster und Speichersysteme bereitgestellt und an den Rechenzentren der Zuwendungsempfänger betrieben. Diese Ressourcen sind für alle D-Grid Communities verfügbar und stellen den Grid Backbone dar.

Darüber hinaus gibt es noch Anbieter domänenspezifischer, VO-eigener Ressourcen. Diese werden nicht allen Nutzern der Grid-Infrastruktur zur Verfügung gestellt, sondern spezifisch einigen oder allen Nutzern einer Community. Dies wird innerhalb der Virtuellen Organisation definiert. Diese Ressourcen werden physikalisch ebenfalls häufig in Universitätsrechenzentren betrieben, die Administration insbesondere der Grid Middleware und der Zugriffsregeln liegt allerdings in Händen der Grid VO.

3.3.3 VO-Administratoren

Die Administratoren einer Virtuellen Organisation organisieren die Mitgliedschaften in der VO sowie die Zugriffsrechte auf VO-eigene Ressourcen. Dies sind meist keine Hardwareressourcen sondern logische Ressourcen wie Daten oder Softwarelizenzen. Durch die Verwaltung der Mitglieder der VO und deren VO-Attributen stellen sie eine wichtige Quelle sowohl von Autorisierungspolicies als auch von autorisierungsrelevanten Attributen dar.

3.3.4 Entwickler domänenspezifischer Gridsoftware

Die Entwickler domänenspezifischer Anwendungssoftware im Grid haben zwei Hauptaufgaben: Zum einen muss domänenspezifische Clientsoftware an das Grid angepasst werden, zum anderen müssen die ressourcenintensiven Komponenten aus der herkömmlichen Standalone-Software herausgelöst und an den Betrieb auf Grid-Ressourcen angepasst werden. Diese Entwickler sind häufig erfahrene Softwareentwickler, die aber keine Vorbildung in der Erstellung von gridspezifischen Diensten haben. Die

Einarbeitung in die entsprechenden APIs stellt hierbei eine hohe Hürde dar. Insbesondere die Konzepte im Bereich der Authentifizierung und Autorisierung sind sehr komplex. Ohne das Verständnis dieser Konzepte ist der korrekte Einsatz der Grid-APIs nicht möglich. Dies ist sowohl für die Clientsoftware also auch für gridifizierte domänenspezifische Dienste relevant, da beide in die Authentifizierungs- und Autorisierungsinfrastruktur eingebunden werden müssen.

3.4 Analyse der attributbasierten Autorisierung mit mehreren Attribute Authorities

3.4.1 Discovery Probleme

Aus der Verteilung der Komponenten in einer organisationsübergreifenden Authentifizierungs- und Autorisierungsinfrastruktur ergeben sich folgende Probleme:

Wenn Dienste bei einer Anfrage Attribute des Subjekts der Anfrage von einer Attribute Authority abfragen (sog. Attribute Pull), ergibt sich das *Identity Provider discovery Problem* [5] oder, allgemeiner ausgedrückt, das *Attribute Authority discovery Problem*. Das bedeutet, dass ein Dienst nicht a priori weiß, welche Attribute Authority für das Subjekt zuständig ist, das den Zugriff auf die Ressource anfragt. Ein Discovery Service bzw. Where Are You From Service kann nicht genutzt werden, da diese eine Nutzerinteraktion benötigen. Dieses Problem kann entweder gelöst werden, indem das Subjekt Informationen zu den für es zuständigen Attribute Authorities in die Anfrage einbettet oder indem das Subjekt alle notwendigen Attribute bereits in die eigentliche Anfrage einbettet (sog. Attribute Push).

Wenn der Nutzer alle notwendigen Attribute in die Anfrage einbettet, z. B. durch Einbinden eines Attribute Certificates oder einer SAML Attribute Assertion in das Proxyzertifikat, ergibt sich ein inverses Problem zum oben beschriebenen AA discovery problem: Das so genannte *Service discovery Problem*. Hierbei entsteht das Problem dadurch, dass ein Nutzer im Grid nicht unbedingt a priori weiß, auf welcher Ressource sein Job ausgeführt wird oder, wenn er es weiß, welche Attribute diese spezifische Ressource benötigt, um den Nutzer erfolgreich zu autorisieren. Dieses Problem kann auf zwei Arten gelöst werden: Erstens kann der Nutzer alle seine verfügbaren Attribute mitsenden, was allerdings sowohl in der Größe der Attribute Assertion als auch aus Aspekten des Datenschutzes bzw. der Datensparsamkeit Pro-

bleme bereitet. Zweitens kann innerhalb eines Grid-Verbundes ein einheitlicher Satz von Attributen vereinbart werden, der auf allen Ressourcen ausreicht, um autorisiert werden zu können.

Aufgrund der Anforderung des Single Sign-Ons kann ein Agreement-Protokoll, das dynamisch aushandelt, welche Attribute eine Ressource benötigt, hier nicht eingesetzt werden. Der Nutzer ist nach dem initiieren des Gridjobs nicht mehr verfügbar, um Attribute freizugeben bzw. seine Credentials bei der Attribute Authority einzugeben, damit seine Attribute abgefragt werden können. Die delegierten Credentials können hierzu auch nicht eingesetzt werden, da aus ihnen nicht hervorgeht, welche Attribute Authority für den Nutzer zuständig ist (siehe AA discovery Problem). Weiterhin benötigt der Shibboleth Identity Provider eine zusätzliche Softwarekomponente, um Attribute Pull von Grid-Ressourcen mittels der Grid Security Infrastructure zu ermöglichen. Dies ist wie oben beschrieben nicht praktisch umsetzbar.

Der Agent-Ansatz, bei dem die Attribute von einem zwischen Subjekt und angefragter Ressource angesiedeltem System eingefügt werden, ist im Grid Computing nicht sinnvoll einsetzbar. Dieser würde sowohl das AA discovery Problem als auch das Service discovery Problem haben, da er keine weiterführenden Informationen über den Client bzw. die Ressource besitzt. Da neben Push und Pull keine weiteren praktikablen Möglichkeiten existieren, muss eine der beiden gewählt werden und mit das entsprechende discovery Problem gelöst werden.

Aufgrund des AA discovery Problems und der Tatsache, dass keine zusätzliche Software auf den Shibboleth Identity Provider Komponenten installiert werden kann, wird der Ansatz mit Attribute Push und einem vorher definierten festen Satz von Attributen im weiteren Verlauf der Arbeit angenommen. Dieser muss von den Serviceprovidern und VO-Administratoren in den entsprechenden Gremien abgestimmt werden.

3.4.2 Trust Proxying

Ein zentrales Problem für verteilte Sicherheitsinfrastrukturen stellt die Sicherstellung der Vertrauenswürdigkeit der für eine Zugriffsentscheidung verwendeten Daten dar. In organisatorisch getrennten Teilen einer Infrastruktur kann Vertrauenswürdigkeit nicht implizit sichergestellt werden, wie es beispielsweise innerhalb eines monolithischen Softwaresystems möglich ist. In einem solchen System kann eine Infor-

mation ohne weitere Absicherung hinterlegt werden, beispielsweise indem im Programmverlauf eine Variable über den Autorisierungszustand auf *true* gesetzt wird. Solange das Programm nicht durch einen Programmierfehler oder direkten Eingriff in den Speicherbereich im Hauptspeicher des Systems kompromittiert wird, gilt dies als sicher.

Ein solches Vorgehen ist in verteilten Infrastrukturen wie dem Grid nicht möglich, da viele Organisationen genau festgelegte Autoritäten haben. Innerhalb dieser Autorität dürfen sie autorisierungsrelevante Aussagen treffen. Diese Aussagen müssen auf vertrauenswürdige Weise an andere Organisationen weitergeleitet werden. Diese Empfänger müssen zweifelsfrei die Authentizität dieser Aussagen überprüfen können. Das beinhaltet nicht nur, dass die Aussage nicht nachträglich verändert wurde, sondern auch, ob die ausstellende Organisation berechtigt ist, genau diese Aussage zu treffen.

Ein Problem, dass in einer solchen Infrastruktur auftreten kann ist *Trust Proxying* [31]. Trust Proxying tritt auf, wenn eine Zwischenstation, d. h. ein Proxy, Attribute von den eigentlichen Attribute Authorities empfängt und neu ausstellt, wie in Abb. 3.1 dargestellt. Dies bedeutet in der Regel, dass der Proxy die Attribute neu mit seinem eigenen privaten Schlüssel signiert.

Ein Trust Proxy besteht aus mehreren Komponenten, die in Abb. 3.2 dargestellt sind. Zur Beschaffung der Attribute von den originalen Attribute Authorities sind eine oder mehrere Senken notwendig, welche die Syntaxen der verschiedenen Kodierungen (z. B. SAML oder Attribute Certificates) für Attribute Assertions interpretieren können und die Signaturen überprüfen. Die Attribute werden als Attribut-Wert-Paare im Attribute Store zwischengespeichert. Sie sind in dieser Form nicht mehr an einen speziellen Syntax gebunden. Wenn Attribute benötigt werden, müssen sie nun nicht mehr bei den originalen Attribute Authorities beschafft werden, sondern können zentral bei dem Trust Proxy abgefragt werden. Dieser stellt die angefragten Attribute in der gewünschten Kodierung aus. Hierzu sind eine oder mehrere Attribute Assertion Quellen notwendig. Der Spezialfall eines Trust Proxys mit genau einer Senke und einer Quelle stellt einen Syntaxwandler für Attribute Assertions, beispielsweise von SAML Attribute Assertion nach Attribute Certificate, dar. Auch dies erfordert die Neusignierung der Assertion, da die Syntaxelemente mit signiert werden und somit die Umwandlung von einer Kodierung in einer andere zwangsläufig

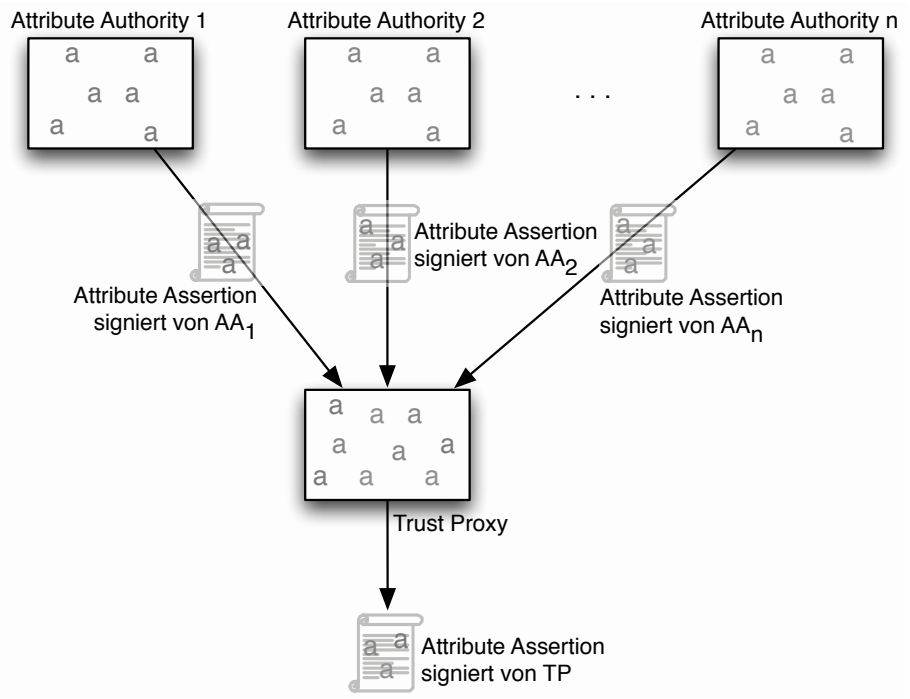

Abbildung 3.1: Übersicht Trust Proxying

die ursprüngliche Signatur invalidiert.

Einige Ansätze für an massiv verteilte Infrastrukturen angepasste Authentifizierungs- und Autorisierungsinfrastrukturen verwenden Trust Proxying. Der Ansatz ist attraktiv, da er es erlaubt, durch die vergleichsweise einfache Verwendung eines Trust Proxies eine kurzfristig verfügbare Implementierung für die verteilte, attributbasierte Autorisierung mit mehreren Attribute Authorities zu erreichen. Dies liegt daran, dass in der gesamten Infrastruktur nur noch an einer zentralen Stelle verschiedene Syntaxen aufeinander abgebildet werden müssen und nicht alle Grid-Ressourcen alle Attribute Authorities kennen und ihnen vertrauen müssen. Es genügt, wenn die eigentlichen Grid-Ressourcen den Trust Proxy kennen und von ihm ausgestellte Attribute Assertions mit dessen Zertifikat validieren können.

Der Fluss eines Attributs a mit dem Inhalt `sn=Groeper`[1] von der eigentlichen Attribute Authority b (*AttributeAuthority(a)=b*) über den Trust Proxy zur Ressource stellt sich wie in Formel 3.1 dar. $k_{priv,x}$ stellt hierbei den privaten Schlüssel von x dar. $k_{pub,x}$ ist der öffentliche Schlüssel von x. Die Funktion $sign(a, k_{priv,x})$ signiert das Attribut a mit dem privaten Schlüssel $k_{priv,x}$ und gibt ein Tupel a_s bestehend

[1] „sn“ steht im eduPersion Schema für „Surname“, also Nachname

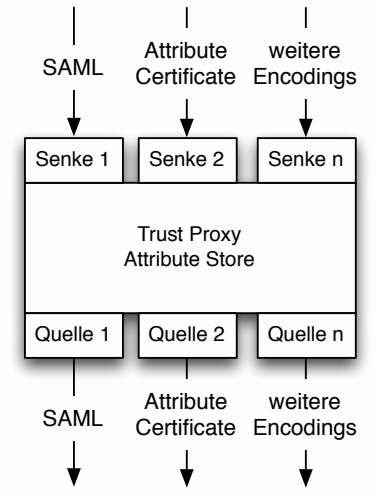

Abbildung 3.2: Komponenten eines Trust Proxys

aus dem Attribut und der Signatur zurück. Die Funktion $verify(a_s, k_{pub,x})$ prüft, ob die Signatur in a_s valide ist, d. h. ob sie mit dem privaten Schlüssel, der zu $k_{pub,x}$ gehört, erstellt worden ist und ob das Attribut nicht verändert wurde.

$$
\begin{aligned}
&\text{In der Attribute Authority b:}\\
&a = (sn, Groeper)\\
&a_{s1} = sign(a, k_{priv,b}) = \{(sn, Groeper), signature_1\}\\
&\text{In der Senke des Trust Proxys t:}\\
&a = (sn, Groeper) | verify(a_{s1}, k_{pub,b}) = true\\
&\text{In der Quelle des Trust Proxys t:}\\
&a_{s2} = sign(a, k_{priv,t}) = \{(sn, Groeper), signature_2\}\\
&\text{Im PIP der Ressource:}\\
&a = (sn, Groeper)\\
&verify(a_{s2}, k_{pub,t}) = true\\
&verify(a_{s2}, k_{pub,b}) = false
\end{aligned}
\tag{3.1}
$$

Im ersten Schritt wird das Attribut von der Attribute Authority mit ihrem privaten Schlüssel signiert ausgegeben. Der Trust Proxy nimmt das signierte Attribut entgegen und überprüft mittels des öffentlichen Schlüssels der Attribute Authority, ob die Signatur valide ist. Das Attribut kann nun beispielsweise in einem Nutzerprofil zwischengespeichert werden oder zur sofortigen Verwendung wieder neu ausge-

stellt werden. Dies kann in einer anderen Kodierung erfolgen, als derjenigen, in der das Attribut empfangen wurde. Hierzu wird das Attribut wieder mit dem privaten Schlüssel signiert, allerdings mit dem privaten Schlüssel des Trust Proxys. Auf den privaten Schlüssel der ursprünglichen Attribute Authority hat der Trust Proxy keinen Zugriff, da diese beiden Systeme nicht in derselben administrativen Domäne oder Organisation angesiedelt sein müssen. In der Ressource, die anhand des Attributs eine Zugriffsentscheidung fällt, wird das signierte Attribut wieder überprüft. Dies kann allerdings nicht mit dem öffentlichen Schlüssel der ursprünglichen Attribute Authority geschehen, da das Attribut mit dem privaten Schlüssel des Trust Proxys signiert wurde. Die fehlerhafte Verifikation durch den öffentlichen Schlüssel der originalen Attribute Authority ist hier nur zu Demonstrationszwecken dargestellt, da diese Attribute Authority der Ressource gar nicht bekannt ist. Daher verwendet die Ressource den öffentlichen Schlüssel des konfigurierten Trust Proxys und kann so sicherstellen, dass die Attribute Assertion von diesem stammt.

Ein direkter Rückschluss auf die ursprüngliche Attribute Authority ist nicht mehr möglich. Da die Ressource wiederum in einer dritten administrativen Domäne bzw. Organisation stehen kann, ist ein implizites Vertrauensverhältnis ebenfalls nicht gegeben. Dieser Verlust von Vertrauen in die Validität von Attributen aus einem Trust Proxying ist der entscheidende Nachteil eines solchen Ansatzes. Wenn dies aufgrund der Policies zwischen den beteiligten Organisationen nicht akzeptabel ist, kann Trust Proxying nicht zur Vereinfachung der Authentifizierungs- und Autorisierungsinfrastruktur einer Grid Umgebung eingesetzt werden.

3.4.3 Binden von Attributen an Identitäten

Binden von Attributen an X.509 Identitäten

Eine Quelle von Vertrauensverlust in verteilten Autorisierungsinfrastrukturen tritt auf, wenn ein Attribut zwar vertrauenswürdig von der Attribute Authority an die Ressource übermittelt wird, aber nicht sichergestellt werden kann, ob das Attribut auch für das Subjekt der Anfrage gilt.

Ein Attribut *Role=„Software Administrator@eine VO“* kann beispielsweise von dem VO Management System einer VO korrekt ausgestellt worden sein und mit korrekter Signatur vorliegen. Dieses Attribut kann von einem Nutzer in eine Anfrage eingebettet werden, um Schreibzugriff auf das VO Software Verzeichnis auf einer

Grid-Ressource zu erlangen. Um die Validität dieses Zugriffs festzustellen, reicht es auf der Ressource nicht aus, die Signatur der Attribute Assertion zu prüfen und festzustellen, ob die ausstellende Attribute Authority autorisiert ist, dieses Attribut auszustellen. Es muss weiterhin geprüft werden, ob das Attribut für das Subjekt der Anfrage gültig ist.

Dies kann nur nach erfolgreicher Identifizierung des Subjekts geschehen. Zusätzlich muss die Attribute Assertion, die das Attribut enthält, an die Identität desjenigen gebunden sein, für den sie ausgestellt wurde. Nur wenn das identifizierte Subjekt und die Attribute aus der Attribute Assertion dieselbe Entität beschreiben, ist das Attribut im aktuellen Kontext gültig.

Attribute Authorities im Grid Computing lassen sich grundlegend unterscheiden nach solchen, die vollständig auf X.509 Public Key Infrastrukturen aufsetzen (wie z. B. das VO-Managementsystem VOMS), und solchen, auf die dies nicht zutrifft (z. B. Identity Provider in einer Shibboleth Föderation). Im Folgenden werden zuerst VO-Managementsysteme betrachtet, die bereits darauf ausgelegt sind, in einer X.509 PKI basierten Umgebung eingesetzt zu werden. Anschließend werden Lösungsansätze erarbeitet, die notwendig sind, um nicht-X.509 PKI basierte Systeme einsetzen zu können, ohne dass die technologischen Unterschiede zu einem Vertrauensverlust in der Authentifizierungs- und Autorisierungsinfrastruktur führen.

Binden von VO-Attributen an Proxycredentials

VO-Attribute beschreiben Eigenschaften eines Mitglieds einer Virtuellen Organisation. Neben den Aufgaben für das VO-Management sollen VO-Attribute auch für die Autorisierung auf Grid-Ressourcen eingesetzt werden. So soll beispielsweise Schreibzugriff auf ein bestimmtes Verzeichnis mit VO-spezifischer Software nur Gridbenutzern gewährt werden, die innerhalb der Virtuellen Organisation die Rolle *Software Administrator* haben. Alle anderen VO-Mitglieder sollen hier nur lesenden Zugriff haben. Darüber hinaus können VO-Attribute auch für Monitoring- und Accountingdienste eingesetzt werden.

Die Basis dessen muss eine sichere und vertrauenswürdige Verifikation von VO-Attributen auf den Grid-Ressourcen sein. VO-Attribute einer bestimmten VO dürfen somit nur von dem VO-Management-System dieser einen Virtuellen Organisation ausgestellt worden sein. Auf keine Weise dürfen Attribute verwendet werden, die von der falschen Attribute Authority ausgestellt wurden oder die auf irgendeine Art

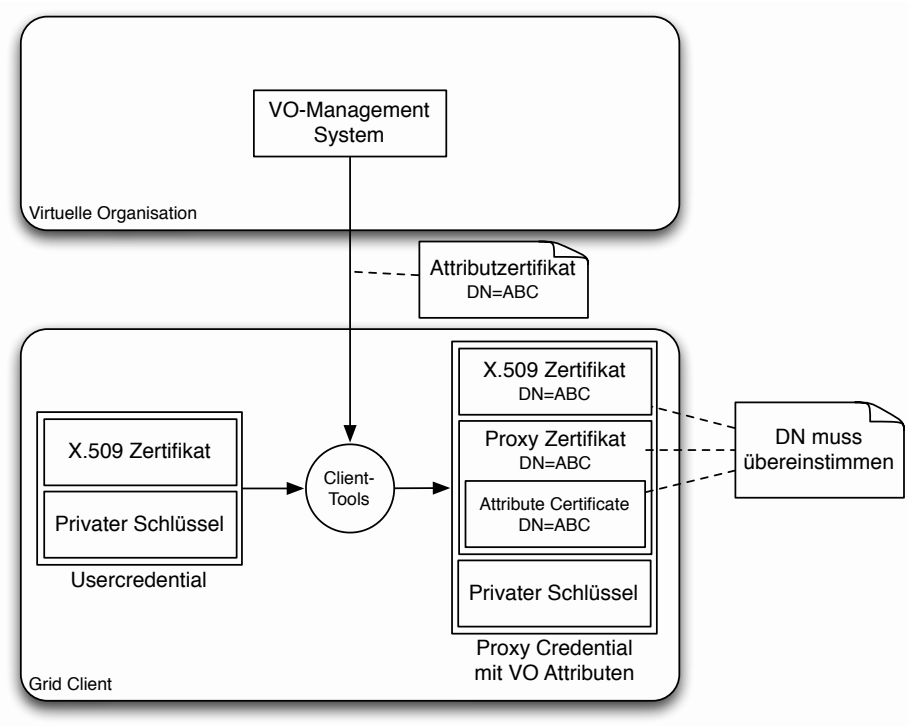

Abbildung 3.3: Einbinden von VO-Attributen in ein Proxycredential

und Weise nachträglich verändert wurden.

Das VOMS System speichert VO-Attribute zusammen mit dem X.509 Distinguished Name des Nutzers, welcher bei der Registrierung für die VO aus dem X.509 Zertifikat des Nutzers ausgelesen wurde. Dieser Distinguished Name ist auch Bestandteil jeder Attribute Assertion, die dieser VOMS Service ausstellt. Da diese Attribute Assertion nun von dem Nutzer in ein Proxyzertifikat eingebunden wird, kann jede Grid-Ressource einfach überprüfen, ob die Attribute vertrauenswürdig sind (Signatur des VOMS intakt) und ob sie tatsächlich den Nutzer beschreiben, für den sie ausgestellt wurden (X.509 Distinguished Name aus dem Attribute Certificate ist identisch mit dem des Proxyzertifikates). Sind diese Voraussetzungen nicht erfüllt, muss die Ressource die Anfrage ablehnen und ein versuchter Missbrauch ist wahrscheinlich. Die von neueren VOMS-Versionen ausgestellten SAML Attribute Assertions enthalten ebenfalls den X.509 Distinguished Name des Nutzers. In Abb. 2.8 kann in den Zeilen 43-56 erkannt werden, dass diese von einem SAML-VOMS ausgestellte SAML Attribute Assertion den X.509 Distinguished Name des Subjekts der Attribute enthält.

```
[groeper@ui1 groeper]$ voms-proxy-info --all
subject   : /O=GermanGrid/OU=UniHannover/CN=Ralf Groeper/CN=proxy
issuer    : /O=GermanGrid/OU=UniHannover/CN=Ralf Groeper
identity  : /O=GermanGrid/OU=UniHannover/CN=Ralf Groeper
type      : proxy
strength  : 512 bits
path      : /tmp/x509up_u531
timeleft  : 11:34:38
=== VO rvs extension information ===
VO        : rvs
subject   : /O=GermanGrid/OU=UniHannover/CN=Ralf Groeper
issuer    : /O=GermanGrid/OU=UniHannover/CN=voms1.gridlab.uni-hannover.de
attribute : /rvs/Role=NULL/Capability=NULL
timeleft  : 11:34:38
```

Abbildung 3.4: Inhalt eines Proxyzertifikats mit Attribute Certificate

Die Clientperspektive ist in dem unteren Teil von Abb. 3.3 dargestellt. Ein X.509 Credential, bestehend aus einem privaten Schlüssel und einem X.509 Zertifikat, wird verwendet, um davon ein Proxycredential abzuleiten. Hierbei wird ein Attribute Certificate vom VOMS Service angerufen und in das erstellte Proxyzertifikat eingebunden.

Es ist irrelevant, ob es sich bei dem ursprünglichen Zertifikat um ein herkömmliches, lang laufendes Zertifikat handelt oder um ein kurzlebiges Zertifikat von einer Online CA, solange die Online CA immer denselben X.509 Distinguished Name verwendet, um einem bestimmten Nutzer ein Zertifikat auszustellen.

Abb. 3.4 zeigt den Inhalt eines Proxyzertifikats mit einem vom VOMS Service ausgestellten Attribute Certificate. Im oberen Teil sind die Daten des Proxyzertifikats dargestellt, unten die Inhalte des Attribute Certificates. Das *Subject* der VO-Attribute ist der X.509 Distinguished Name. *Identity* ist dagegen der X.509 Distinguished Name aus dem umfassenden Zertifikat. Diese beiden Werte müssen übereinstimmen

Binden von Campusattributen an langlebige Zertifikate

Campusattribute beschreiben die Rollen und Zugehörigkeiten in einer realen Organisation, wie beispielsweise einer Universität. Diese Attribute sind zwar nicht gridspezifisch, stellen aber trotzdem eine wertvolle Quelle von Informationen dar, anhand derer im Grid autorisiert werden kann.

Campusattribute werden innerhalb der Heimateinrichtung eines Nutzers in einem Identity Management System verwaltet. Bisher waren solche Systeme nur innerhalb der Einrichtung verfügbar, da es weder Protokolle noch Policies gab, welche die organisationsübergreifende Verwendung erlaubten. In den letzten Jahren hat sich dies geändert und Standards wie SAML und Shibboleth stellen die technische Grundlage dar, anhand derer sich Einrichtungen zu Föderationen zusammenschließen. Die Föderation der deutschen Lehr- und Forschungseinrichtungen ist die Authentifizierungs- und Autorisierungs-Infrastruktur des DFN-Vereins, die *DFN-AAI*.

Attribute aus dieser Quelle müssen auf Grid-Ressourcen verfügbar sein und ihre Validität muss von diesen Ressourcen überprüft werden können, um sichere Autorisierungsentscheidungen treffen zu können. Die Validität umfasst hierbei nicht nur, dass sichergestellt ist, dass ein Attributsatz von der zuständigen Autorität ausgestellt wurde und nicht nachträglich verändert wurde. Ein weiterer Aspekt der Validität ist, dass ein gegebener Attributsatz denjenigen Nutzer beschreibt, der autorisiert werden soll und nicht einen anderen.

Die erste Anforderung lässt sich anhand der Metadaten der Shibboleth Föderation überprüfen. Diese Metadaten sind frei erhältlich und enthalten (u. a.) die Zertifikate aller Identity Provider der Föderation sowie die Scopes, für die sie Attribute ausstellen dürfen. Durch den öffentlichen Schlüssel aus den Zertifikaten lässt sich die Integrität einer Attribute Assertion überprüfen und die Scopes stellen sicher, dass ein Identity Provider keine Attribute ausstellt, für die er nicht zuständig ist (bspw. dass der IdP der Einrichtung A Attribute ausstellt, die einen Nutzer als Angestellten von Einrichtung B ausweisen). Ein Ansatz, der auch für Attribute ohne Scope funktioniert und außerdem flexibler ist, da er die Möglichkeit bietet, diese Entscheidung in den Policies der Ressource zu treffen und nicht über die Metadaten einer Föderation, ist in Kapitel 3.4.4 dargestellt.

Die zweite Anforderung, dass ein verifiziertes Attribut den aktuellen Nutzer beschreibt und nicht einem anderen Nutzer, ist schwieriger zu erfüllen. Dies liegt daran, dass für die Authentifizierung und die Autorisierung technisch unabhängige Systeme eingesetzt werden, nämlich Proxyzertifikate nach RFC 3820 für die Authentifizierung und SAML Attribute Assertions für die Autorisierung. In beiden Systemen wird der Nutzer durch einen eindeutigen Deskriptor identifiziert. Dies ist im Falle der Proxyzertifikate der X.509 Distinguished Name (Beispiel: `O=GermanGrid/`

`OU=UniHannover/CN=Ralf Groeper`). In der Shibboleth Föderation der DFN-AAI wird das Attribut *eduPersonPrincipalName* (ePPN) verwendet, das aus einem lokalen Teil (lokale User ID des Nutzers) und einem Scope, der die Heimateinrichtung des Nutzers kennzeichnet, besteht. Die beiden Teile sind durch ein @ getrennt. (Beispiel: `nhmqralf@uni-hannover.de`).

Wie zu erkennen ist, ist es nicht automatisch möglich, festzustellen, ob ein X.509 Distinguished Name aus der Grid Public Key Infrastructure und ein eduPersonPrincipalName aus der Shibboleth Föderation dieselbe Person bezeichnen. Bei der Gridnutzung wird ein Nutzer anhand seines X.509 Distinguished Name identifiziert, die Autorisierung soll aber anhand der mitgelieferten Shibboleth Attribute durchgeführt werden. Eine Grid-Ressource kann aber nicht überprüfen, ob die Attribute die authentifizierte Person beschreiben oder ob – beabsichtigt oder unbeabsichtigt – die Attribute einer anderen Person vorliegen.

Da eine automatische Abbildung von X.509 Distinguished Names auf eduPersonPrincipalNames nicht möglich ist, müssen andere Wege gefunden werden, um eine nachvollziehbare Abbildung von X.509 DNs auf ePPNs zu ermöglichen. Hierdurch wird ein X.509 DN an einen eduPersonPrincipalName *gebunden*.

Bindung durch die Certificate Authority: Ein möglicher Ansatz ist das Einbetten von Attributen in X.509 Zertifikate durch die ausstellende CA. Eine CA ist eine Einrichtung, deren Existenz von dem Vertrauen aller Beteiligten abhängt, dass sie ihren privaten Schlüssel unter allen Umständen geheim hält und dass sie verantwortungsvoll bei der Ausstellung von Zertifikaten vorgeht. Wenn eine CA also eine Attribute Assertion in ein Zertifikat einbettet, kann ihr – entsprechende nicht-technische Policies vorausgesetzt – vertraut werden, dass diese Attribute den Besitzer des Zertifikats beschreiben. Dies darf nicht mit dem Problem des Trust Proxyings verwechselt werden: Die Signatur der Attribute Assertion bleibt unangetastet. Die CA sichert nur zu, dass sie bei Ausstellung des Zertifikates überprüft hat, dass die an den eduPersonPrincipleName gebundenen Attribute und der X.509 Distinguished Name des Zertifikates beide die gleiche Person beschreiben. Hier ist die Abbildung von X.509 Distinguished Name auf eduPersonPrincipalName also implizit durch die Signatur der CA über das gesamte Zertifikat inklusive der signierten Attributzusicherung gegeben.

Problematisch ist hierbei, dass herkömmliche X.509 Zertifikate lange Gültigkeits-

zeiträume von über einem Jahr haben. In dieser Zeit können sich die Attribute aus der Campusföderation ändern, aber diese Änderungen werden nicht in die Grid-Infrastruktur übernommen. Der Nutzer kann sich mit den Attributen in seinem Zertifikat so lange autorisieren lassen, wie das Zertifikat gültig ist. Ein derartiges Vorgehen verstößt unter Umständen gegen die Policies der Shibboleth Föderation und/oder der Grid-Infrastruktur. Es ist möglich, eine Attribute Assertion mit einer Gültigkeitsdauer zu beschränken, dieser Ansatz führt aber dazu, dass der Nutzer ein prinzipiell gültiges Zertifikat nicht mehr im Grid nutzen kann, da die eingebetteten Attribute nicht mehr gültig sind.

eduPersonPrincipalName in X.509 Zertifikat einbinden: Eine weitere Möglichkeit ist es, den eindeutigen eduPersonPrincipalName (ePPN) aller Nutzer in ihre X.509 Zertifikate einzubinden. Somit werden nicht die eigentlichen Attribute des Nutzers Bestandteil des Zertifikats, aber ein Nutzer kann nachweisen, dass die Attribute in einer Attribute Assertion ihn beschreiben, wenn die ePPN in beiden Artefakten übereinstimmen.

Dies setzt voraus, dass die Registration Authorities der Certificate Authorities diese Angabe bei der Beantragung eines Zertifikates überprüfen. In der Praxis wird dieser Ansatz nicht verfolgt, da hierfür verlässliche explizite Policies zwischen allen beteiligten Shibboleth Föderationen, Certificate Authorities und Grid-Infrastrukturen notwendig wären. Dies würde wiederum entsprechende Gremien analog zu den Policy Management Authorites der CAs erfordern und ist in der Praxis somit nur mit erheblichem Aufwand zu etablieren.

X.509 Distinguished Name in Attribute Assertion einbinden: Die Bindung kann durch das Einbinden des X.509 Distinguished Name in die Attribute Assertion geschehen. Dies entspricht dem Vorgehen des Virtual Organization Membership Service (VOMS) bei VO-Attributen. In diesem Fall kann die vom Identity Provider signierte Attribute Assertion vom Nutzer selber in ein Proxyzertifikat eingebunden werden. Das Vorgehen ist in Abb. 3.5 dargestellt.

Es ist nicht notwendig, die Attribute Assertion von einer vertrauenswürdigen Stelle (d. h. der CA) einbinden zu lassen. Da der Nutzer ein Proxyzertifikat selber signiert, kann die Grid-Ressource sicherstellen, ob die Attribute Assertion den aktuell über den X.509 Distinguished Name identifizierten Nutzer beschreibt. Da der

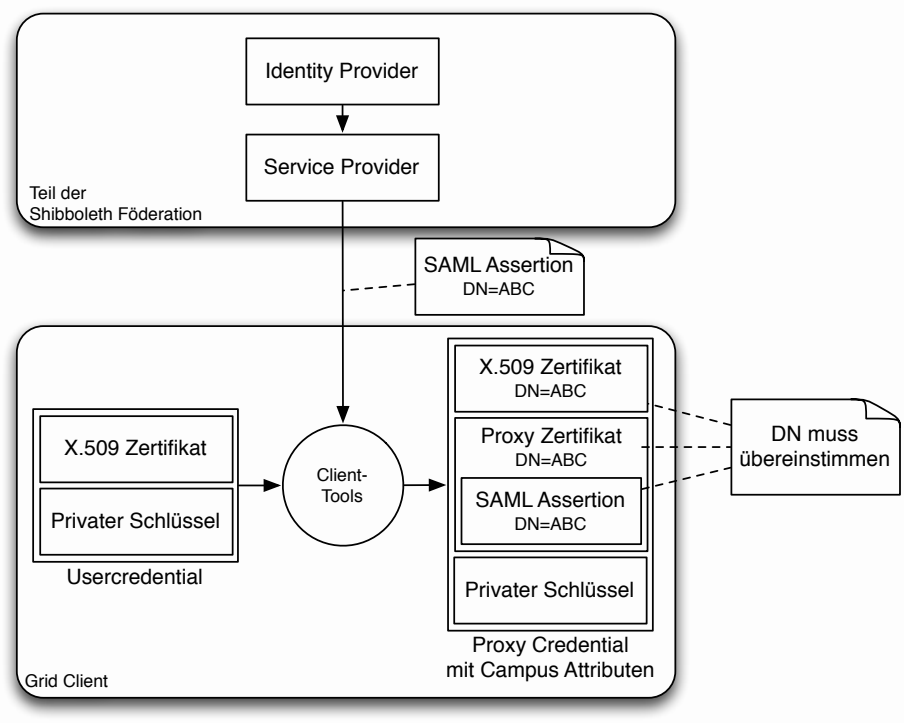

Abbildung 3.5: Einbinden von Campusattributen in ein langlebiges Zertifikat

X.509 DN Bestandteil der Attribute Assertion ist, muss die Grid-Ressource diesen mit dem bei der Authentifizierung aus dem Proxycredential gewonnenen DN vergleichen. Wenn diese identisch sind, kann die Ressource die Attribute akzeptieren.

Für die Umsetzung dieses Ansatzes ist ein Service Provider innerhalb der Föderation notwendig, bei dem der Nutzer über die Authentifikation bei seinem Identity Provider eine von diesem signierte SAML Attribute Assertion beziehen kann. Eine direkte Ausstellung der Attribute Assertion durch den Identity Provider ist in Shibboleth nicht vorgesehen. Ein solcher Dienst muss entwickelt und im Rahmen der Shibbolethföderation betrieben werden. Da dieser Dienst von der Virtuellen Organisation unabhängig für eine ganze Grid-Infrastruktur betrieben werden kann, muss für jede Kombination von Grid-Infrastruktur und Shibbolethföderation nur ein solcher Dienst betrieben werden, der sich aus Sicht der Shibboleth Föderation nicht von anderen Service Providern unterscheidet.

Der Nachteil ist, dass das X.509 Zertifikat und damit der Distinguished Name von einer Grid CA ausgestellt sind. Hierfür ist in den Identity Providern der Shibboleth Föderation ein neues Attribut, welches von allen Identity Providern der Shibboleth

Föderation für alle Nutzer verwaltet wird, notwendig. Allerdings ist es üblicherweise nicht möglich, derartige gridspezifische Informationen in der generischen Shibboleth Föderation zu verwalten und zu speichern.

Binden von Campusattributen an kurzlebige Zertifikate

In diesem Ansatz wird technisch gesehen der erste Ansatz aus Sektion 3.4.3 (Bindung durch die Certificate Authority) wieder aufgegriffen. In diesem Fall wird das Problem der langen Gültigkeitsdauer von klassischen Zertifikaten durch kurzlebige Zertifikate von einer Online CA gelöst. Wiederum sichert die CA zu, dass die eingebetteten Attribute den Zertifikatsinhaber beschreiben. Hier ist die Bindung allerdings noch stärker als bei der Lösung mit langlebigen Zertifikaten, da bereits der Login bei der Online CA über die Identity Provider der Shibboleth Föderation durchgeführt wird. Das bedeutet, die Online CA hat einen Sicherheitskontext des Nutzers, auf Basis dessen sie ein Zertifikat ausstellt und aus dem auch die Campusattribute kommen. Daher ist das versehentliche Binden einer falschen Attributzusicherung an ein Zertifikat ausgeschlossen. Im Kontext des D-Grid wird sowohl die Shibboleth Föderation als auch die Online CA vom DFN Verein betrieben. Daher ist auch organisatorisch sichergestellt, dass kein Vertrauensverlust durch diese Lösung eintritt.

Durch die deutlich kürzere Gültigkeitsdauer der kurzlebigen Zertifikate von 1 Mio. Sekunden bzw. $\approx 11\frac{1}{2}$ Tagen gegenüber der von mehr als einem Jahr von herkömmlichen Zertifikaten ist es in diesem Fall nicht möglich, langfristig veraltete Attribute einsetzen zu können. Daher kann dieser Ansatz im Einklang mit den Policies der Grid-Infrastruktur und der Shibboleth Föderation betrieben werden. Das Vorgehen ist in Abb. 3.6 dargestellt.

Der Zugriff auf die Online CA findet über den Webbrowser statt. Die Online CA ist ein regulärer Service Provider in der Shibboleth Föderation. Von der Webseite der Online CA wird nach dem Login über den Identity Provider der Heimatorganisation eine Java WebStart Anwendung initiiert, die ein Schlüsselpaar erzeugt. Der öffentliche Schlüssel wird in einem Certificate Signing Request (CSR) an die CA übermittelt. Diese ergänzt die Metadaten wie den X.509 Distinguished Name und fügt die vom Shibboleth Identity Provider erhaltene SAML Attribute Assertion ein. Das gesamte Artefakt wird dann mit dem privaten Schlüssel der CA signiert und an den Nutzer zurückgeliefert. Dieser hat nun ein Grid-Credential bestehend aus

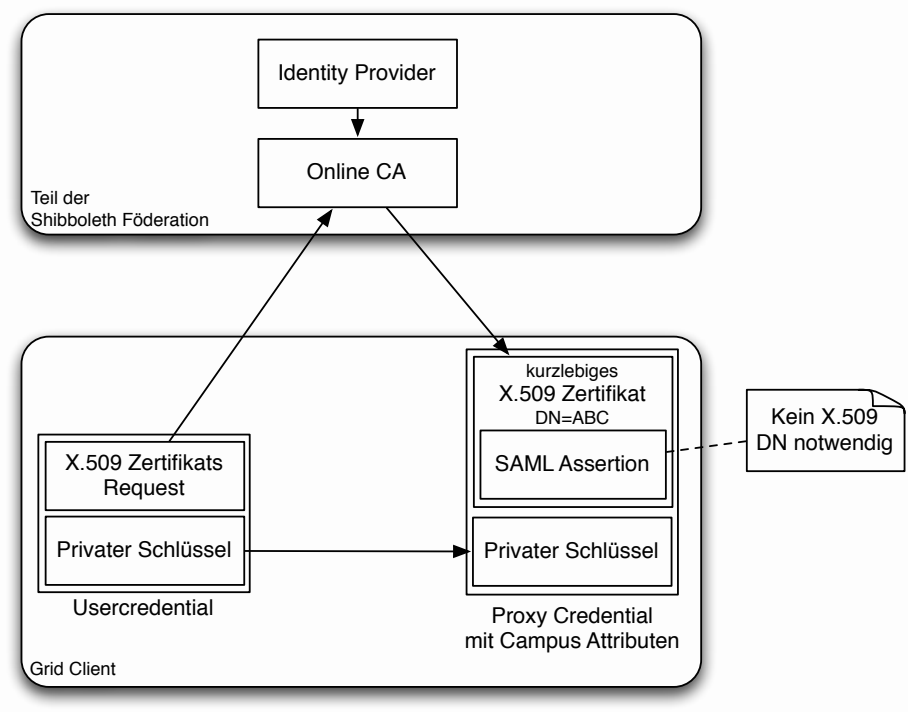

Abbildung 3.6: Einbinden von Campusattributen in ein kurzlebiges Zertifikat

einem kurzlebigen Zertifikat mit eingebetteter SAML Attribute Assertion und dem zugehörigen privaten Schlüssel. Die Bindung von Attributen an den X.509 Distinguished Name findet implizit über die Signatur der CA statt. Mit diesem Credential kann der Nutzer auf das Grid zugreifen oder bei Bedarf noch VO-Attribute in ein abgeleitetes Proxyzertifikat einbinden.

Wenn dies nicht ausreicht, um Vertrauen herzustellen, ist es zusätzlich möglich, dass die Online CA den eduPersonPrincipalName als Erweiterung in das ausgestellte Zertifikat aufnimmt. In diesem Fall kann die Grid-Ressource explizit vergleichen, ob das Zertifikat und die Attribute Assertion dieselbe Person beschreiben.

3.4.4 Multiple Attribute Authority Autorisierung

Aktueller Forschungsstand: Attribute-Based Multipolicy Access Control

Ein Ansatz zur formalen Modellierung von attributbasierten Autorisierungen in Multipolicyumgebungen wurde 2006 von Lang et. al. [42] veröffentlicht. Dieser Ansatz erweitert das Konzept der attributbasierten Zugriffskontrolle (ABAC, vgl. Kapitel 2.3.2) um die Fähigkeit, mehrere Policies in einer Autorisierungsentscheidung

berücksichtigen zu können. Daher heißt dieser Ansatz *Attribute Based Multipolicy Access Control (ABMAC)*. Die einzelnen Policies können technisch unterschiedlich implementiert sein, solange ein Policy Decision Point existiert, der dem Policy Enforcement Point ein standardisiertes Interface zur Verfügung stellt.

In der multipolicyfähigen Variante ABMAC wird die $can\_access(s, r, e)$ Funktion erweitert durch die Berücksichtigung mehrerer Policies, die jede für sich evaluiert und die Einzelergebnisse durch die $combine\_f()$ Funktion kombiniert werden.

Das Vorgehen ist in Abb. 3.7 dargestellt. Im ersten Schritt greift das Subjekt auf die geschützte Ressource zu und wird vom entsprechenden Policy Enforcement Point abgefangen. Dieser beauftragt in Schritt 2 den Master-PDP mit der Zugriffsentscheidung. Der PDP stellt den Policy Information Points die zur Verfügung stehenden Credentials und Attribute Assertions zur bereit, die hieraus Attribute extrahieren und dem PDP zurückliefern (Schritt 3). Für jede konfigurierte Policy (z. B. VO-Policy und Site-Policy) wird ein normaler ABAC-PDP aufgerufen (Schritt 4). Alle ABAC-PDPs erhalten alle verfügbaren Attribute. Die ABAC-PDPs liefern ihre jeweiligen Zugriffsentscheidungen zurück, welche vom Master-PDP kombiniert werden. Diese Entscheidung wird im fünften Schritt dem PEP übergeben, welcher dann in Schritt 6 den Zugriff an die Ressource leitet, wenn die Entscheidung positiv ist.

Es ergibt sich die $can\_access()$ Funktion in Formel 3.2.

$$\begin{aligned} &ABMAC\_Policy : can\_access(s, r, e) \leftarrow \\ &combine\_f(Policy_1(s, r, e), Policy_2(s, r, e), \ldots, Policy_n(s, r, e)) \end{aligned} \tag{3.2}$$

Die $combine\_f()$ Funktion verwendet einen der üblichen Kombinierungsalgorithmen für Policies, *First Applicable*, *Allow Overrides* oder *Deny Overrides* (siehe Kapitel 2.3.3). Da jede einzelne Policy wie in ABAC umgesetzt wird, entsprechen die einzelnen $can\_access()$ Funktionen der ABAC PDPs denen aus ABAC (siehe Formel 3.3). Hierbei stellt jede Policy mit ihrem PDP ein eigenes vollständiges ABAC System dar.

$$\begin{aligned} &Policy_i : can\_access(s, r, e) \leftarrow \\ &f\_policy_i(ATTR(s), ATTR(r), ATTR(e)) \end{aligned} \tag{3.3}$$

Zusammengenommen ergibt sich für die $can\_access()$ Funktion von ABMAC Formel 3.4.

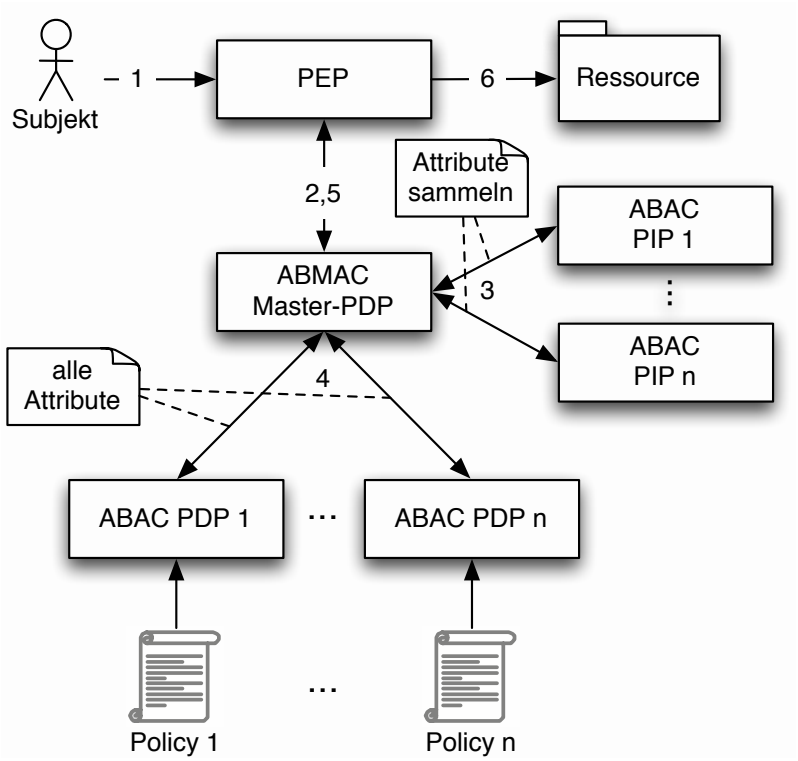

Abbildung 3.7: ABMAC Autorisierungsarchitektur (nach [42])

$$
\begin{aligned}
&ABMAC\_Policy : can\_access(s, r, e) \leftarrow \\
combine\_f(&f\_policy_1(ATTR(s), ATTR(r), ATTR(e)), \\
&f\_policy_2(ATTR(s), ATTR(r), ATTR(e)), \\
&\ldots, \\
&f\_policy_n(ATTR(s), ATTR(r), ATTR(e)))
\end{aligned}
\tag{3.4}
$$

Da der Ansatz in dieser Arbeit nicht nur Policies aus unterschiedlichen Quellen erfordert, sondern auch mehrere unabhängige Attribute Authorities zum Einsatz kommen, ist eine zusätzliche Erweiterung notwendig. ABMAC unterstützt zwar bereits Attribute aus unterschiedlichen Quellen, nach der Prüfung der Signaturen der Attribute Assertions in den Policy Information Points sind die Attribute aber nicht mehr nach ihrer Herkunft unterscheidbar. Das heißt, ein Policy Decision Point kann eine positive Zugriffsentscheidung treffen, obwohl die Attribute, auf denen er diese Entscheidung basiert, nicht von der vorgesehenen Attribute Authority ausgestellt wurden. Die ausstellende Attribute Authority muss zwar ein Attribut mit diesem Namen ausstellen dürfen, da der Policy Information Point das Attribut andernfalls

verwerfen würde, aber bei der Verwendung von mehreren Attribute Authorities kann es vorkommen, dass eine Attribute Authority ein Attribut regulär ausstellt, in der Policy auf der Ressource aber ein gleichnamiges Attribut einer anderen Attribute Authority gemeint war.

Der Ansatz, dieses Problem mittels Scopes zu lösen, d. h. die Werte der Attribute mit einem Scope zu versehen, der angibt wo dieses Attribut gilt (z. B. professor@uni-hannover.de mit dem Scope uni-hannover.de), löst nur einen Teil dieses Problems. Zum einen sind nicht alle Attribute im eduPerson-Schema mit einem Scope versehen und zum anderen schränkt dieser Ansatz die Flexibilität ein. Attribute werden von den Policy Information Points abgelehnt und verworfen, wenn der Scope nicht zur ausstellenden Attribute Authority passt. Der Policy Decision Point wird über solche Attribute nicht informiert und er bekommt sie nicht zur Verfügung gestellt. Es ist flexibler, diese Entscheidung über die Autorisierungspolicies zu lösen, anstatt Attribute statisch im Policy Information Point zu verwerfen.

Ein neu entwickelter, erweiterter Ansatz zur attributbasierten Autorisierung mit expliziter Unterstützung mehrerer Attribute Authorities im Policy Decision Point, der diesen Nachteil behebt, wird im Folgenden dargestellt.

Multi-Authority Attribute-Based Access Control

In einer Grid-Umgebung gibt es nicht nur getrennte Autorisierungspolicies von der Virtuellen Organisation und des Ressourcenbetreibers, sondern auch getrennte Attribut Autoritäten für die Attribute, die in diesen Policies verwendet werden. Während die VO eine eigene Attribute Authority betreibt (z. B. einen VOMS Service) basiert die Policy des Ressourcenbetreibers auf Attributen aus dem lokalen Identitätsmanagement oder auf Attributen aus einer Shibboleth Föderation. Um die Vertrauenswürdigkeit der Autorisierungsentscheidungen sicher zu stellen, muss also gewährleistet sein, dass der VO-PDP nicht eine positive Zugriffsentscheidung anhand von Attributen aus der Shibboleth Föderation trifft, sondern nur aufgrund von Attributen aus der Attribute Authority der VO. Streng genommen handelt es sich bei dem herkömmlichen Vorgehen um eine Variante des Trust Proxying (siehe Kapitel 3.4.2), da der Policy Information Point die Attribute an die Policy Decision Points weiterreicht, ohne dass diese prüfen können, von welcher Attribute Authority diese ausgestellt wurden.

In ABAC bzw. ABMAC sind Attribute Tupel der Form *(Name, Value)*. Um die

Quelle der Attribute im Policy Decision Point bewerten zu können, müssen die Attribute zu Tripeln erweitert werden. Diesen Ansatz nenne ich *Multi-Authority Attribute Based Access Control, MAABAC*. Die Attribut-Tripel haben die Form *(Name, Value, AttributeAuthority)*. Aus einem Attribut[2] *(Name='sn', Value='Groeper')* wird somit beispielsweise *(Name='sn', Value='Groeper', AttributeAuthority='IdP der Leibniz Universitaet Hannover)'*.

Es gibt zwei Möglichkeiten, die Überprüfung der Attribute Authority durchzuführen:

- Zum einen kann sie vor dem Aufruf der einzelnen ABAC Policy Decision Points durchgeführt werden, wenn jeder Policy Decision Point und damit jede Policy nur Attribute genau einer Attribute Authority verarbeitet. Dies hat den Vorteil, dass wie bei ABMAC die eigentlichen PDPs und Policies nicht angepasst werden müssen. Der Nachteil ist, dass diese Lösung weniger flexibel ist, da in jeder Policy nur Regeln mit Attributen von genau einer Attribute Authority enthalten sein können.

- Die andere Möglichkeit ist es, die Korrektheit der Attribute Authority erst im eigentlichen Policy Decision Point zu prüfen. Dieser Ansatz ist flexibler, da in einer Policy und dem zugehörigen Policy Decision Point Attribute von mehreren Attribute Authorities verwendet werden können, ohne die Vertrauenswürdigkeit einzuschränken. Der Nachteil ist, dass die existierenden Policy Decision Points, Policies und Policy Information Points nicht verwendet werden können.

Prüfung der Attribute Authority vor dem PDP: Die Prüfung, ob eine Policy p und der hierfür zuständige PDP für ein gegebenes Attribut von x zuständig sind wird vor den Aufrufen der einzelnen ABAC PDPs ausgeführt. Jede $can\_access()$-Funktion wird nur mit den Attributen aufgerufen, die für die hinter diesem PDP stehende Policy geeignet sind. Es ist prinzipiell auch möglich, Attribute mehrerer Attribute Authorities zuzulassen, aber es ergibt sich dann wieder das Problem, welches mit diesem Ansatz gelöst werden soll: Da Attribute-Value-Pairs mehrerer Attribute Authorities ohne Angabe ihres Ursprungs im PDP vorliegen, ist es nicht mehr nachvollziehbar, von welcher Attribute Authority ein Attribut ausgestellt wurde. Die

[2] „sn“ steht im eduPersion Schema für „Surname“, also Nachname

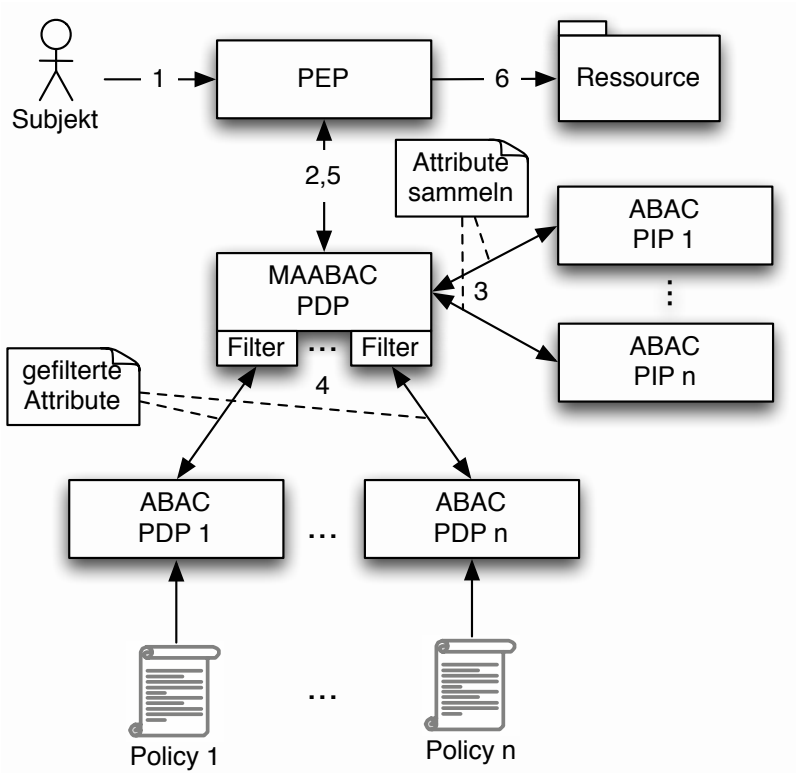

Abbildung 3.8: MAABAC mit vorheriger Filterung der Attribute

Policies, d. h. die Regeln der Policies, müssen nicht geändert werden, aber zusätzlich eine Konfiguration gepflegt werden, welche die für einen PDP zuständigen Attribute Authority spezifiziert.

In Abb. 3.8 ist das Vorgehen dargestellt. Es entspricht dem Vorgehen von ABMAC mit der Erweiterung, dass der MAABAC Master-PDP für jeden ABAC-PDP einen Filter besitzt, der den ABAC-PDPs nur Attribute von den konfigurierten Attribute Authorities übergibt und nicht alle gesammelten Attribute.

Da die Attribute Authority vor dem Aufruf des Policy Decision Points geprüft werden muss, ergibt sich Formel 3.5 für die $can\_access()$ Funktion des MAABAC Master PDPs. Hierbei bezeichnet A_i die Menge aller von den Policy Information Points gelieferten Attribut-Tripel a_j von den für Policy i geeigneten Attribute Authorities. Es gilt $a_j \in A_i$. Die eigentliche Policyfunktion nach Formel 3.3 ändert sich nicht, es werden nur nicht alle Attribute an alle ABAC PDPs übergeben.

$$
\begin{aligned}
&MAABAC\_Policy : can\_access(s, r, e) \leftarrow combine\_f(\\
&f\_policy_1(ATTR(s), ATTR(r), ATTR(e) \mid ATTR(s), ATTR(r), ATTR(e) \in A_1), \\
&f\_policy_2(ATTR(s), ATTR(r), ATTR(e) \mid ATTR(s), ATTR(r), ATTR(e) \in A_2), \\
&\ldots, \\
&f\_policy_n(ATTR(s), ATTR(r), ATTR(e) \mid ATTR(s), ATTR(r), ATTR(e) \in A_n))
\end{aligned}
\tag{3.5}
$$

Das programmatische Vorgehen des MAABAC Master PDPs ist in Algorithmus 1 dargestellt.

Algorithm 1 Aufrufen der PDPs und Ausführen des Combining Algorithms

```
1: boolean function can_access(list_of_attributes)
2: list_of_results = ∅
3: for all configured_PDP p do
4:     filtered_attributes = ∅
5:     for all attributes a do
6:         if a.attribute_authority == p.attribute_authority then
7:             filtered_attributes ← filtered_attributes + a
8:         end if
9:     end for
10:    list_of_results ← list_of_results + p.can_access(filtered_attributes)
11: end for
12: return f_combine_alg(list_of_results)
```

Prüfung der Attribute Authority im PDP: Die Prüfung der Attribute in den Slave PDPs erfordert eine Anpassung dieser PDPs. Die ABAC PDPs, die bisher verwendet wurden, sind hier nicht einsetzbar. Die Signatur der *can_access*() Funktionen ändert sich nicht, sie werden weiterhin mit den Attributen von Subjekt, Ressource und Environment aufgerufen, wobei die Attribute wie oben beschrieben nun Tripel und keine Tupel mehr sind. Auch in den Policies steht zusätzlich zum Namen und Wert des Attributs die zuständige Attribute Authority. Dies kann durch einen Authority Scope realisiert werden. Dieser ist analog zu einem Attribut-Scope

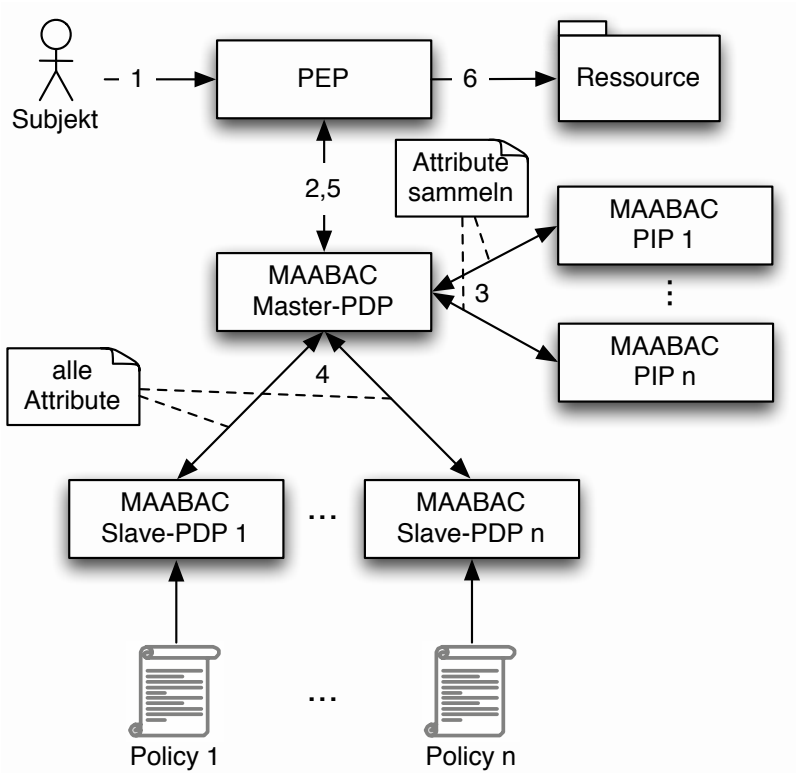

Abbildung 3.9: MAABAC ohne vorherige Filterung der Attribute

definiert. Ein Attribut-Scope zeigt an, für welchen Bereich ein Attributwert gilt. So ist beispielsweise das Attribut

eduPersonScopedAffiliation=student@uni-hannover.de

durch den Wert „student“ und den Scope „uni-hannover.de“ belegt. Zusätzlich kann noch ein Authority Scope angefügt werden. Dieser wird durch ein Doppelkreuz (#) abgetrennt. Es ergibt sich ein Attribut

eduPersonScopedAffiliation=student@uni-hannover.de#idp.uni-hannover.de

Dies zeigt an, dass der Identity Provider der Universität Hannover die zuständige Attribute Authority ist. Das Vorgehen ist in Abb. 3.9 dargestellt und entspricht vom zeitlichen Ablauf her den vorhergehenden Modellen. Es werden wie in ABMAC wieder alle Attribute an alle PDPs weitergeleitet, aber mit der Information über die ausstellende Attribute Authority. Die MAABAC Slave-PDPs wählen selber aus, welche Attribute sie akzeptieren und welche nicht.

Die Policy Information Points müssen derart modifiziert werden, dass sie dem MAABAC-PDP neben dem Attributnamen und dem Attributwert auch die aus-

stellende Attribute Authority zur Verfügung stellen. Die ABMAC_Policy-Funktion des Master-PDPs ändert sich gegenüber ABMAC nicht, da weiterhin alle Attribute an die einzelnen $can\_access()$ Funktionen der Slave PDPs übergeben werden. Diese müssen nun selber für jede Policyregel diejenigen Attribute auswählen, die von der angegebenen Attribute Authorities stammen. Somit ergibt sich beispielhaft eine $can\_access()$ Funktionen nach Formel 3.6.

$$\begin{aligned} &R1 : can\_access(s, r, e) = \\ &[s.Role = \text{`}student@uni - hannover.de\text{'} \wedge Role.AA = \text{`}idp.uni - hannover.de\text{'}] \wedge \\ &[Name(r) = \text{`}BuchAusleihe\text{'} \wedge AA(Name) = \text{`}Lokales\ Environment\text{'}] \end{aligned} \tag{3.6}$$

Der programmatische Ablauf im Master PDP ist in 2 beschrieben. Hierbei bezeichnet x ein Subjekt, eine Ressource oder das Environment und p die entsprechende Policy.

Algorithm 2 Funktion des MAABAC Master-PDPs

```
1: boolean function query_pdps()
2: list_of_results = ∅
3: for all attributes a do
4:     list_of_attributes ← list_of_attributes + a
5: end for
6: for all configured_PDP p do
7:     list_of_results ← list_of_results + p.can_access(list_of_attributes)
8: end for
9: return f_combine_alg(list_of_results)
```

Die Slave PDPs müssen so erweitert werden, dass sie die Gleichheit der Attribute von der Policy und der Attribute aus der Anfrage inklusive der ausstellenden Attribute Authority überprüfen. Das Vorgehen ist in Algorithmus 3 dargestellt. Es muss für jede Regel der Policy geprüft werden, ob die in ihnen geforderten Attribute in den Attributen der Anfrage enthalten sind. Ist dies der Fall, wird die Regel angewandt und ihr Ergebnis gespeichert. Falls mehrere Regeln aus der Policy zutreffen, müssen diese Ergebnisse noch durch die üblichen Combining Algorithmen zum Endergebnis der Autorisierungsentscheidung des PDP zusammengeführt werden.

Algorithm 3 funktion der MAABAC Slave-PDPs

```
1: boolean function can_access(list_of_attributes)
2: list_of_results = ∅
3: for all rules r from policy p do
4:    for all attributes ap from r do
5:       for all attributes al from list_of_attributes do
6:          if ap.attribute_authrotity = al.attribute_authority then
7:             list_of_results ← list_of_results + r.result
8:          end if
9:       end for
10:   end for
11: end for
12: return combine(list_of_results)
```

Wenn eine Regel diese Match-Bedingung erfüllt, wird sie auf die aktuelle Anfrage angewandt, d. h. in den Kombinierungsalgorithmus wird „Allow“ oder „Deny“ ausgegeben. Diese Erweiterung lässt sich äquivalent sowohl auf die Attribute Authorities des Subjekts anwenden als auch auf die Environment- und Resources AAs. Der häufigste Anwendungsfall sind allerdings die Attribute der Subjekte einer Autorisierungsentscheidung. Im den Anforderungen der D-Grid Communities spielen Environmentattribute bisher keine Rolle. Ressourcenattribute werden dagegen gefordert, beispielsweise ob eine Ressource eine „High-Security“ oder „Low-Security“ Einstufung hat. Anhand dieser kann beispielsweise entschieden werden, ob kurzlebige Zertifikate zur Authentifizierung akzeptiert werden.

Mit Hilfe des MAABAC-Ansatzes ist es möglich, die durch die Bereitstellung von Attributen aus mehreren Quellen ermöglichte Funktionserweiterung in Autorisierungsentscheidungen auf Ressourcen einzusetzen. Es erweitert die Möglichkeiten, die bisher durch Verwendung eines Scopes bei Attributen umgesetzt werden. Zusätzlich verhindert es den Fall, dass ein Attribute zwar korrekt von einer Attribute Authority ausgestellt, aber in der Policy einer Ressource ein gleichnamiges Attribut einer anderen Attribute Authority gemeint ist.

3.5 Analyse von Technologien für die attributbasierte Autorisierung im Grid Computing

3.5.1 Aktueller Forschungsstand

Aktueller Forschungsarbeiten befassen sich mit drei Aspekten, die in dieser Arbeit relevant sind:

Der erste Aspekt ist die Einbindung von VO-Attributen in die Autorisierung im Grid. Dieser Bereich ist bereits weitgehend implementiert in aktuellen Grid Middlewares. Mit VOMRS und VOMS existieren Systeme, die diese Attribute verwalten und alle Grid Middleware unterstützen mit unterschiedlicher Produktionsreife die Autorisierung anhand von Attributen aus VOMS Attribute Certificates. In [57] wird die Integration des SAML-VOMS in UNICORE diskutiert. Die Integration von VOMS in das Globus Toolkit wird in einem Globus Incubator Projekt vorangetrieben [2]. Die Ergebnisse dieser Arbeiten werden bei der konkreten Implementierung der Autorisierung im D-Grid genutzt.

Der zweite Aspekt ist die Einbindung von Shibboleth Föderationen in die Autorisierung im Grid Computing. Diese wurde u. a. im Rahmen des GridShib [6] Projektes behandelt. Die Ergebnisse dieses Projekts sind maßgeblich in diese Arbeit eingeflossen. Aufgrund der unterschiedlichen Anforderungen im GridShib Projekt und der dieser Arbeit, insbesondere der Einbindung von VO-Attributen zusätzlich zu den Attributen aus der Shibboleth Föderation, ist dieser Ansatz alleine allerdings nicht ausreichend. Weitere Projekte, die Shibboleth und Grid Computing kombinieren sind myVocs [8] und VASH [22]. Sie werden hier ebenfalls ausführlich betrachtet und so kombiniert, dass die Anforderungen von Grid-Infrastrukturen mit Campus- und VO-Attributen erfüllt werden.

Der dritte Aspekt ist die Interoperabilität von Autorisierungsinfrastrukturen zwischen verschiedenen Grid Middlewares und Grid-Infrastrukturen[3]. In [28] wird eine middlewareübergreifende attributbasierte Autorisierungsinfrastruktur beschrieben, die das Globus Toolkit basierte Open Science Grid [50] und das gLite basierte EGEE [54] verbindet. Hierbei wird ein VOMS Service eingesetzt, um VO-Attribute zu verwalten und als Attribute Certificates auszustellen. Der Fokus liegt hierbei allerdings

[3]Für die meisten Grid-Infrastrukturen sind Middleware und Infrastruktur gleichbedeutend. Das D-Grid ist hier eine Ausnahme, da es drei verschiedene Middlewares einsetzt, von denen keine im D-Grid selber entwickelt wurde.

auf der Bereitstellung eines Zentralen Policy Decision Points bei jedem Ressourcen Provider, der für alle dort basierten Ressourcen die Autorisierungsentscheidungen fällt. Die explizite Unterstützung mehrerer Attribute Authorities in einer Anfrage wird nicht behandelt. PERMIS [10] ist ein umfangreiches Autorisierungsframework, welches sich für den Einsatz im Grid Computing eignet [9]. Es bietet jedoch keine ausreichenden Funktionen, um Attribute mehrerer Attribute Authorities nach den Kriterien dieser Arbeit vertrauenswürdig an Ressourcen zu übermitteln. Es kann in Zukunft allerdings erforscht werden, ob es sich als zentraler PDP für Ressourcenanbieter eignet, der anhand der mit den in dieser Arbeit vorgestellten Mitteln bereitgestellten Attributen Autorisierungsentscheidungen trifft.

Es gibt bisher keine Arbeiten zur Verbindung aller drei genannten Aspekte. Es ist das Thema dieser Arbeit, diese Verbindung zu erforschen und dabei entstehende Vertrauensprobleme zu erkennen und zu bewerten.

3.5.2 X.509-basierte Technologien zur Verwaltung von VO-Attributen

Der Virtual Organization Membership Service (VOMS) und der Virtual Organization Management Registration Service (VOMRS)

Der VOMS Service aus dem gLite Projekt dient dem webbasierten Management von Virtuellen Organisationen im Grid Computing. Er bietet eine Webschnittstelle, mittels der Mitglieder in eine VO aufgenommen und gelöscht werden können. Darüber hinaus können Attribute der VO-Mitglieder verwaltet werden. VOMS außerdem die Funktion, Nutzerattribute als signierte Attribute Assertion auszustellen. Ursprünglich konnten nur Attribute Certificates ausgestellt werden, während aktuelle Versionen auch SAML Attribute Assertions unterstützen. Für die Abfrage von Attribute Certificates dient eine proprietäre Schnittstelle während SAML Attribute Assertions über eine SOAP-Schnittstelle abgefragt werden können. Zusätzlich bietet VOMS eine Java und eine C-API an, die sowohl für Remoteabfragen als auch für lokale Abfragen von Nutzerinformationen entsprechende Methoden bereitstellt.

VOMRS ist ebenfalls ein webbasiertes System zur Verwaltung der Mitglieder Virtueller Organisationen und bietet eine ähnliche Funktion wie VOMS. VOMRS wird vom VOX-Projekt am Fermilab in den USA entwickelt und dort im Zusammenspiel mit einer Globus Toolkit basierten Grid-Infrastruktur eingesetzt. Im Unter-

schied zu VOMS bietet VOMRS ausgefeilte Workflows für die Beantragung der VO-Mitgliedschaft und für die regelmäßige Aufforderung an die Nutzer, die Nutzungsbedingungen der VO zu bestätigen. Dies beinhaltet u. a. die automatische Verifikation der angegeben Emailadresse von neuen Mitgliedern. Im Gegensatz zu VOMS bietet VOMRS keine Schnittstelle für die Ausgabe der Nutzerattribute als Attribute Assertions. Die Attribute können lediglich über einen Webbrowser verwaltet und eingesehen werden. Auch eine API zur lokalen Abfrage der Nutzerinformationen ist nicht vorhanden. Die Informationen müssen für die Verwendung außerhalb der Weboberfläche direkt aus der Datenbank ausgelesen werden.

VOMRS bietet eine Funktion an, die das Synchronisieren der Nutzerdaten in eine VOMS Datenbank bereitstellt. Beide Systeme können also komplementär gemeinsam eingesetzt werden: VOMRS bietet als führendes System umfangreiche Workflows für die Verwaltung von VO-Mitgliedschaften an, während VOMS die Ausgabe der Nutzerattribute als Attribute Certificate oder SAML Attribute Assertion übernimmt. Die Weboberfläche des VOMS wird in diesem Fall nicht benötigt.

Funktionen, die sowohl VOMRS als auch VOMS bieten, sind:

- Eine Datenbank für die Speicherung der Nutzerinformationen.
- Speicherung sowohl von beliebigen Nutzerattributen als Attribut-Wert-Paare als auch der pseudohierarchischen Group, Role und Capability-Attribute die von gLites Autorisierungssystem verwendet werden.
- Ein Web-Frontend für die Nutzerregistrierung und Attributverwaltung.

Funktionen, die nur VOMS bietet, sind:

- Ein Web Service basierter Zugriff auf SAML Attribute Assertions.
- Eine Schnittstelle für den Zugriff auf Attribute Certificates.
- Eine API für den Zugriff auf Attribute Certificates und lokalen Zugriff auf die VO-Mitgliedsdaten.

Funktionen, die nur VOMRS bietet, sind:

- Umfangreiche Workflows für die Registrierung und Verwaltung von VO-Mitgliedern und ihrer Attribute.

- Regelmäßig notwendige Bestätigung der Acceptable Usage Policy (AUP) der Virtuellen Organisation.
- Speicherung von Stammdaten der Nutzer wie Emailadresse und Telefonnummer.
- Synchronisierung der VO-Mitgliedsdaten mit einer VOMS Datenbank.

Beide Systeme werden im D-Grid bereits erfolgreich zusammen eingesetzt und von allen D-Grid VOs verwendet. Hinzu kommt noch das Grid-Ressourcen-Management, das notwendig ist, um aus den VO-Mitgliedsinformationen Zugriffspolicies zu erstellen und diese auf die Grid-Ressourcen zu verteilen. Um dies zu ermöglichen, wurde am Forschungszentrum Jülich eine Schnittstelle zwischen der VOMRS-Datenbank und der Ressourcenverwaltung GRRS (Grid Resource Registration Service) entwickelt. Im GRRS werden die vorhandenen Grid-Ressourcen verwaltet und neue werden über ihn registriert. Hierzu gehört insbesondere die Liste der durch den Ressourcenprovider unterstützten Virtuellen Organisationen. Wenn Autorisierungspolicies auf Basis der VO-Mitgliedschaften für Grid-Ressourcen erstellt werden, kann somit der GRRS alle Mitglieder der Unterstützten VOs und gegebenenfalls noch entsprechende Attribute wie „VO-Softwareadministrator“ aus der VOMRS-Datenbank auslesen und daraus eine VO-Policy generieren. Diese wird auf die Grid-Ressourcen übertragen, welche der VO zur Verfügung stehen und dort für die Autorisierung verwendet.

3.5.3 Autorisierungsfunktionen in Grid Middlewares

Die in dieser Arbeit diskutierte Autorisierungsinfrastruktur soll middlewareübergreifend funktionieren. Die von den in Deutschland verwendeten Middlewares Globus Toolkit, gLite und UNICORE bereitgestellten Autorisierungsfunktionen werden im Folgenden analysiert. Die in dieser Arbeit erforschten erweiterten Autorisierungsansätze verwenden diese Funktionen, soweit sie vorhanden und anwendbar sind.

UNICORE

Ältere Versionen von UNICORE unterstützen die Grid Security Infrastructure nicht. Das heißt die Authentifizierung mit Proxyzertifikaten ist nicht möglich und auch die Delegation von Rechten ist nicht vorgesehen. Diese ursprüngliche Version ist

keine Grid Middleware, sondern ein grafisches Zugangssystem für Clustersysteme. Von den Autorisierungsmöglichkeiten her unterschied sich diese Version nicht von herkömmlichen Zugängen zu Hochleistungsrechenressourcen basierend auf `ssh` oder `rsh`. Eine organisationsübergreifende Nutzung der Ressourcen war somit mit dem üblichen Verwaltungsaufwand verbunden: Jeder Nutzer musste auf jeder Ressource einen lokal gültigen Login besitzen.

Um UNICORE auch in Grids verwenden zu können, wurde die Unterstützung für die Grid Security Infrastructure implementiert und in UNICORE5 veröffentlicht. Hiermit ist, analog zum Globus Toolkit, nur die identitätsbasierte Autorisierung anhand des X.509 Distinguished Names möglich. Das Mapping von DN auf lokalen UNIX Account wird in der *Unicore User Database (UUDB)* verwaltet. Neuere Versionen bieten zusätzlich attributbasierte Autorisierung, z. B. mit VOMS Attributen [57] und die aktuelle UNICORE6 Version unterstützt SAML Attribute Assertions. Für den Einsatz in rein UNICORE-basierten Infrastrukturen gibt es einen eigenen VO Management Service, das *UNICORE VO System, UVOS* [7]. Dieser ermöglicht die Autorisierung anhand von VO-Attributen.

gLite

Der Virtual Organization Management Service (VOMS) verwaltet die Attribute aller Nutzer und stellt diese in Form von Attribute Certificates nach RFC 3281 [19] aus. Im einfachsten Fall wird eine Anfrage durch Proxycredentials authentifiziert und anschließend anhand der eingebetteten VO-Zugehörigkeit autorisiert. In der gLite Middleware wurde bereits frühzeitig die attributbasierte Autorisierung im Zusammenspiel mit sogenannten Poolaccounts umgesetzt. Hierbei wird der Job auf einen lokalen UNIX-Account abgebildet, mit dessen UNIX-ID der Job auf der Grid-Ressource ausgeführt wird. Hierbei wird kein 1:1-Mapping von Nutzern auf einen Account verwendet, sondern für jeden Job ein lokaler Account ausgewählt, der nach Beendigung des Jobs aufgeräumt und wieder neu vergeben wird. Diese Lösung skaliert gut für große Mengen von Nutzern, wird in Grid-Infrastrukturen mit erhöhten Sicherheitsanforderungen allerdings nicht eingesetzt, da durch die gemeinsame Nutzung von Nutzeraccounts die Vertraulichkeit von Daten nicht zugesichert werden kann.

Die gLite Grid Middleware setzt seit Beginn ihrer Entwicklung auf das VO Managementsystem VOMS. Dieses wird im Rahmen von gLite entwickelt und stellt den

Grid-Ressourcen die Attribute in Form von Attribute Certificates zur Verfügung. Diese werden per Attribute Push an die Grid-Ressourcen übermittelt (siehe Kapitel 3.5.2). Viele Komponenten von gLite unterstützen die attributbasierte Autorisierung anhand der vom VOMS ausgestellten Attribute Certificates.

Globus Toolkit

Das Globus Toolkit 4 [24] unterstützt ohne zusätzliche Komponenten ausschließlich die identitätsbasierte Autorisierung. Hierfür ist ein sogenanntes `grid-mapfile` notwendig, welches X.509 Distinguished Names mit einem 1:1 Mapping auf lokale UNIX Accounts abbildet. Wird ein Nutzer authentifiziert, wird sein X.509 Distinguished Name in dieser Liste gesucht. Wird ein passender Eintrag gefunden, ist der Nutzer autorisiert und wird für die Ausführung des Jobs auf der Ressource auf den angegebenen UNIX Account abgebildet. Ein n:1-Mapping, d. h. von mehreren Nutzern auf einen UNIX-Account, ist technisch möglich, aber es gibt dann keine Möglichkeit zu verhindern, dass mehrere Nutzer gleichzeitig auf denselben Account zugreifen. Dies führt nicht nur zu offensichtlichen Problemen bei der Vertraulichkeit der Daten sondern auch zu technischen Problemen, da sich z. B. die Standardausgabekanäle mehrerer gleichzeitiger Jobs überlagern.

3.5.4 Technologien zur Einbindung von Shibboleth in Grid Middlewares

Da bisher keine Grid Middleware auf Shibboleth für die Authentifizierungs- und Autorisierungsinfrastruktur aufsetzt und Shibboleth selber nur web-basierte Service Provider unterstützt, die somit nur per Webbrowser zugreifbar sind, ist weitere Software notwendig, die zwischen der Shibboleth-Welt und der Grid-Welt vermittelt. Hierbei können alle der bereits diskutierten Szenarien eintreten, die einen Verlust von Vertrauen in die verwendeten Attribute mit sich bringen. Im Folgenden werden existierende Ansätze untersucht und analysiert, inwieweit sie Vertrauensprobleme implizieren. Weiterhin wird dargestellt, in welcher Form diese Ansätze von den Projekten eingesetzt werden, in deren Rahmen sie entstanden sind. In Kapitel 4.2 werden darüber hinaus verschiedene Ansätze diskutiert, wie sich diese Ansätze abwandeln und kombinieren lassen, um die in dieser Arbeit formulierten Anforderungen zu erfüllen.

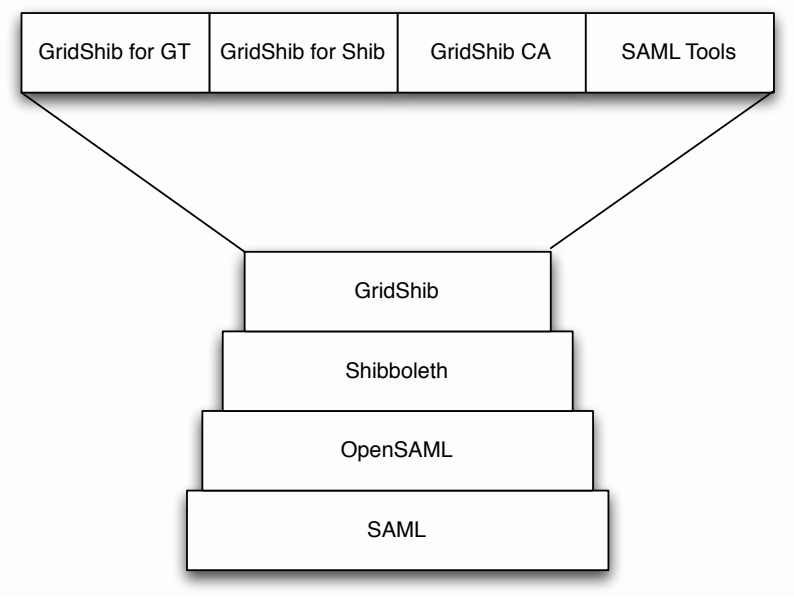

Abbildung 3.10: Komponenten von GridShib

GridShib

GridShib wurde bis 2009 im Rahmen der Globus Alliance entwickelt. Die GridShib CA wird inzwischen im Rahmen des CILogon Projektes [11] weiter gepflegt, während die anderen Komponenten von den ursprünglichen Entwicklern nicht mehr gepflegt werden. Sie stehen aber weiterhin als Open Source Software zur Verfügung.

GridShib besteht aus mehreren Teilen, die es im Zusammenspiel ermöglichen, auf Globus-basierten Gridkomponenten Autorisierungsentscheidungen basierend auf Attributen aus Shibbolethföderationen zu treffen. Darüber hinaus ermöglicht die GridShib CA den Zugriff auf das Grid ohne ein langlebiges, von einer Certificate Authority ausgestelltes X.509 Zertifikat. Die einzelnen Komponenten und ihre Abhängigkeiten sind in Abb. 3.10 dargestellt. Shibboleth basiert auf dem SAML Standard und der OpenSAML Referenzimplementierung. Hierauf setzt GridShib mit seinen vier Komponenten auf.

Die vier Komponenten sind:

- *GridShib for Globus Toolkit* ist ein Policy Decision Point (PDP) für die WSRF-basierten Dienste aus dem Globus Toolkit, wie dem Reliable File Transfer Service (RFT) oder dem Grid Resource Allocation Manager (GRAM). Der PDP fällt Autorisierungsentscheidungen basierend auf den Attributen aus der

Shibboleth Föderation.

- Die *GridShib SAML Tools* dienen dazu, SAML-Assertions in ein X.509 Proxyzertifikat einzubetten. So können die in der SAML-Assertion enthaltenen Attribute kryptographisch gegen Veränderung gesichert an die Grid-Ressource transportiert werden (Attribut-"push").

- Die *GridShib CA* ist eine webbasierte Online Certification Authority. Im Shibbolethkontext ist sie ein regulärer Service Provider (SP). Sie stellt, basierend auf einem Login über Shibboleth, kurzlebige X.509 Zertifikate (*Short Lived Certificates*, SLC) aus. Die Nutzerlegitimation erfolgt hierbei nicht wie bei regulären CAs über eine persönliche Überprüfung durch eine Registration Authority (RA), sondern durch den Shibboleth Login[4]. Da dies als weniger vertrauenswürdig angesehen wird, ist die Gültigkeit der ausgestellten Zertifikaten auf 1 000 000 Sekunden ($\approx 11\frac{1}{2}$ Tage) [13] beschränkt. Ein solcher Dienst wird auch *Short Lived Certificate Service* (SLCS) genannt.

- *GridShib for Shibboleth* ist eine Erweiterung des Shibboleth Identity Providers (IdP), der es der GridShib for Globus Toolkit Komponente ermöglicht, Nutzerattribute per „pull"-Verfahren zu beziehen. Diese Komponente wird nicht mehr unterstützt und auch nicht mehr benötigt, da Attribut-„push" verwendet wird. Der Einsatz einer solchen Komponente wäre kaum umsetzbar, da die Shibboleth Föderationen üblicherweise nicht Grid-spezifisch sind, und auf Grund ihrer hohen Sicherheitsanforderungen der Einsatz anwendungsspezifischer Softwarekomponenten auf den Identity Providern nicht sinnvoll ist.

In Abb. 3.11 ist der gesamte Vorgang eines Zugriffs mit Attribut-"pull" auf eine Grid-Ressource mittels eines Shibboleth-basierten Logins dargestellt. Die Schritte mit einer schwarzen Umrandung sind auf Nutzerinteraktion angewiesen, die übrigen werden automatisch ausgeführt. Bei Attribute-"push" werden die Attribute nicht direkt mit der Gridkomponente ausgetauscht, sondern werden als signierte Attribute Assertion in das Grid-Credential eingebunden und über an die Ressource übertragen.

Die Schritte sind im Einzelnen:

[4]Bei der Aufnahme in das Identity Management der Heimatorganisation ist üblicherweise ein ähnliches Vorgehen wir zur Ausstellung eines Zertifikats vorgeschrieben

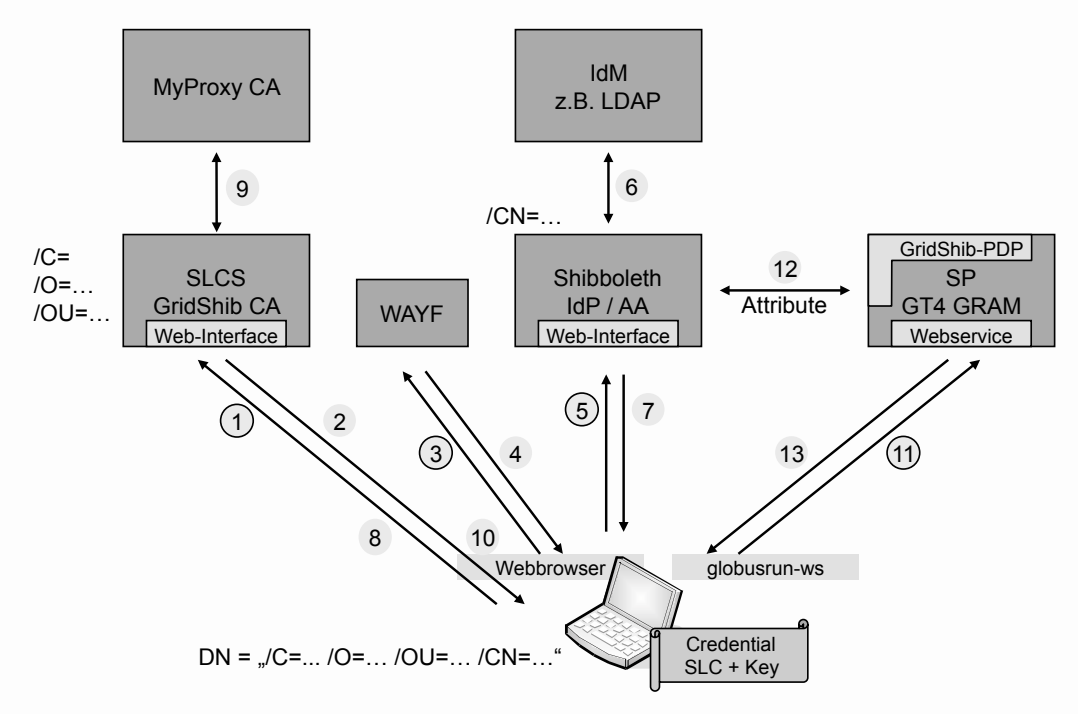

Abbildung 3.11: Ablauf eins Gridzugriffs mit GridShib

1. Der Nutzer greift mittels eines Webbrowsers auf die Webseite der GridShib CA zu, um ein Short Lived Certificate ausgestellt zu bekommen.

2. Die GridShib CA leitet den Nutzer an den Discovery Service der Shibbolethföderation weiter.

3. Der Nutzer wählt den Shibboleth Identity Provider seiner Heimatorganisation aus einer Liste aller verfügbaren IdPs aus.

4. Der Discovery Service leitet den Nutzer an den ausgewählten Identity Provider weiter.

5. Der Nutzer authentifiziert sich bei seinem IdP, bspw. mittels Benutzername/Passwort.

6. Der Identity Provider überprüft die eingegebenen Nutzercredentials über das verwendete Identity Management (IdM) System, bspw. einem LDAP Server. Hierbei werden auch die Nutzerattribute aus der Datenbank ausgelesen.

7. Der Identity Provider leitet den Nutzer wieder an den ursprünglichen Service

Provider, also die GridShib CA, weiter. Die kryptografisch gesicherte Authentifizierungsinformation wird über den Webbrowser des Nutzers an den Service Provider geliefert.

8. Nach der Überprüfung der Authentifizierungsinformation durch den Service Provider wird über Java Web Start eine Anwendung auf dem Rechner des Nutzers gestartet. Diese generiert ein Schlüsselpaar und einen Certificate Signing Request (CSR). Der CSR wird an die GridShib CA gesendet.

9. Wenn die GridShib CA MyProxy als Backend für die Signierung von Certificate Signing Requests verwendet, wird der CSR des Nutzers an diesen weitergeleitet und signiert. Alternativ kann die GridShib CA den CSR auch direkt mittels der OpenSSL Bibliotheken signieren. Die Verwendung von MyProxy ermöglicht den Einsatz von hardwarebasierten Schlüsselmanagementmodulen. Dies wird üblicherweise von Certificate Authorities verlangt, damit der private Schlüssel der CA besser geschützt ist.

10. Der Nutzer erhält das signierte Zertifikat (genauer: das Short Lived Certificate). Mit dem zuvor generierten privaten Schlüssel kann dieses Zertifikat nun wie ein gewöhnliches Credential verwendet werden. Die Java Web Start Anwendung auf dem Rechner des Nutzers speichert dieses Credential in `/tmp/x509_u<UID>`, analog zu herkömmlichen Proxycredentials.

11. Der Nutzer startet einen gewöhnlichen Gridjob, z. B. mittels `globusrun-ws`.

12. Optional: Die Grid-Ressource baut eine Verbindung zum Identity Provider auf und bezieht weitere Nutzerattribute.

13. Schließlich erhält der Nutzer das Ergebnis seines Grid Jobs.

GridShib bringt mehrere neue Funktionen in die Grid Sicherheitsinfrastruktur ein. Es müssen allerdings nicht alle der vier Komponenten eingesetzt werden. So ist es möglich, die GridShib CA ohne GridShib for Globus Toolkit zu verwenden und umgekehrt. Die GridShib CA vereinfacht das Zertifikathandling, während GridShib for GT die attributbasierte Autorisierung ermöglicht. Bei Verwendung von Attribut-"push" wird GridShib for Shibboleth nicht benötigt.

Die SAML-Tools werden nur bei Attribut-push benötigt und können entweder in der GridShib CA verwendet werden, um SAML-Assertions direkt in die Short Lived

Certificates einzubetten, oder auf dem Rechner des Nutzers, wenn er Shibbolethattribute in ein reguläres Proxyzertifikat einbetten möchte (d. h. ohne Verwendung der GridShib CA). Es gibt also viele Möglichkeiten, die Komponenten aus GridShib zu kombinieren und alle haben ihre spezifischen Vor- und Nachteile. Basierend auf diesen Funktionen und den weiteren hier betrachteten Komponenten wird in Kapitel 4.2 eine Architektur entworfen, die sowohl den hier definierten Anforderungen entspricht, als auch den Sicherheitspolicies der GridPMAs und der Ressourcenbetreiber.

Die Komponenten von GridShib werden nun einzeln betrachtet und im Hinblick auf ihre Eignung für den Einsatz in einer attributbasierten Authentifizierungs- und Autorisierungsinfrastruktur im Grid Computing analysiert.

Der Vorteil der GridShib Online CA ist, dass Gridnutzer keine regulären X.509 Zertifikate mehr benötigen. Das Zertifikatshandling wird vereinfacht, da das Beziehen des Zertifikats in den Workflow einer Jobabgabe eingebunden werden kann und der Nutzer sich nur bei seinem Identity Provider authentifizieren muss, wie er es auch aus anderen Anwendungen der Shibboleth Föderation kennt. Das im Hintergrund ein Zertifikat bezogen wird und eine Delegation von Proxyzertifikaten stattfindet, um den Gridjob zu authentifizieren und zu autorisieren, kann vollständig vor dem Nutzer verborgen werden. Ein gesondertes jährliches Beantragen des Zertifikats und das Handling von Zertifikat und privatem Schlüssel auf dem Clientrechner des Nutzers entfällt. Hierbei muss von der Online CA beachtet werden, dass ein Nutzer immer exakt denselben Distinguished Name in seinem Zertifikat erhält, damit die VO Managementsysteme den Nutzer authentifizieren können, um seine VO-Attribute auszustellen. Dies gilt ebenso für Grid-Ressourcen, die identitätsbasiert autorisieren und somit die Identität des Nutzers in ihren Polcies enthalten.

Aus der Sicht der bestehenden D-Grid Infrastruktur kann GridShib verwendet werden, um die bestehende DFN-AAI in die D-Grid Infrastruktur einzubinden. Der DFN-Verein betreibt bereits eine EUGridPMA akkreditierte Online CA für die Verwendung im Grid Computing, die allerdings bisher keine SAML Attribute Assertions in die Zertifikate einbindet. Diese Funktionalität kann parallel zur Einführung entsprechender Unterstützung auf den Grid-Ressourcen nachträglich eingeführt werden.

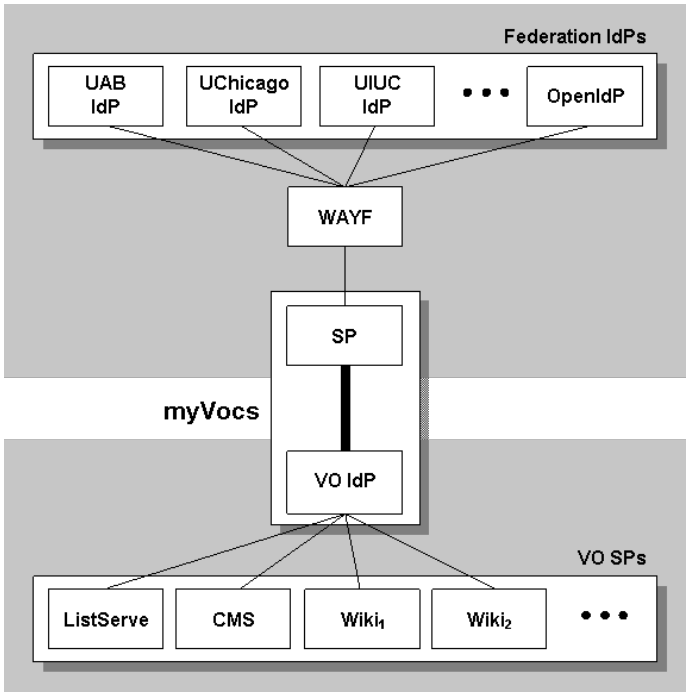

Abbildung 3.12: myVocs: Schematische Übersicht [8]

myVocs

myVocs basiert auf Shibboleth und ermöglicht es, in einer Shibboleth-basierten Infrastruktur das Konzept von Virtuellen Organisationen zu realisieren. myVocs ist dabei eine Brücke zwischen der Shibboleth Föderation und der Virtuellen Organisation. Der myVocs Service hat eine Service Provider Komponente, die in der Shibboleth Föderation auftritt. Das heißt, Nutzer können sich über den Login bei ihrer Heimatorganisation auf dem myVocs System einloggen. Auf der Seite der vom myVocs Dienst verwalteten Virtuellen Organisation tritt er als Identity Provider auf, das heißt, er kann Attribute Assertions mit Nutzerattributen an Service Provider in der VO ausstellen. Diese Service Provider werden VO SPs genannt. myVocs verwaltet die VO-Attribute der VO-Mitglieder, während die Campus-Attribute aus der Shibboleth Föderation übernommen werden. Beide Attributtypen werden in einer einzelnen von myVocs signierten Attribute Assertion ausgegeben. Es wird also Trust Proxying eingesetzt. Der schematische Aufbau ist in Abb. 3.12 dargestellt.

Wie Shibboleth unterstützt myVocs nur webbasierte Service Provider. Um Grid-Dienste als Service Provider einzubinden sind also wie bei Shibboleth Föderationen weitere Komponenten notwendig, beispielsweise GridShib oder ein spezieller Service

Provider, der die SAML Attribute Assertion eines angemeldeten Nutzers ausgibt.

Die Verwendung von myVocs mit Grid-Ressourcen, die auf der Grid Security Infrastructure basierte Sicherheitsfunktionen haben, ist von den Entwicklern nicht vorgesehen. Es ergeben sich also beim Einsatz von myVocs für das VO-Management dieselben Probleme, wie bei der Einbindung von Shibboleth Föderationen in das Grid.

VASH

Das Swiss Research Network SWITCH hat im Rahmen des EGEE-II Projekts VASH (VOMS Attributes from Shibboleth, [22]) implementiert, das Shibboleth und VOMS für die attributbasierte Autorisierung auf Grid-Ressourcen verwendet. Wie bei den Anforderungen aus dieser Arbeit sollen Campusattribute aus einer Shibboleth Föderation sowie VO-Attribute unterstützt werden. Für die Verwaltung der VO-Attribute wird VOMS eingesetzt. Zusätzlich werden die Campusattribute der VO-Mitglieder im VOMS Service gespeichert. Um die Campus Attribute aus der Shibboleth Föderation in den VOMS Service zu übertragen, wird der VASH Service eingesetzt. Dieser stellt einen regulären Service Provider in der Shibboleth Föderation dar. Wenn ein Nutzer sich auf ihm einloggt, wird eine SAML Attribute Assertion vom Identity Provider des Nutzers an den VASH Service übertragen. Dieser entnimmt die Attribute und überträgt sie in die Datenbank des VOMS Service. Der VOMS kann nun in einem Attribute Certificate gemeinsam Campus- und VO-Attribute der Nutzer ausstellen.

Das Vorgehen bei der Übertragung der Attribute in den VOMS ist in Abb. 3.13 dargestellt. Zuerst loggt der Nutzer sich bei dem VASH Service ein, indem er sich gegenüber seinem Identity Provider authentifiziert (Schritt 1-3). Nun wird eine SAML Attribute Assertion mit den Campus Attributen vom Identity Provider an den VASH Service ausgestellt (Schritt 4) und die bereits im VOMS gespeicherten Attribute werden bezogen (Schritt 5). Nun bekommt der Nutzer eine Liste aller verfügbaren Attribute angezeigt und muss bestätigen, dass er der Speicherung dieser Attribute im VOMS zustimmt (Schritt 6). Abschließend werden die Attribute wieder in der VOMS Datenbank gespeichert und stehen nun über den VOMS Service zur Verfügung (Schritt 7).

Es wird somit ähnlich wie bei myVocs Trust Proxying eingesetzt. Die Kombina-

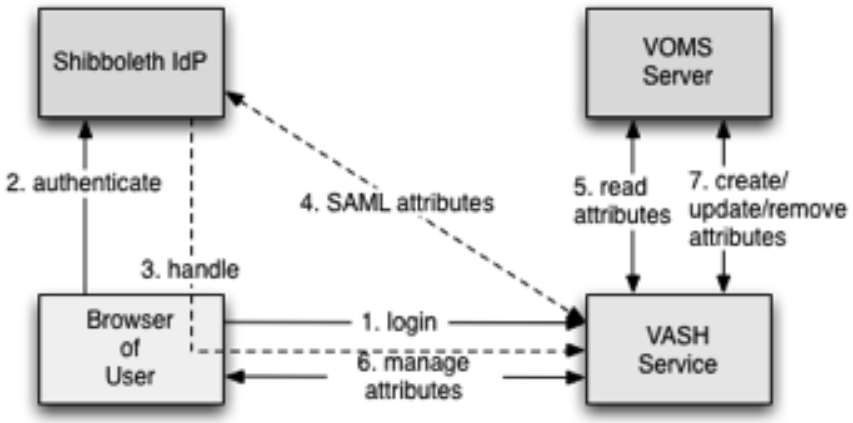

Abbildung 3.13: VASH: Schematische Übersicht [22]

tion von VASH und VOMS entspricht konzeptuell der Funktionalität von myVocs, wobei myVocs rein Shibboleth-basiert ist, während VASH Shibboleth und das X.509 PKI-basierte VOMS verbindet. Der Vorteil ist, dass das System somit direkt im Grid eingesetzt werden kann, da die Komponente zur Verwaltung der Virtuellen Organisation auf VOMS aufsetzt und somit bereits kompatibel mit gLite Grid Komponenten ist.

Die Übertragung der Attribute aus der Shibboleth Föderation in den VOMS über den VASH Service muss regelmäßig einmal pro Halbjahr durchgeführt werden. Danach läuft ihre Gültigkeit ab und der VASH Service löscht sie wieder aus der VOMS Datenbank. Der Nutzer erhält eine Aufforderung per Email, den Vorgang zu wiederholen, um die Attribute zu aktualisieren. Innerhalb des halben Jahres kann es aber passieren, dass der VOMS Campusattribute ausstellt, die im Identity Provider des Nutzers bereits abgelaufen sein können. Dies verstößt unter Umständen gegen die Policies sowohl der Shibboleth Föderation als auch der Grid-Infrastruktur.

Es ist geplant, mit zukünftigen Versionen diese Zeit dramatisch zu verkürzen, indem ein automatischer Refresh der Attribute in deutlich kürzeren Abständen erfolgt. Hierfür wird der Nutzer einmal einwilligen müssen, dass der VASH Service regelmäßigen Zugriff auf seine Campus Attribute bei seiner Heimateinrichtung bekommt.

3.6 Analyse von Fehlerquellen im Umgang mit komplexen Authentifizierungs- und Autorisierungsinfrastrukturen

Die Erfahrungen aus der täglichen Arbeit mit Grid-Infrastrukturen zeigen, dass der sichere Betrieb und die sichere Nutzung einer derartig komplexen IT-Umgebung einen großen Aufwand bedeutet. Sowohl die Einarbeitung in die Konzepte der Grid Security Infrastructure als auch in die korrekte Bedienung der entsprechenden APIs bei der Entwicklung gridfähiger Software sind wichtige Aufgaben, die aufgrund ihrer hohen Komplexität nicht immer korrekt ausgeführt werden.

Ein Ziel dieser Arbeit ist es, durch die Vereinfachung der von Programmierern von Grid-Anwendungen zu nutzenden Schnittstellen die Einstiegshürde soweit abzusenken, dass der Einarbeitungsaufwand und die Wahrscheinlichkeit einer Fehlbedienung soweit minimiert wird, dass ein sicherer Betrieb fachspezifischer Software in einer Grid-Infrastruktur ermöglicht wird.

Wenn bereits die Verwendung der Grid Sicherheitsfunktionen für die Entwickler von gridfähiger Software weitestgehend vereinfacht ist, wird auch die Verwendung der resultierenden Software durch die Nutzer vereinfacht, da die Entwickler die Funktionen weiter in ihrer Software kapseln und abstrahieren können. Die Gefahr, unnötig viele Aspekte der Grid Security an den Anwender zu exponieren, wird verringert.

Um diesen Beitrag leisten zu können, müssen zunächst jene Fehlerquellen analysiert werden, die zu unsicherer oder fehlerhafter Gridsoftware führen. Hierauf aufbauend kann dann ein API entworfen werden, das eine auf diese Probleme zugeschnittene Vereinfachung der Schnittstellen zu den Grid-Konzepten bietet.

3.6.1 Bösartige Angreifer

Grid-Infrastrukturen stellen aufgrund ihrer immensen Rechen- und Speicherkapazitäten sowie schneller Internetanbindungen ein ideales Ziel für Angreifer dar. Zum einen eignen sich die Ressourcen für illegale Aktivitäten wie die Verteilung von Spam, zum anderen sind auf den Ressourcen Daten von vielen verschiedenen Nutzern und Nutzergruppen vorhanden. Diese Daten enthalten potentiell wertvolle Inhalte aus aktuellen Forschungsprojekten. Bösartige Angreifer nutzen häufig Schwachstellen in der Programmierung von Internetdiensten oder schwache, per Brute Force erratbare

Passworte aus, um Zugriff auf Ressourcen zu erlangen. Schutz vor diesen Angriffsvektoren ist nicht das Ziel dieser Arbeit. Die hier erarbeiteten Ansätze erlauben eine konzeptionell vertrauenswürdige Authentifizierungs- und Autorisierungsinfrastruktur. Eine entsprechende Implementierung muss durch geeignete Verfahren der Software-Qualitätssicherung [53] gegen Programmierfehler abgesichert werden.

3.6.2 Fehlbedienung durch Entwickler

Entwickler aus Anwendungsdomänen, wie z. B. den Geodateninfrastrukturen, haben Schwierigkeiten, die Anforderungen der Sicherheit im Grid korrekt umzusetzen. Sowohl auf konzeptioneller wie auch auf technischer Ebene ist die Komplexität derart hoch, dass selbst bei erfahrenen Entwicklern entweder unsichere Software entsteht oder dass das Grid nicht erfolgreich eingebunden werden kann. Es besteht die Gefahr, dass die Software die Sicherheitsfunktionen der Grid Middlewares nicht korrekt bedient. Dies führt dazu, dass Angreifer die gesamte Grid-Infrastruktur kompromittieren können, indem sie fehlerhafte domänenspezifische Dienste angreifen. Zusätzlich besteht in Grid-Infrastrukturen mit Accounting und Billing die Möglichkeit, dass die Nutzung von Grid-Infrastruktur falsch abgerechnet wird. Dies kann auch ohne bösartige Angreifer zu Verdienstausfällen bei den Ressourcen-Providern führen, da Gridjobs den falschen Personen oder Einrichtungen zugeordnet werden, die die Rechnung dann nicht bezahlen.

In der Informatik ist es eine bewährte Herangehensweise, große Komplexität durch Verwenden einer Abstraktion zu kapseln und die Funktionalität über ein vereinfachtes Interface bereitzustellen. Dieser Ansatz erleichtert die Entwicklung sicherer Software durch domänenspezifische Entwickler. Daher wird eine solche Abstraktion in dieser Arbeit entwickelt, um die genannten Gefahren zu vermindern.

3.6.3 Fehlbedienung durch Anwender

Fehlbedienung durch Anwender kann in zwei Kontexten auftreten: Zum einen können Anwender ihre Credentials durch Unachtsamkeit unbefugten Dritten zugänglich machen oder fehlerhafte Policies für ihre Daten im Grid vergeben, wenn diese durch Discretionary Access Control im Einflussbereich der Nutzer liegen.

Beide Probleme können in der Anwendungsentwicklung von gridifizierter Clientsoftware angegangen werden. Sinnvolle Defaulteinstellungen und weitestgehendes

Verbergen der Crendentialverwaltung vor dem Nutzer in der Clientsoftware sind möglich, setzen aber entweder erfahrene Gridsoftware-Entwickler voraus oder, wie oben beschrieben, entsprechende Abstraktionen für domänenspezifische Softwareentwickler. Die Entwicklung von Clientsoftware mit Hilfe der in dieser Arbeit definierten Abstraktion führt dazu, dass Entwickler aufgrund von Unkenntnis die Sicherheitsfunktionen nicht in unnötigem Umfang an den Nutzer exponieren. Daher wird auch diese Fehlerquelle minimiert.

Kapitel 4

Eine vertrauenswürdige Architektur für die attributbasierte Autorisierung im Grid Computing

4.1 Übersicht

Basierend auf den bisher untersuchten Technologien werden in diesem Kapitel verschiedene Ansätze designed und auf ihre Vertrauenswürdigkeit analysiert, die die Anforderungen an die attributbasierte Autorisierung im Grid Computing erfüllen. Es wird der Ansatz ausgewählt, der die Anforderungen erfüllt und keine Vertrauensprobleme verursacht. In Kapitel 5 werden die Maßnahmen diskutiert, die notwendig sind, um den ausgewählten Ansatz in einer realen Grid-Infrastruktur umzusetzen.

Um in komplexen Grid-Infrastrukturen sichere Software entwickeln zu können, ist eine Abstraktion notwendig, die diese Komplexität kapselt und den Entwicklern die Anpassung ihrer bisher nicht gridfähigen Software ermöglicht. Ein API, das diese Abstraktion bereitstellt, wird im zweiten Teil dieses Kapitels konzipiert.

4.2 Entwicklung von Ansätzen für die attributbasierte Autorisierung mit mehreren Attribute Authorities

Keiner der betrachteten Ansätze erfüllt einzeln betrachtet die Anforderungen, die in dieser Arbeit erhoben werden. In den jeweiligen Gridumgebungen, in deren Rahmen die Programme entworfen und implementiert wurden, werden diese meist einzeln eingesetzt, das heißt sie decken den gesamten Anforderungsraum dieser Grid-Infrastrukturen ab.

Eine komplette Neuentwicklung basierend auf den Anforderungen ist allerdings auch nicht notwendig, da die existierenden Ansätze derart neuartig kombiniert werden können, dass sie diese erfüllen. Es ist das Ziel dieses Kapitels, diese Kombinationen zu beschreiben und zu nachzuweisen, dass sie die erhobenen funktionalen Anforderungen umsetzen und ob sie dabei die identifizierten Aspekte der Vertrauenswürdigkeit der Übermittlung der Attribute erfüllen.

Es werde drei Ansätze betrachtet, die geeignet sind, Campus Attribute aus einer Shibboleth Föderation in eine Grid-Infrastruktur einzubringen. Diese sind das Shibboleth-basierte myVocs, das X.509-basierte VOMRS/VOMS System und die direkte Einbettung der Campus Attribute in ein kurzlebiges X.509 Zertifikat durch eine GridShib online CA. Diese Ansätze lassen sich zum Teil mit dem beschriebenen

Trust Proxying und zum Teil ohne Trust Proxying einsetzen.

Basierend auf der Analyse der existierenden VO Management-Technologien ergeben sich mehrere Möglichkeiten, diese Technologien mit den oben genannten Technologien für die Einbindung von Campus-Attributen zu kombinieren, um das Ziel der Autorisierung basierend auf Campus- und VO-Attributen zu erreichen.

Da viele Grid Communities planen, in Zukunft für ihre wachsenden Nutzerzahlen auf die Verwendung einer Online CA und kurzlebige Zertifikate zu setzen, wird dieser Fall im Folgenden als Grundlage angenommen. Es wird jeweils darauf eingegangen, welche Änderungen sich bei der Nutzung herkömmlicher lang laufender Zertifikate ergeben.

4.2.1 SAML-Basiertes VO-Management: MyVocs mit Trust Proxying

Bei diesem Ansatz wird myVocs derart eingesetzt, dass Attribute von der Campus AAI von der Shibboleth Service Provider Komponente von myVocs bezogen werden. Diese werden dann von der Shibboleth Identity Provider Komponente als Attribute Authority der Grid Virtuellen Organisation wieder ausgegeben (siehe Kapitel 3.5.4). Der vollständige Workflow ist in Abb. 4.1 dargestellt.

Falls ein Nutzer ein langlebiges Zertifikat besitzt und somit kein kurzlebiges Zertifikat von einem Short Lived Certificate Service (d. h. einer Online CA) benötigt, beginnt folgender Workflow bei Schritt 7. Die Schritte aus Abb. 4.1 sind im Einzelnen:

1. Der Nutzer greift mittels Webbrowser auf die GridShib CA zu, um ein kurzlebiges Zertifikat zu beantragen.

2. Der Nutzer wird an den Discovery Service der Shibboleth Föderation weitergeleitet.

3. Der Nutzer wählt seine Heimatorganisation (genauer: den Identity Provider seiner Heimatorganisation) aus der Liste aus und wird zur ihr weitergeleitet. Der Nutzer wird authentifiziert und wird bei erfolgreicher Authentifizierung an die GridShib CA zurückgeleitet.

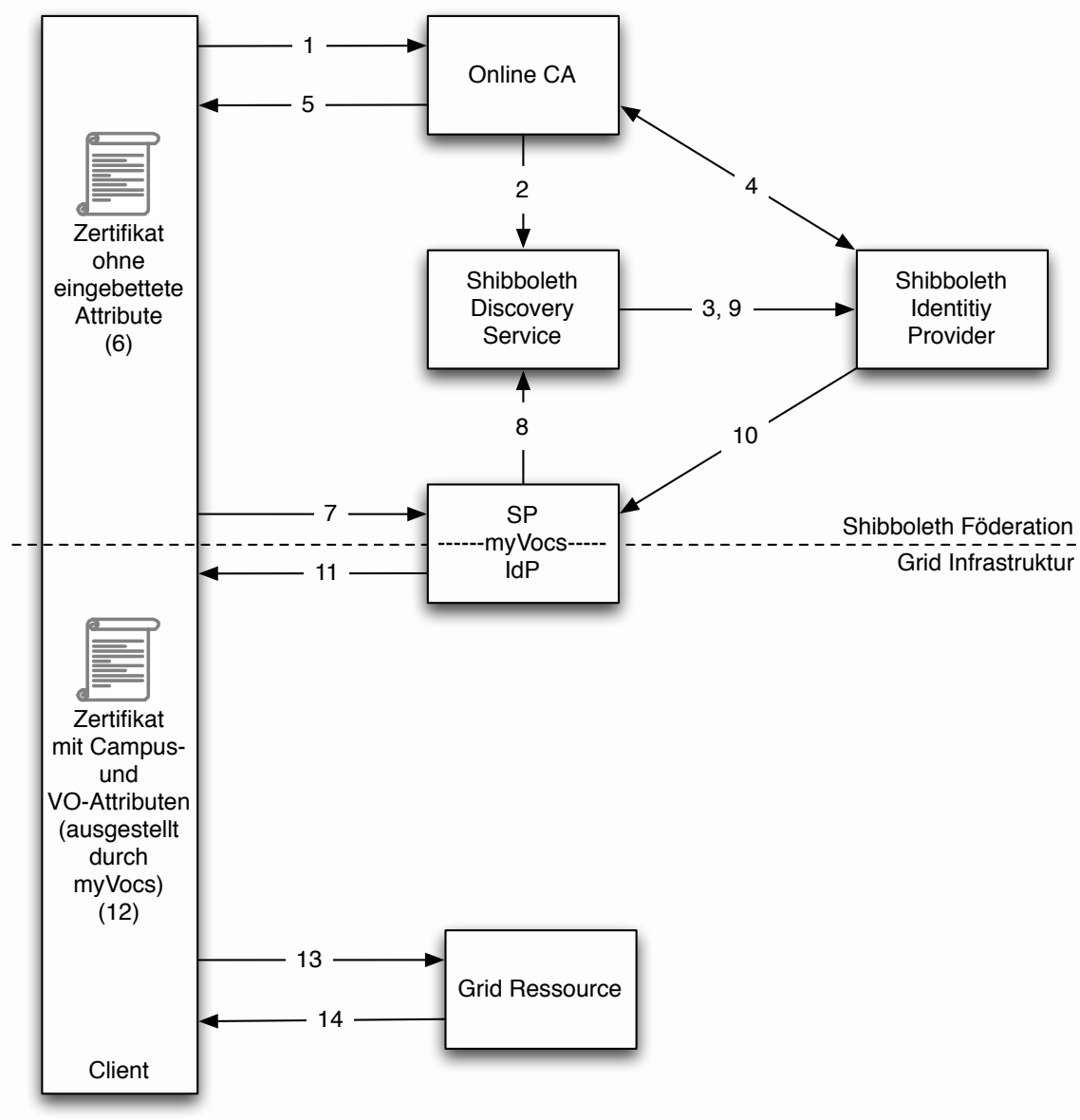

Abbildung 4.1: Workflow mit myVocs als Trust Proxy, Nutzer ohne langlebiges Zertifikat

4. Basierend auf der erfolgreichen Authentifizierung über den Identity Provider und eventuell zusätzlich übermittelter Attribute vom Identity Provider überprüft die Online CA, ob der Nutzer ein kurzlebiges Zertifikat erhalten darf und stellt es gegebenenfalls aus.

5. Das kurzlebige Zertifikat wird an den Nutzer übertragen.

6. Der Nutzer besitzt nun ein gültiges kurzlebiges Zertifikat, das allerdings noch keine weiteren Attribute enthält.

7. Der Nutzer greift nun auf den myVocs Service zu, um sowohl seine VO- als auch seine Campusattribute zu beziehen.

8. Der Nutzer wird für die Authentifizierung von der Service Provider Komponente von myVocs wieder an den Discovery Service weitergeleitet...

9. ... und weiter zum Identity Provider seiner Heimateinrichtung...

10. ... und zurück zum myVocs Service. Hierbei können Nutzerattribute vom Identity Provider an den myVocs Service übergeben werden.

11. Die Identity Provider Komponente des myVocs Service stellt eine signierte SAML Assertion aus, welche die vom Identity Provider der Heimatorganisation erhaltenen Campusattribute und die im myVocs gespeicherten VO-Attribute enthält. Der Nutzer kann diese signierte Attribute Assertion mittels der GridShib SAML Tools in sein Grid-Proxycredential einbetten.

12. Der Nutzer besitzt nun ein Grid-Credential, welches sowohl Campus- als auch VO-Attribute in Form einer von myVocs signierten SAML Assertion enthält.

13. + 14. Der Nutzer kann nun auf Grid-Ressourcen zugreifen, indem er sich mit dem Proxycredential authentifiziert. Grid-Ressourcen haben die Möglichkeit, Autorisierungsentscheidungen auf Basis der eingebetteten Attribute zu fällen.

Analyse

MyVocs als Trust Proxy einzusetzen vereinfacht das Trustmanagement auf den Ressourcen: Die Grid-Ressourcen müssen nicht Bestandteil der Shibboleth Föderation sein bzw. alle Identity Provider der Föderation kennen (d. h. ihre Zertifikate und ihren Geltungsbereich, um ausgestellte Attribute Assertions validieren zu können).

Der größte Nachteil dieser Lösung ist die Verwendung von Trust Proxying (vgl. Kapitel 3.4.2). Es muss anhand der Sicherheitsanforderungen und Policies sowohl der Shibboleth Föderation als auch der Grid-Infrastruktur entschieden werden, ob dies akzeptabel ist.

Sowohl bei Nutzung eines langlebigen Zertifikats als auch bei der Nutzung der Online CA ist das Binden der X.509 Identität auf die Attribute Assertion sowohl für die Campus- als auch für die VO-Attribute nicht automatisch gegeben. myVocs kennt den Nutzer nur unter seiner Shibboleth-Identität und stellt Campus- und VO-Attribute entsprechend für diese Identität aus. Eine der in Kapitel 3.4.3 vorgestellten Lösungen muss eingesetzt werden, wovon als einzige das Verwalten des X.509 Distinguished Names in myVocs praktikabel ist. Die korrekte Eintragung des Distinguished Names muss allerdings überprüft werden, was wieder zusätzlichen Aufwand für den Nutzer und den VO Administrator bedeutet. Über den normalen Workflow einer Aufnahme eines Nutzers in die VO kann dies nicht automatisch abgebildet werden.

4.2.2 SAML-Basiertes VO-Management: MyVocs ohne Trust Proxying

MyVocs kann auch ohne den Einsatz von Trust Proxying eingesetzt werden. Hierbei werden die Campusattribute nicht durch den myVocs Service neu ausgestellt, sondern die originale SAML Assertion vom ausstellenden Identity Provider beibehalten. Diese wird entweder vom Nutzer in ein von seinem langlebigen Zertifikat abgeleitetes Proxyzertifikat eingebunden oder von einer Online CA in eine kurzlebiges Zertifikat eingebettet.

Der myVocs Dienst stellt in dieser Konstellation nur die VO-Attribute aus, für die er die zuständige Attribute Authority ist. Die VO-Attribute werden in ein vom ursprünglichen Zertifikat abgeleitetes Proxyzertifikat eingebunden. In der Zertifikatskette befinden sich somit zwei SAML Assertions in zwei unterschiedlichen (Proxy-)Zertifikaten. Eine davon beinhaltet die Campusattribute, die andere die VO-Attribute. Da die gesamte Zertifikatskette vorhanden sein muss, um die Gültigkeit zu validieren, sind auch beide SAML Assertions auf allen Grid-Ressourcen vorhanden, auf denen der Job ausgeführt wird. Der Workflow für einen Nutzer ohne langlebiges Zertifikat ist in Abb. 4.2 dargestellt.

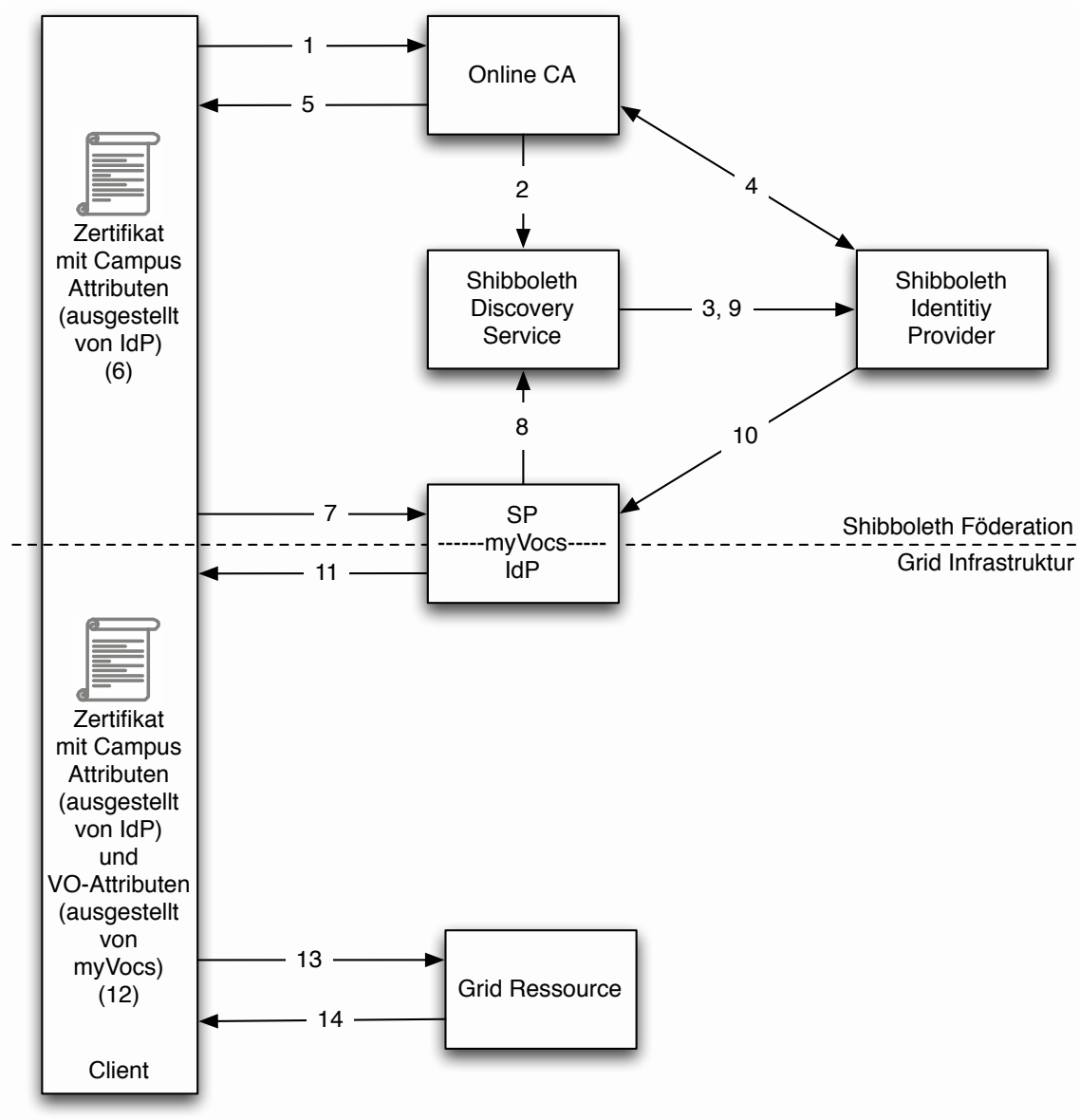

Abbildung 4.2: Workflow mit myVocs ohne Trust Proxying, Nutzer ohne langlebiges Zertifikat

Die Schritte sind im Einzelnen:

1. Der Nutzer greift mittels Webbrowser auf die Online CA zu, um ein kurzlebiges Zertifikat zu beantragen.

2. Der Nutzer wird an den Discovery Service der Föderation weitergeleitet.

3. Der Nutzer wählt seine Heimatorganisation aus und wird zum entsprechenden Identity Provider weitergeleitet.

4. Nach erfolgreicher Authentifizierung durch den IdP wird der Nutzer wieder an die Online CA weitergeleitet. Die Online CA stellt ein kurzlebiges Zertifikat aus und bettet vom Identity Provider bezogene Attribute in Form der originalen signierten SAML Assertion in dieses ein.

5. Das kurzlebige Zertifikat mit den Campusattributen wird an den Nutzer übertragen.

6. Auf dem Rechner des Nutzers steht nun ein kurzlebiges Grid-Credential bereit, das seine Campusattribute enthält.

7. Der Nutzer greift auf den myVocs Service zu, um seine VO-Attribute zu erhalten und diese dann mittels der GridShib SAML Tools in ein von seinem Credential abgeleitetes Proxycredential einzubinden.

8. Der Nutzer wird zur Authentifizierung wieder an den Discovery Service geleitet.

9. Falls die Browsersession nicht unterbrochen wurde, wird der existierende Authentifizierungskontext erkannt und der Nutzer sofort an den Identity Provider weitergeleitet.

10. Der Identity Provider erkennt den Sicherheitskontext oder fragt erneut nach der Nutzerauthentifizierung. Der Nutzer wird wieder an den myVocs Service weitergeleitet, bei dem er nun angemeldet ist. Optional können zusätzliche Attribute übertragen werden, aber diese werden nur zur Autorisierung auf dem myVocs Service verwendet und nicht vom myVocs Service für die Verwendung im Grid neu ausgestellt.

11. Die Identity Provider Komponente des myVocs Service stellt dem Nutzer eine SAML Assertion aus, die seine VO-Attribute enthält. Der Nutzer bindet diese Assertion mittels der GridShib SAML Tools in ein weiteres abgeleitetes Proxycredential ein.

12. Der Nutzer hat nun ein Proxycredential, das sowohl Campus- als auch VO-Attribute enthält. Diese sind als zwei getrennte signierte SAML Assertions kodiert, die in diesem Falle von verschiedenen Identity Providern signiert sind (IdP der Heimatorganisation für die Campusattribute, IdP Komponente des myVocs Service für die VO-Attribute)

13. + 14. Der Nutzer kann nun auf das Grid zugreifen und dazu ein Grid-Credential mit VO- und Campusattribute verwenden.

Nutzer mit langlebigen Zertifikaten müssen die Attribute Assertion mit ihren Campusattributen direkt von dem zuständigen Identity Provider ihrer Heimateinrichtung beziehen und diese dann in ein Proxyzertifikat einbetten. Anschließend geht der Workflow bei Schritt 7 weiter. Es gelten die in Kapitel 3.4.3 diskutierten Voraussetzungen für das Binding der Attribute Assertion an das Proxyzertifikat.

Analyse

In diesem Ansatz wird myVocs ausschließlich als VO Management System eingesetzt und stellt somit nur signierte Attribute Assertions aus, die Attribute enthalten, für die dieser Dienst die zuständige Attribute Authority ist. Trust Proxying findet somit nicht statt und alle Grid-Ressourcen können die signierten Attribute mit den Zertifikaten der jeweils zuständigen Attribute Authority validieren.

Während dies die Trustprobleme des Trust Proxying löst, müssen dafür alle Grid-Ressourcen, die mittels Campusattributen Autorisierungsentscheidungen durchführen müssen, alle Identity Provider der Campusföderation kennen, d. h. ihre Attributscopes und Zertifikate.

Bei dieser Lösung ist das Binden der X.509 Identität an die ausgestellten VO-Attribute wie auch im vorherigen Ansatz nicht automatisch gegeben. VO-Attribute enthalten nicht den X.509 DN, da das VO Management über die Shibboleth-Identität durchgeführt wird. In diesem Fall kann das Problem gelöst werden, indem die Online CA den eduPersonPrincipalName mit als Feld in das ausgestellte Zertifikat aufnimmt.

Ein Problem beim Binden der Campusattribute an die X.509 Identität des kurzlebigen Zertifikats ergibt sich bei diesem Ansatz nicht, da das Ausstellen des Zertifikats mit der X.509 Identität auf Basis der Authentifizierung beim Identity Provider der Heimatorganisation erfolgt. Nutzer mit klassischen langlebigen Zertifikaten können daher allerdings nicht unterstützt werden.

4.2.3 X.509-basiertes VO-Management: VOMRS/VOMS mit Trust Proxying durch GridShib CA

Eine GridShib Online CA kann eingesetzt werden, um Nutzerattribute vom Identity Provider der Heimatorganisation des Nutzers in ein kurzlebiges Zertifikat einzubetten und diese dabei neu zu signieren [33].

Der Ablauf besteht aus folgenden Schritten:

1. Der Nutzer greift auf die Online CA zu, um ein kurzlebiges Zertifikat zu beziehen.
2. Der Nutzer wird an den Discovery Service der Föderation weitergeleitet.
3. Der Nutzer wählt den Identity Provider seiner Heimatorganisation aus und wird an diesen weitergeleitet. Der Nutzer authentifiziert sich bei diesem, wird wieder an die Online CA weitergeleitet und Nutzerattribute werden vom IdP an die Online CA übertragen.
4. Die Online CA bettet die empfangenen Attribute in das auszustellende kurzlebige Zertifikat ein, wobei diese in eine neue SAML Assertion verpackt werden, die die Online CA signiert[1].
5. Die Online CA liefert das kurzlebige Zertifikat an den Nutzer aus.
6. Der Nutzer besitzt nun ein gültiges kurzlebiges Zertifikat mit einer eingebetteten SAML Assertion. Diese enthält die Campusattribute des Nutzers.
7. Der Nutzer fordert seine VO-Attribute von einem VOMS Service an.

[1] Das Signieren der SAML Assertion kann in diesem Fall sogar unterbleiben, da sie Bestandteil des Zertifikats wird, das von der Online CA signiert wird. In jedem Fall muss die Signatur des Identity Providers entfernt werden.

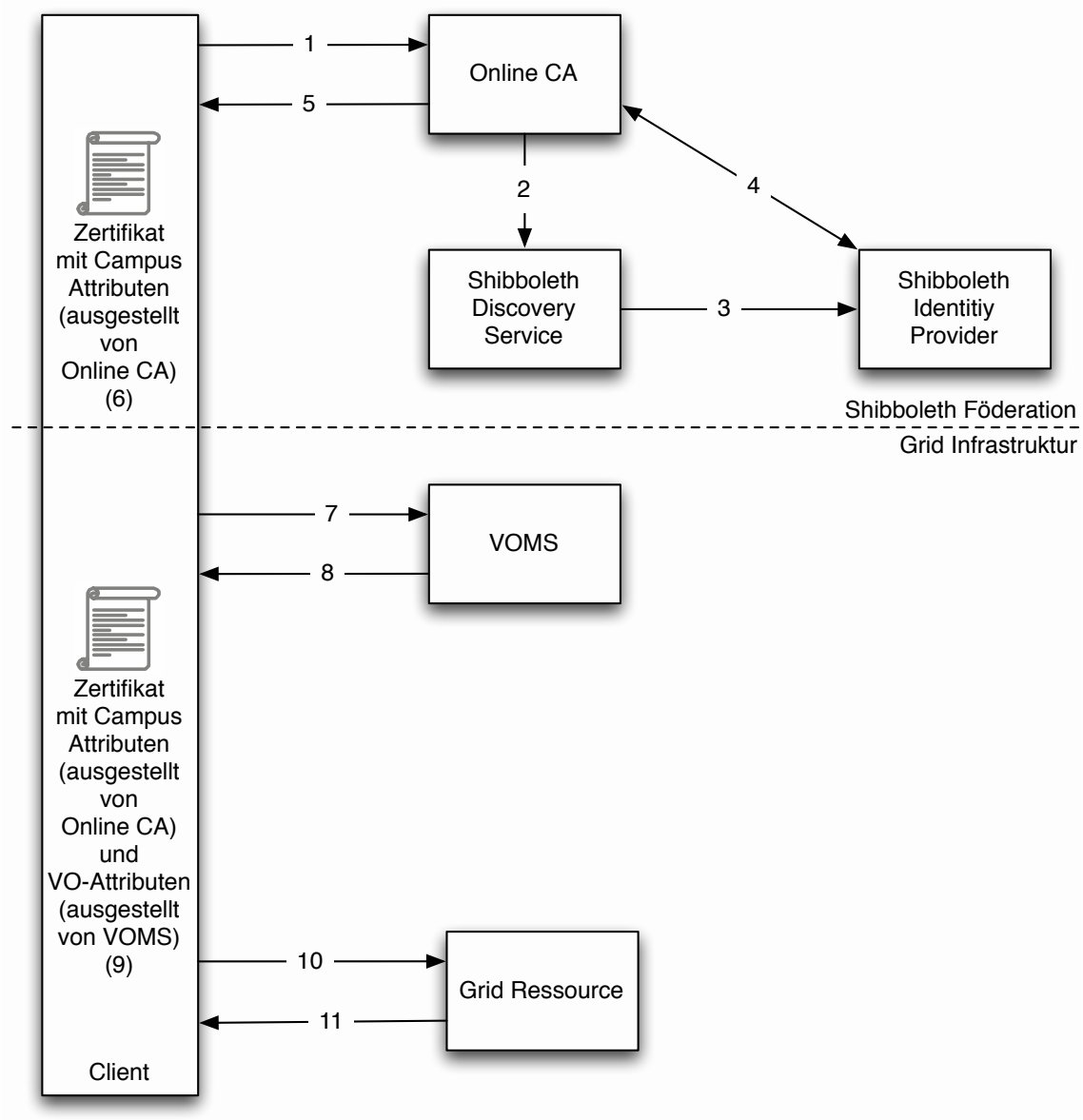

Abbildung 4.3: Workflow mit VOMS mit Trust Proxying durch Online CA, Nutzer ohne langlebiges Zertifikat

8. Die VO-Attribute werden in Form eines Attribute Certificates oder einer SAML Attribute Assertion in ein von dem kurzlebigen Zertifikat abgeleitetes Proxyzertifikat eingebunden.

9. Dem Nutzer steht nun ein Proxycredential zur Verfügung, das eine SAML Attribute Assertion mit seinen Campusattributen enthält und ein Attribute Certificate oder eine SAML Attribute Assertion mit seinen VO-Attributen.

10. + 11. Der Nutzer kann nun auf die Grid-Ressourcen mit dem Proxycredential zugreifen.

Analyse

Das verwendete Trust Proxying führt zu den bereits beschriebenen Vertrauensproblemen. Die Kodierung der Attribute entspricht bei Verwendung des SAML-VOMS derjenigen der vorhergehenden Ansätze (nur SAML Assertions). Bei Verwendung des klassischen VOMS wird mit den Attributzertifikaten ein weiterer Kodierungstyp für signierte Attribute verwendet. Dieser muss von den Grid-Ressourcen unterstützt werden, wenn sie anhand von VO-Attributen autorisieren. Die Campusattribute sind weiterhin als SAML Assertion vorhanden und müssen dementsprechend unterstützt werden.

Für Nutzer mit kurzlebigen Zertifikaten stellt dieser Ansatz kein Problem für das Binden der Attribute an die X.509 Identität des Nutzers dar. Die Campusattribute werden von der Online CA eingebunden, die das Zertifikat auf Basis von Attributen aus der Shibboleth-Föderation ausstellt. Die VO-Attribute sind durch die Verwendung von VOMS automatisch an die X.509 Identität des Nutzers gebunden.

Für Nutzer mit langlebigem Zertifikat ist diese Variante nicht einsetzbar, da sie keine Möglichkeit haben, ihre Campusattribute in ein Grid-Credential vertrauenswürdig neu zu signieren und einzubauen.

4.2.4 X.509-basiertes VO-Management: VOMRS/VOMS mit Trust Proxying durch VASH

Bei der Verwendung von *VOMS Attributes from Shibboleth* (VASH) werden Attribute aus einer Shibboleth Föderation, also die Campusattribute, periodisch für die Gridbenutzer vom Identity Provider der Heimatorganisationen der Nutzer in

den für ihre Virtuelle Organisation zuständigen VOMS Service übertragen. Dies kann beispielsweise einmal pro akademischem Semester geschehen. Danach stellt der VOMS Service diese Attribute neben den eigentlichen VO-Attributen aus. Da er nicht die Attribute Authority für die Campusattribute ist, verwendet diese Lösung somit Trust Proxying. Der Ablauf aus Nutzersicht ist in Abb. 4.4 dargestellt. Hierbei wird im Gegensatz zu den anderen Ansätzen keine Online CA verwendet. Dies dient der Übersichtlichkeit, da in diesem Ansatz kein Unterschied zwischen Nutzern von langlebigen Zertifikaten und kurzlebigen Zertifikaten besteht. Nutzer kurzlebiger Zertifikaten müssen vor dem dargestellten Workflow ein kurzlebiges Zertifikat ohne weitere Einbettung von Attributen bei der Online CA beziehen.

Folgende Schritte muss der Nutzer durchführen, um seine Attribute in den VOMS Service zu übertragen:

a: Der Nutzer greift auf den VASH Service (Service Provider in der Shibboleth Föderation) zu, um den Transfer der Campusattribute zu initiieren.

b: Der Nutzer wird an den Discovery Service weitergeleitet und wählt den Identity Provider seiner Heimatorganisation aus.

c: Der Nutzer wird an den Identity Provider seiner Heimatorganisation weitergeleitet und authentifiziert sich.

d: Nach erfolgreicher Authentifizierung wird der Nutzer wieder an den VASH Service weitergeleitet. VASH zeigt die Attribute an, die er von dem Identity Provider erhalten hat. Der Nutzer kann nun eine Auswahl treffen, welche Attribute er zum VOMS Service übertragen möchte.

e: Die ausgewählten Attribute werden in der VOMS Datenbank als generische Attribut-Wert-Paare gespeichert.

Der Nutzer kann nun jederzeit eine Gridsession starten, indem er folgende Schritte ausführt:

1: Der Nutzer erzeugt ein Proxyzertifikat und fragt hierbei seine Attribute von einem VOMS Service ab.

2: Der VOMS Service liefert die Nutzerattribute (d. h. Campus- und VO-Attribute) als Attribute Certificate oder SAML-Assertion aus. Das Attribute Certificate bzw. die SAML-Assertion wird in das Proxyzertifikat eingebettet.

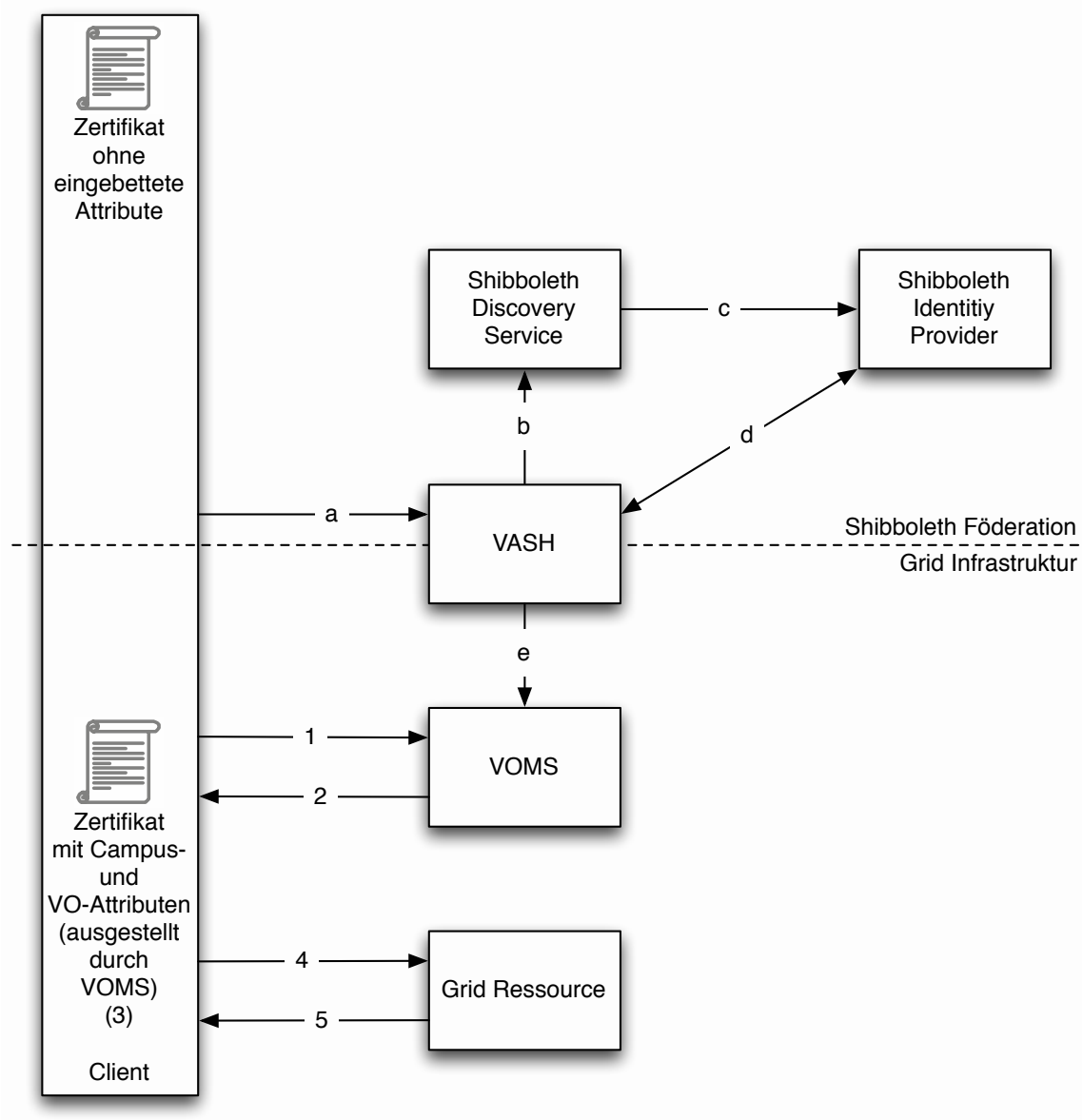

Abbildung 4.4: Workflow mit VOMS mit Trust Proxying durch VASH, Nutzer mit langlebigem Zertifikat

3: Der Nutzer besitzt nun ein Proxyzertifikat, das Campus- und VO-Attribute in einem einzelnen Attribute Certificate oder einer einzelnen SAML-Assertion enthält.

4: + 5. Der Nutzer kann hiermit nun auf Grid-Ressourcen zugreifen. Die Autorisierung kann anhand seines X.509 Distinguished Names oder der eingebetteten Attribute durchgeführt werden.

Analyse

VASH löst einige Probleme einer auf Campus- und VO-Attributen basierenden Authentifizierungs- und Autorisierungsinfrastruktur. So ist die Verwaltung der beiden Typen von Attributen vollständig getrennt und findet ausschließlich in der Organisationseinheit statt, wo diese Attribute ihre Bedeutung erhalten. Weiterhin muss keine Shibboleth- oder Gridkomponente sowie der VOMS Service angepasst werden, da dieses Vorgehen für diese Komponenten transparent ist. Der Hauptnachteil liegt bei diesem Ansatz zum einen im Einsatz von Trust Proxying. Zum anderen werden die Campusattribute im VOMS Service über längere Zeit gespeichert, d. h. eventuelle Änderungen in der Shibboleth Föderation werden in der Grid VO zunächst nicht bemerkt. Ein solches Verhalten wird unter Umständen von den Policies der Shibboleth Föderation und/oder der Grid-Infrastruktur nicht zugelassen.

Aufgrund des Trust Proxying für die Campusattribute ist hier kein Binden der Attribute an die Shibboleth-Identität des Nutzers notwendig. Da die VO-Attribute gemeinsam mit den Campusattributen im VOMS verwaltet werden, sind diese auch automatisch an die X.509 Identität des Nutzers gebunden. Der Ansatz unterstützt Nutzer mit langlebigen und Nutzer mit kurzlebigen Zertifikaten gleichermaßen.

4.2.5 X.509-basiertes VO-Management: VOMRS/VOMS ohne Trust Proxying

In diesem Ansatz wird VOMS ausschließlich als VO Management-Tool eingesetzt, d. h. VOMS kommt mit den Campusattributen nicht in Kontakt. Wie im Ansatz mit der GridShib CA als Trust Proxy werden die Campusattribute von der Online CA als SAML Assertion an den Nutzer herausgegeben (falls der Nutzer diese in ein Proxyzertifikat einbinden möchte) oder die Online CA bindet die SAML Assertion direkt in ein ausgestelltes kurzlebiges Zertifikat ein. Abweichend von obiger Lösung

wird hier aber die SAML Assertion nicht neu signiert, sondern die Originalassertion inklusive Signatur des Identity Providers des Nutzers verwendet.

Das Vorgehen besteht im Einzelnen aus folgenden Schritten:

1. Der Nutzer greift auf die Online CA zu, um ein kurzlebiges Zertifikat zu beziehen.

2. Der Nutzer wird an den Discovery Service weitergeleitet, wo er seine Heimatorganisation auswählt.

3. Der Nutzer wählt seine Heimatorganisation aus und wird an deren Identity Provider weitergeleitet. Der Nutzer authentifiziert sich bei diesem und wird wieder an die Online CA weitergeleitet.

4. Die Online CA bettet die vom Identity Provider erhaltenen Attribute in das auszustellende kurzlebige Zertifikat ein. Hierbei wird die originale, signierte SAML Assertion unverändert verwendet.

5. Das kurzlebige Zertifikat mit den Attributen wird an den Nutzer ausgeliefert.

6. Der Nutzer besitzt nun ein kurzlebiges Zertifikat, das von der Online CA signiert ist. In diesem Zertifikat sind die Campusattribute des Nutzers eingebunden. Diese sind als vom Identity Provider signierte SAML Assertion kodiert.

7. Der Nutzer leitet von dem kurzlebigen Zertifikat ein Proxyzertifikat ab. Hierbei fragt er seine VO-Attribute vom VOMS Service ab.

8. Die VO-Attribute werden vom VOMS Service entweder als signierte SAML Assertion oder als signiertes Attribute Certificate ausgeliefert und in das neu erstellte Proxyzertifikat eingebunden.

9. Der Nutzer besitzt nun ein Proxycredential mit einer SAML Assertion mit seinen Campusattributen und einer SAML Assertion oder einem Attribute Certificate mit seinen VO-Attributen.

10. + 11. Hiermit kann der Nutzer nun auf Gridkomponenten zugreifen und anhand der in das Proxycredential eingebetteten Attribute autorisiert werden.

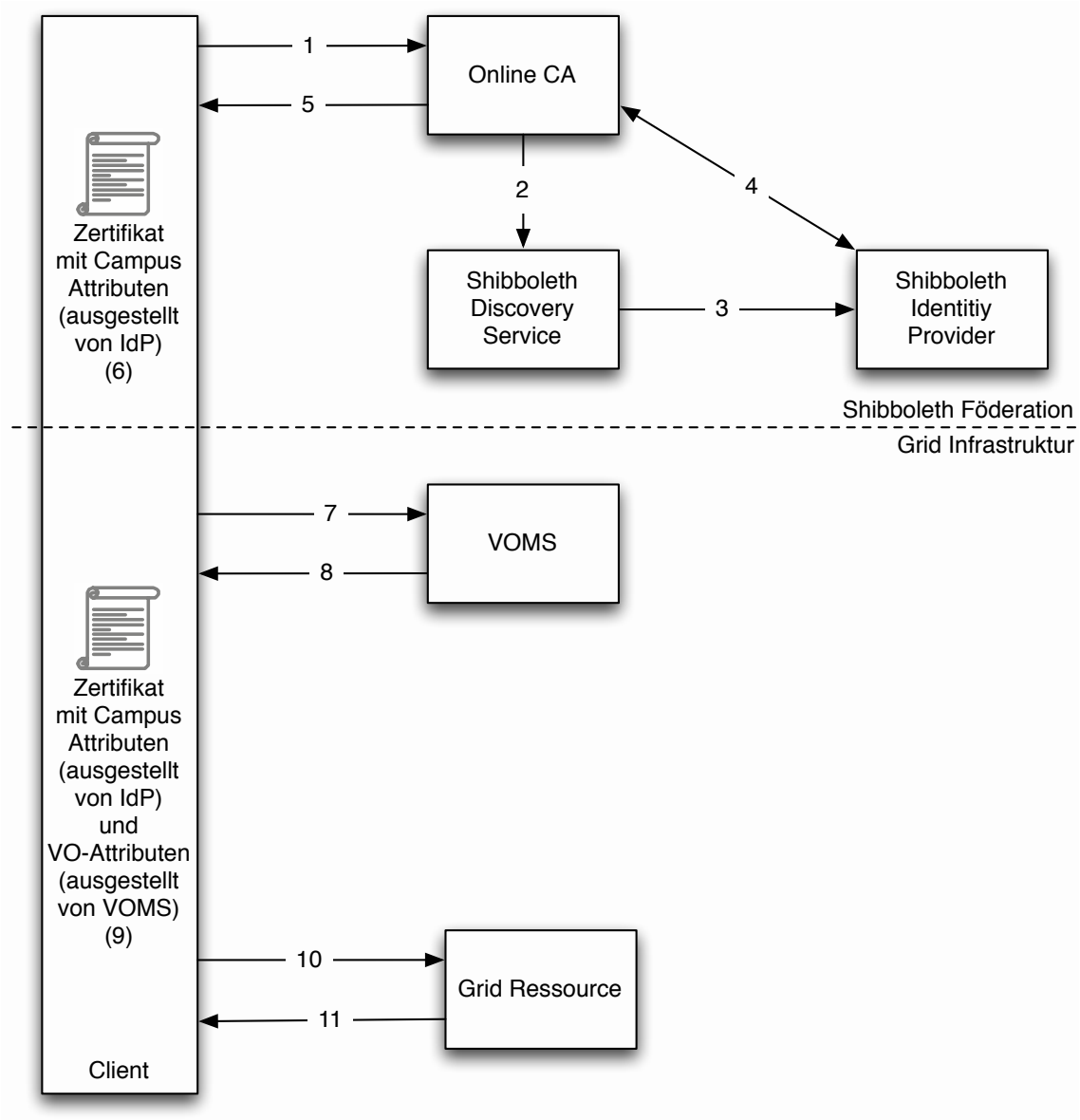

Abbildung 4.5: Workflow mit VOMS ohne Trust Proxying, Nutzer ohne langlebiges Zertifikat

Analyse

In diesem Ansatz wird kein Trust Proxying eingesetzt. Campusattribute werden vom zuständigen Identity Provider signiert, VO-Attribute vom zuständigen VO Management System und das kurzlebige Nutzerzertifikat von der akkreditierten Online CA. Dafür müssen auf den Grid-Ressourcen mehr Informationen verwaltet werden: Insbesondere die Zertifikate von allen Shibboleth Identity Providern und deren Attributscopes müssen bekannt sein, um die Campusattribute verifizieren zu können. Dies gilt gleichfalls für die Zertifikate der Online CA und des VOMS Services. Dies ist allerdings standardmäßig der Fall, da es sich hierbei um Dienste der Grid-Infrastruktur handelt.

Die Bindung der Campusattribute an die X.509 Identität des Nutzers ist durch die Online CA gegeben. Nutzer mit langlebigen Zertifikaten können nur unter den Voraussetzungen, die in Kapitel 3.4.3 diskutiert wurden, unterstützt werden. Bei den VO-Attributen erfolgt die Bindung durch den VOMS Service.

4.3 Design der attributbasierten Autorisierungsinfrastruktur mit mehreren Attribute Authorities

Die vorgestellten Ansätze verfügen alle über spezifischen Vor- und Nachteile. Eine Übersicht über die Eigenschaften aller vorgestellter Lösungen findet sich in Tabelle 4.1.

Hieraus ergibt sich, dass der Ansatz basierend auf VOMRS/VOMS für das VO Management und der GridShib CA für die Einbindung der Campusattribute in ein kurzlebiges Grid-Credential ohne Trust Proxying am besten geeignet ist. Es wird auf Trust Proxying verzichtet, bei Nutzung kurzlebiger Zertifikate ergibt sich kein Bindungs-Problem zwischen der X.509 Identität des Nutzers und der als SAML Attribute Assertion ausgestellten Campusattribute. Das Vorgehen für VO-Attribute verwendet den VOMRS/VOMS VO Management Service. Ebenso wird die existierende Online CA genutzt. Folgende Komponenten sind somit Bestandteil der Infrastruktur:

- VO-Management Frontend: VOMRS
- VO-Management Dienst: VOMS (mit oder ohne SAML-Unterstützung)

Ansatz	Trust Proxying	Bindungsproblem	Weiterverwendung D-Grid VO-Management	Weiterverwendung DFN Online CA
myVocs mit Trust Proxying und SLC	Ja	Ja (lösbar)	Nein	Ja
myVocs mit Trust Proxying und regulärem Zertifikat	Ja	Ja	Nein	Nein
myVocs ohne Trust Proxying und SLC	Nein	Ja (lösbar)	Nein	Ja
myVocs ohne Trust Proxying und regulärem Zertifikat	Nein	Ja	Nein	Nein
VOMRS/VOMS mit Trust Proxying durch VASH	Ja	Nein	Ja	(Ja, optional)
VOMRS/VOMS mit Trust Proxying durch Online CA	Ja	Nein	Ja	Ja
VOMRS/VOMS ohne Trust Proxying und SLC	Nein	Nein	Ja	Ja
VOMRS/VOMS ohne Trust Proxying und regulärem Zertifikat	Nein	Ja	Ja	Ja

Tabelle 4.1: Zusammenfassung der betrachteten Lösungen

- Online CA: GridShib Online CA
- Campus-Attributverwaltung: Shibboleth
- Attribute Certificate-Unterstützung für Globus Toolkit: VOMS-PDP
- Attribute Certificate-Unterstützung für gLite: eingebaut
- Attribute Certificate-Unterstützung für UNICORE5: im D-Grid Projekt „IVOM“ prototypisch entwickelt
- SAML-Unterstützung für Globus Toolkit: GridShib4GT
- SAML-Unterstützung für gLite: gJAF [12] oder ARGUS [41]
- SAML-Unterstützung für UNICORE5: im D-Grid Projekt „IVOM“ prototypisch entwickelt, UNICORE6: eingebaut

Hinzu kommt noch der Grid Resource Registration Service, der aus den Daten des VO Managements Policies für die bei ihm registrierten Grid-Ressourcen erstellt (siehe Kapitel 3.5.2).

Während die einzelnen Komponenten größtenteils bereits existieren und sich im produktiven Einsatz befinden, werden sie bisher nicht kombiniert eingesetzt. Die in dieser Arbeit dargelegten Forschungsergebnisse zeigen, wie sie im Verbund eine vertrauenswürdige Autorisierungsinfrastruktur ergeben, die durch die Verwendung von Attributen deutlich leistungsfähiger ist, als die identitätsbasierte Autorisierung, die momentan Stand der Technik ist.

Die Gesamtarchitektur der hier entwickelten Authentifizierungs- und Autorisierungsinfrastruktur ist in Abb. 4.6 als Komponentendiagramm dargestellt. Oben in der Abbildung ist zu erkennen, dass das VO-Management und die Campusföderation keine direkte Beziehung zueinander haben. Es ist keine organisatorische Verbindung zwischen diesen getrennten Entitäten notwendig. Der Client, der bisher beide Systeme getrennt genutzt hat (das VO-Management für den Zugriff auf das Grid, die Campusföderation beispielsweise für den Zugriff auf Universitätsbibliotheken), führt seine Nutzerinformationen aus beiden Quellen zusammen und ermöglicht so der Gridressource flexible Autorisierungsfunktionen. Durch die Modellierung von Vertrauensproblemen und der Analyse, ob die betrachteten Kombinationen der eingesetzten

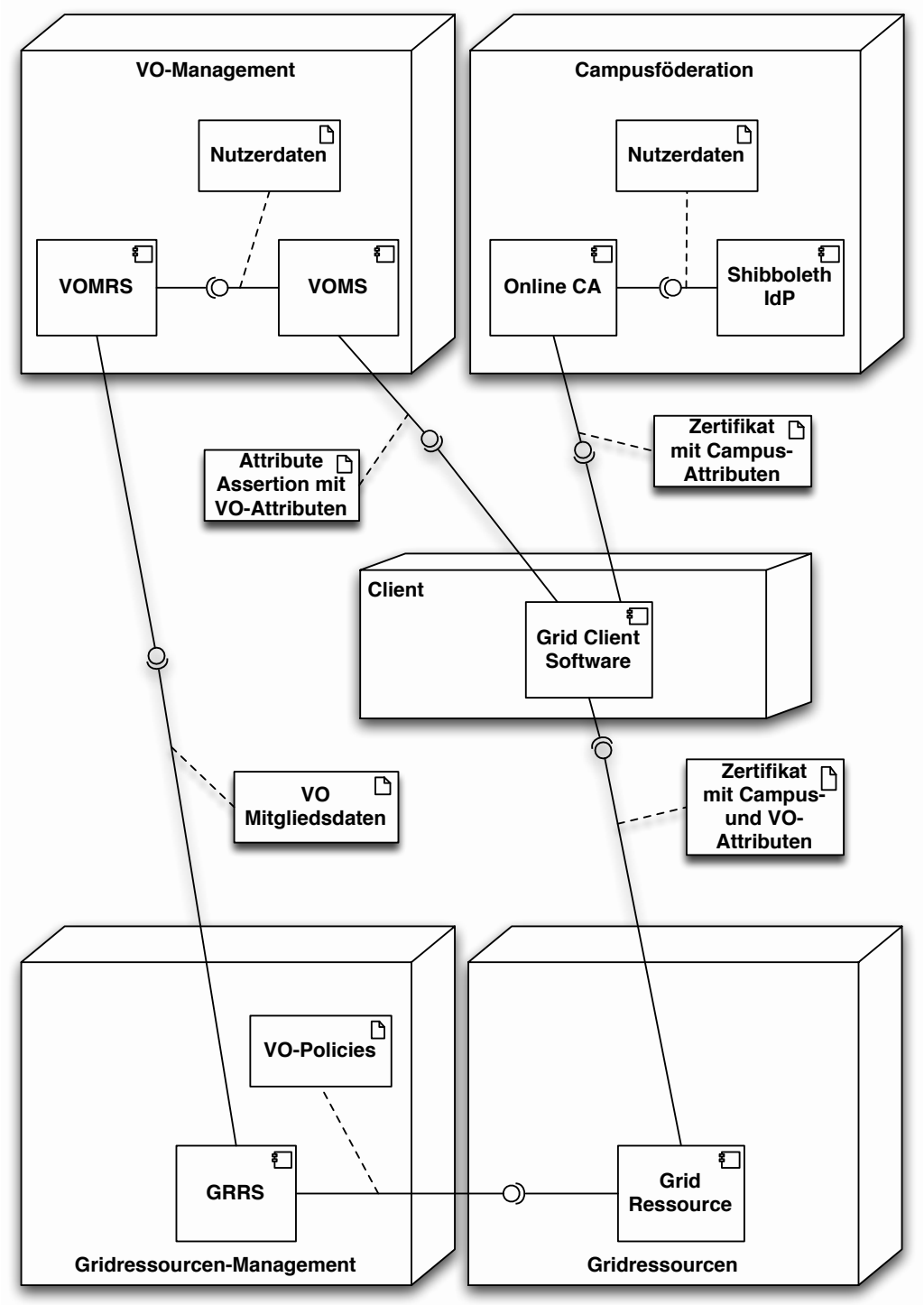

Abbildung 4.6: Gesamtübersicht als Komponentendiagramm

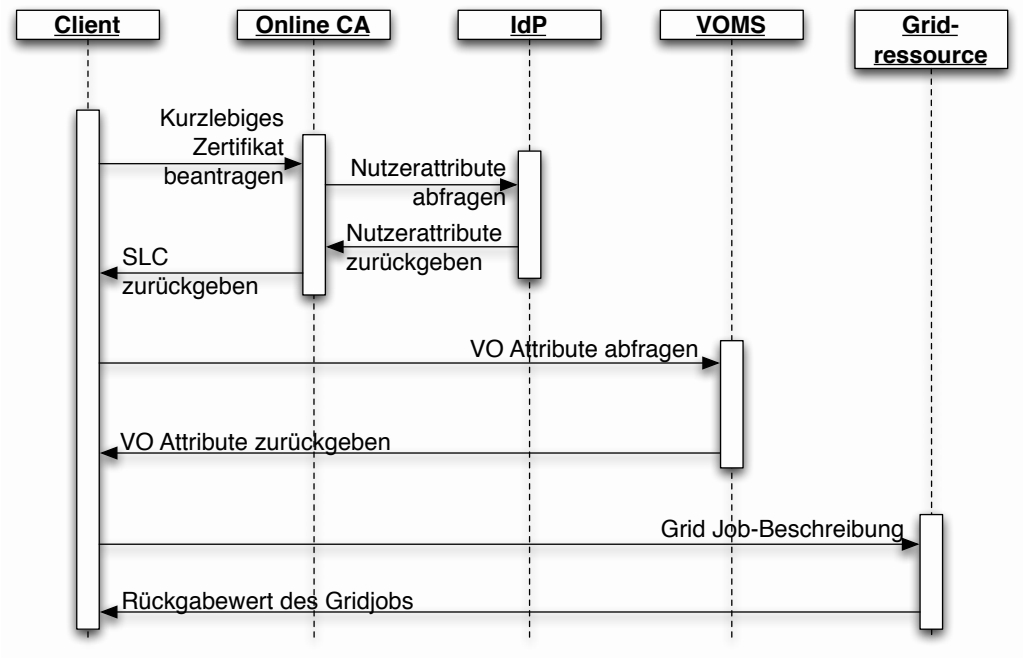

Abbildung 4.7: Gesamtübersicht als Sequenzdiagramm

Komponenten diese Probleme aufwerfen, ist sichergestellt, dass die hier diskutierte Lösung sicher und vertrauensvoll umsetzbar ist.

Der zeitliche Ablauf aus Nutzersicht wird als Sequenzdiagramm in Abb. 4.7 dargestellt und entspricht weitgehend der weniger formal modellierten Darstellung aus der Vorstellung des Ansatzes oben (Kapitel 4.2.5). Es ist zu erkennen, dass auf dem Clientsystem drei Interaktionen gestartet werden müssen, nämlich das Beantragen eines kurzlebigen Zertifikats, die Anforderung der VO-Attribute und die Abgabe des eigentlichen Gridjobs. Da nur der letzte Schritt aus Nutzersicht wichtig ist und die beiden ersten Schritte aus Sicht des Nutzers unproduktiven Overhead bedeuten, muss dies vor dem Nutzer so weit wie möglich verborgen werden. Durch die entsprechende Komponente der im Folgenden entworfenen Abstraktionsschicht wird es den Entwicklern der Clientsoftware vereinfacht, dies zu erreichen, ohne über Expertenwissen in der Gridsecurity zu verfügen.

Die Unterstützung für die verwendeten Kodierungen der Attribute durch die betrachteten Grid Middlewares bzw. deren Autorisierungskomponenten ist noch nicht vollständig vorhanden. In Kapitel 5.2.2 wird eine Roadmap diskutiert, mit dem die hier betrachtete middlewareübergreifende Infrastruktur in zwei Schritten um diese Funktionen erweitert werden kann.

4.4 Design der Abstraktionsschicht für domänenspezifische Softwareentwickler

Wie in Kapitel 3.6.2 dargestellt, ist eine der größten Quellen unsicherer oder nicht funktionierender Software im Grid die fehlerhafte Anwendung der Sicherheitskonzepte und -APIs durch die Entwickler domänenspezifischer Gridsoftware. Um diesen Entwicklern die Arbeit zu vereinfachen und somit häufige Fehlerquellen vor den Entwicklern zu verbergen, wird eine Abstraktion spezifiziert, welche die Komplexität der Grid-Funktionen so weit wie möglich abstrahiert.

Die konkreten Anwendungsfälle im Grid Computing sind sehr vielseitig und unterscheiden sich aus Clientsicht zum Teil stark von Community zu Community bzw. von Virtueller Organisation zu Virtueller Organisation. Die hier berücksichtigten Anwendungsfälle sind derart ausgewählt, dass sie die Basis für möglichst viele communityspezische Grid-Infrastrukturen darstellen können. Darüber hinaus gehende Anwendungsfälle werden innerhalb einer Virtuellen Organisation durch eigene Ressourcen und Implementierungen umgesetzt und sind nicht Bestandteil dieser Arbeit.

Die Gesamtarchitektur, in der dieses API eingesetzt wird, ist in Abb. 4.8 als Schichtenmodell dargestellt. Ausgehend vom Nutzer ist die oberste Schicht eine dedizierte Clientapplikation. Zwei Typen von Clients werden unterstützt: Gridfähige und nicht gridfähige Clients.

Eine gridfähige Clientapplikation kann auf drei Arten auf das Grid zugreifen: Direkt auf die generische Grid Middleware, also auf reine Compute- und Speicherdienste. In diesem Fall werden gar keine domänenspezifischen Dienste in Anspruch genommen. Außerdem kann die Clientapplikation auf domänenspezifische Grid-Dienste zugreifen, die wiederum im Backend die generischen Grid-Dienste verwenden, um z. B. komplexe Rechnungen durchzuführen. Drittens kann die Clientapplikation auf eine Workflowengine zugreifen, welche mehrere tieferliegende Dienste orchestriert. Die domänenspezifische Grid-Schicht umfasst somit mehrere Dienstetypen und liegt zwischen dem Client und der generischen Grid Middleware, wie sie von der D-Grid Infrastruktur bereitgestellt wird. Die domänenspezifischen Grid-Dienste greifen, je nach Art des Dienstes und der Anforderungen der Anfrage, wiederum auf generische Grid-Dienste über eine Grid Middleware zu. Diese abstrahieren die eigentlichen Hardwareressourcen wie Fileserver oder Computecluster.

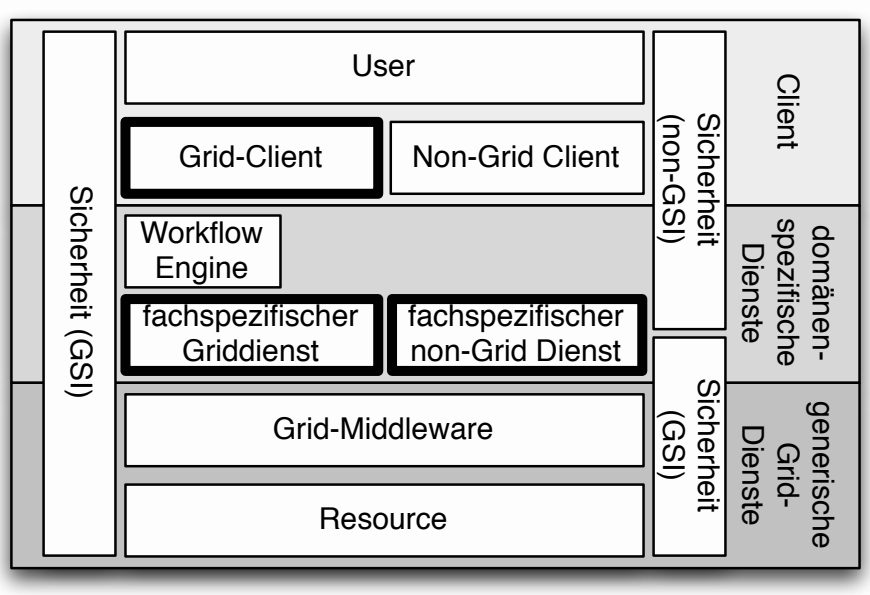

Abbildung 4.8: Schichtenmodell einer domänenspezifischen Grid-Infrastruktur. Fett umrandet sind die Komponenten, in denen domänenspezifischen Softwareentwicklern Gridfunktionen integrieren müssen

Non-Grid Clients greifen auf nicht gridkompatible Schnittstellen fachspezifischer Dienste zu. Dies sind beliebige Services, z. B. SOAP Web Services, ohne gridspezifische Funktionen. Dies betrifft insbesondere die Sicherheitsinfrastruktur. Da die Grid Security Infrastructure das lückenlose Delegieren von Credentials vom Client zur Ressource erfordert, muss eine Lösung gefunden werden, wie der fachspezifische Dienst ein Credential des Nutzers erhält, um auf die generischen Grid-Dienste zugreifen zu können.

An zwei Stellen müssen die Entwickler domänenspezifischer Software auf Gridfunktionen zugreifen: Zum einen im Client, der neben der Initiierung eines Gridjobs und dem Zugriff auf Grid-Dienste auch für das Handling der Credentials zuständig ist. Zum anderen in den domänenspezifischen Diensten, die auf die generischen Grid-Dienste zugreifen müssen. An diesen Stellen wird die Abstraktionsschicht eingesetzt, die in dieser Arbeit definiert wird.

4.4.1 Sicherheit

Da sowohl gridfähige als auch nicht gridfähige Clients unterstützt werden sollen, müssen zwei Lösungen gefunden werden, welche die Kette von Delegationen von

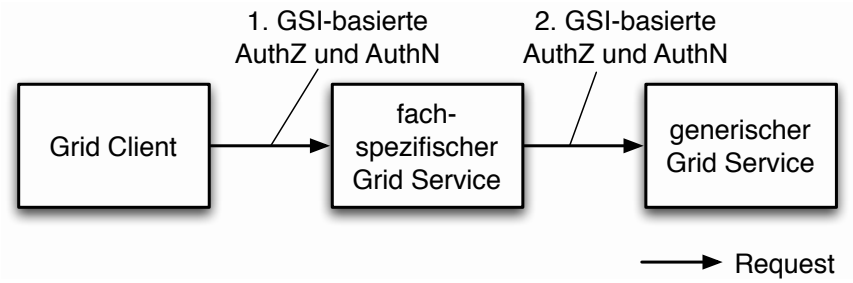

Abbildung 4.9: Sicherheit für Zugriffe mit Grid Client

Proxycredentials sicherstellen.

Gridfähige Clients und gridifizierte Dienste sind einfacher umzusetzen. In diesem Fall geschieht die Delegation des Nutzercredentials über die Grid Security Infrastructure direkt vom Clientprogramm aus zu den Grid-Diensten und zwischen den Grid-Diensten. Dies ist in Abb. 4.9 dargestellt: Der Grid Client greift unter Verwendung der GSI auf einen fachspezifischen Grid Service zu. Dieser verwendet das delegierte Credential, um im Namen des Nutzers beim generischen Grid Service authentifiziert und autorisiert werden zu können. Die Einbettung von Campus- und VO-Attributen in das Nutzercredential vor Initiierung des Gridjobs muss durch das API ermöglicht werden.

Die Anbindung von nicht Grid Security Infrastructure-kompatibler Software und entsprechenden Diensten erzeugt eine Lücke in den verwendeten Sicherheitskontexten. Die nicht GSI-kompatiblen fachspezifischen Dienste rufen in diesem Anwendungsfall Dienste auf GSI-kompatiblen Ressourcen auf. Diese Komponenten müssen somit ein GSI-kompatibles Grid Credential des eigentlichen Nutzers erhalten, um Grid-Dienste in dessen Namen aufrufen zu können. Dieses Problem wird gelöst, indem ein vom Nutzer vorher hinterlegtes Grid-Credential von einem MyProxy Service bezogen wird.

Der Nutzer delegiert regelmäßig seine Rechte über ein Proxycredential an den MyProxy Service, beispielsweise jeden Morgen. Hierzu verwendet der Nutzer entsprechende MyProxy Upload Tools, die zwar auf seinem Arbeitsplatzrechner installiert sein müssen, aber nicht mit der bereits vorhandenen Clientsoftware interagieren. Das delegierte Credential ist auf dem MyProxy Service über einen Benutzernamen und ein Passwort gesichert, so dass nicht jeder beliebige Client das Credential des Nutzers abfragen kann. Diese Benutzername/Passwort-Kombination muss dem Service

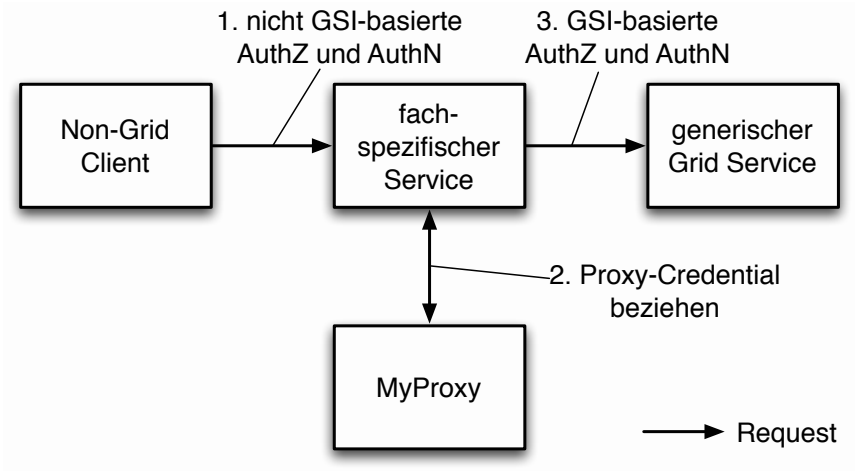

Abbildung 4.10: Sicherheit für Zugriffe ohne GSI

bekannt sein, um das Credential des Nutzers vom MyProxy Service beziehen zu können. Das prinzipielle Zusammenspiel der Komponenten ist in Abb. 4.10 dargestellt. Im ersten Schritt greift der Non-Grid Client unter Verwendung geeigneter, aber nicht grid-kompatibler Authentifizierungs- und Autorisierungsmechanismen auf den fachspezifischen Dienst zu und übermittelt hierbei den Benutzernamen und das Passwort für das auf dem MyProxy Dienst hinterlegte Credential. Dieses ruft der Dienst im zweiten Schritt ab und kontaktiert hiermit im dritten Schritt unter Verwendung der Grid Security Infrastructure den generischen Grid Service.

Damit diese Informationen nicht von Angreifern abgefangen werden können, ist es notwendig die Kommunikation zwischen Client und Web Service zu verschlüsseln. Dies kann über HTTPS statt HTTP geschehen (Transport Layer Security), über verschlüsselte SOAP-Nachrichten (Message Layer Security) oder über einen gesicherten Tunnel bzw. ein Virtuelles Privates Netzwerk (VPN).

Um die Funktion und Vertrauenswürdigkeit dieser Lösung sicherzustellen, müssen folgende Anforderungen erfüllt sein:

- Der Dienst, der das Proxycredential vom MyProxy Service abholt, muss die Identität des anfragenden Nutzers sicherstellen. In keinem Fall darf eine Anfrage mit einem anderen Proxycredential als dem des aktuellen Nutzers ins Grid geleitet werden.

- Eine gültige Delegation muss auf dem MyProxy Service verfügbar sein. Sie darf noch nicht abgelaufen sein und die Restlaufzeit muss größer oder gleich

der Laufzeit des Jobs sein.

Die erste Anforderung wird durch die Verwendung der vom Nutzer zum Schutz seiner Credentials vergebenen MyProxy Username/Passwort-Kombination sichergestellt. Die Verantwortung liegt beim Nutzer, diese vertraulichen Daten nicht weiterzugeben. Die zweite Anforderung kann entweder erfüllt werden, indem der Nutzer darauf achtet, immer eine gültige Delegation auf dem MyProxy Service zu hinterlegen, sobald die alte abläuft, oder durch das einmalige Hinterlegen einer sehr lange gültigen Delegation. Im Extremfall ist es technisch möglich, ein Proxycredential mit der selben Laufzeit wie der des ausstellenden Nutzerzertifikates an den MyProxy Service zu delegieren. Dies wird aber nicht empfohlen, da hier bei Missbrauch nur ein Rückruf des Nutzerzertifikates den Missbrauch stoppen kann. Da ein Missbrauch nicht automatisch erkannt werden kann, ist dies ein ungenügendes Vorgehen. Aktuelle Forschungen versuchen Methoden zu finden, die den Missbrauch von Proxycredentials über Heuristiken erkennen können [40]. Es ist aber weiterhin nicht empfehlenswert, eine derartig langlebige Delegation aus dem persönlichen Einflussbereich herauszugeben. Die Empfehlungen für kurzlebige Zertifikate (Siehe Kapitel 3.5.4) sollten somit auch hier beachtet werden, d. h. die Delegation an den MyProxy Service sollte nicht länger als 1 000 000 Sekunden ($\approx 11\frac{1}{2}$ Tage) gültig sein.

In Abb. 4.11 ist der zeitliche Ablauf des beschriebenen Ansatzes als Sequenzdiagramm dargestellt. Zuerst kontaktiert der nicht gridifizierte Client den Web Service. Authentifizierung und Autorisierung werden hier durch Mechanismen sichergestellt, die nicht mit der GSI kompatibel sind. Der Dienst kontaktiert nun den MyProxy Service und bezieht mittels der vom Nutzer in der Anfrage übermittelten Username/Passwort-Kombination das Proxycredential des Nutzers. Die Anfrage wird mittels des Proxycredentials an den eigentlichen Grid-Dienst gestellt. Hier wird nun mittels Grid Security Infrastructure authentifiziert und autorisiert sowie eine Delegation der Rechte an den Gridservice ausgeführt. Mittels dieser Delegation kann der Gridservice nun, je nach Notwendigkeit, im Namen des Nutzers weitere Grid-Ressourcen verwenden, um die Anfrage auszuführen.

Die Einbettung von Campus- und VO-Attributen muss in diesem Fall durch das MyProxy Upload Tool ermöglicht werden, damit bereits auf dem MyProxy Service ein mit Attributen versehenes Credential vorliegt. Die Funktionen des MyProxy Upload-Tools sind nicht Bestandteil dieser Arbeit, da dieses nicht von den domä-

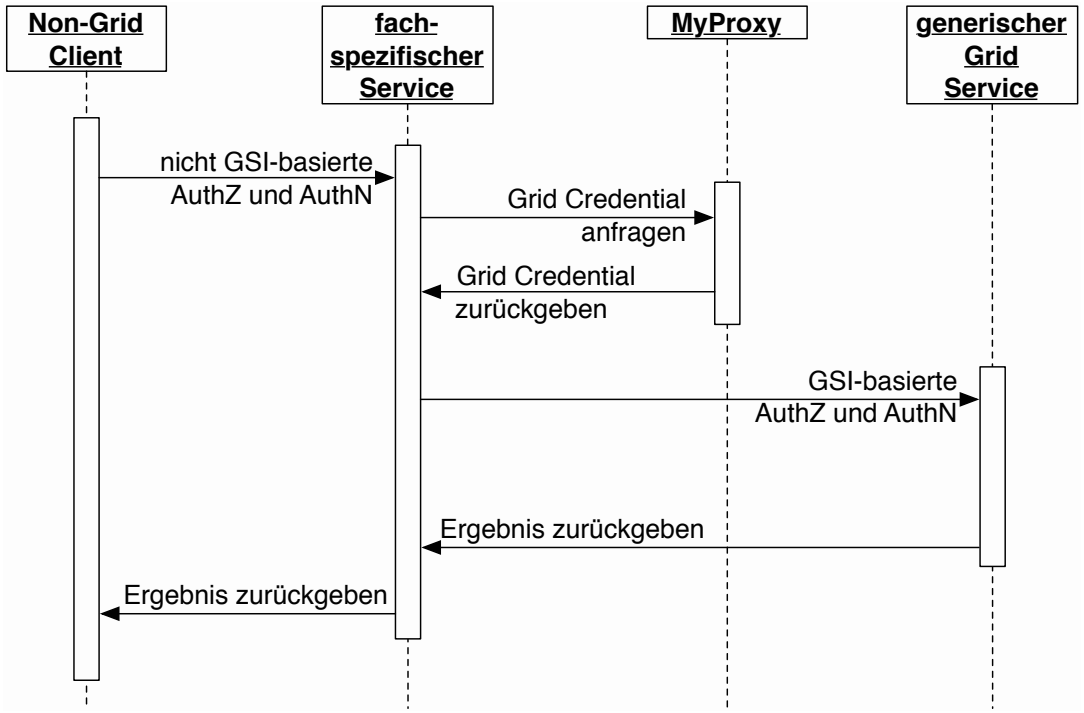

Abbildung 4.11: Sequenzdiagramm eines Zugriffs über einen nicht gridifizierten Web Service

nenspezifischen Entwicklern implementiert werden muss. Ein grafisches MyProxy Upload Tool wurde in [45] implementiert, welches allerdings noch nicht die Einbettung von Attributen ermöglicht und noch entsprechend erweitert werden muss.

4.4.2 Notwendige Funktionen von Grid Client Programmen

Gridfähige domänenspezifische Clientprogramme benötigen neben den eigentlichen Fachmethoden folgende Funktionen:

Erstellen eines Grid Credentials: Ein Grid Credential muss aus dem lokal vorhandenen Zertifikat abgeleitet werden. Hierzu muss der Nutzer die Passphrase eingeben, mit welcher der private Schlüssel geschützt ist. Wenn kein lokales Zertifikat vorhanden ist, muss eines von der Online CA abgerufen werden. Hierzu muss der Nutzer seine Heimateinrichtung angeben und seine Logindaten eingeben. Von einer Online CA bezogene Credentials können Campusattribute enthalten. Eine explizite Unterstützung durch die Clientsoftware ist nicht notwendig.

Einbetten von VO-Attributen: In das Proxyzertifikat des Grid-Credentials müssen VO-Attribute eingebunden werden, damit diese auf den Grid-Ressourcen für die Autorisierung verwendet werden können. Hierzu wird der VO-Management Service der gewünschten Virtuellen Organisation abgefragt. Wenn der Nutzer Mitglied mehrerer Virtuellen Organisationen ist, muss er auswählen, welche VO oder Kombination von VOs er verwenden möchte.

Delegieren des Credentials an domänenspezifischen Grid-Dienst: Wenn die Fachmethode eines domänenspezifischen Dienstes angesprochen wird, muss das Credential an diesen Dienst delegiert werden, damit im weiteren Verlauf des Jobs generische Dienste der Grid-Infrastruktur angefragt werden können. Die Verwendung der eigentlichen Fachmethode des Dienstes ist nicht Bestandteil dieser Abstraktionsschicht.

Zugriff auf Grid Middleware: Wenn direkt aus der Clientsoftware Rechen- oder Speicherressourcen angesprochen werden sollen, muss diese auf generische Grid-Dienste zugreifen können. Dies entspricht dem Vorgehen eines domänenspezifischen Dienstes zur Einbindung von Gridfunktionen (siehe nächster Abschnitt).

4.4.3 Notwendige Funktionen domänenspezifischer Dienste

Domänenspezifische Dienste benötigen neben den eigentlichen Fachmethoden folgende Funktionen:

Empfang der Delegation eines Credentials vom Client: Sind Dienst und Client gridkompatibel, wird das Credential direkt vom Client an den Dienst delegiert. Eingebundene Attribute können vom Dienst zur Autorisierung verwendet werden.

Beziehen von Credentials von MyProxy: Wenn der Dienst von einem nicht gridfähigen Client angesprochen wird, muss er ein Nutzercredential von einem MyProxy Service beziehen. Hierfür muss der Nutzer seinen Benutzernamen und sein Passwort angeben. Dieses Credential kann nun genutzt werden, um Grid-Middlewaredienste ansprechen zu können. Attribute müssen hier nicht mehr eingebunden werden, da das Credential mit allen notwendigen Attributen auf den MyProxy hochgeladen wird. Eventuell vorhandene Attribute können vom Dienst zur Autorisierung verwendet werden.

Initiierung eines Grid-Jobs über generische Grid-Dienste: Um rechen- oder speicherintensive Aufgaben erledigen zu können, spricht der domänenspezifische Grid-Dienst eine Grid-Middlewarekomponente an. Hierzu verwendet er entweder das direkt delegierte oder das von MyProxy bezogene Credential.

Die erste Funktion wird, wie in Kapitel 4.4.1 diskutiert, nur von Diensten mit nicht gridfähigen Interfaces für herkömmliche Clients benötigt. Die zweite Funktion wird von allen Diensten benötigt, die generische Grid Computeressourcen kontaktieren, um ihre Funktion zu erfüllen.

Die Auflistung der hier diskutierten Eigenschaften eines „myGridAPI“, welches diese Abstraktionsschicht implementiert, findet sich in Anhang A als Lastenheft bzw. Anhang B in Form eines Pflichtenheftes nach [4] formal dokumentiert.

Kapitel 5

Implementierung der vertrauenswürdigen attributbasierten Autorisierung im Grid Computing

5.1 Übersicht

Die Implementierung der bisher geschilderten Konzepte erweitert existierende Software, die für die Umsetzung angepasst oder neuartig kombiniert werden muss. Die Wiederverwendung existierender Software stellt zum einen die Kompatibilität mit anderen Grid-Infrastrukturen sicher und zum anderen sorgt sie dafür, dass die sicherheitsrelevanten Funktionen durch gut getestete und gewarteten Komponenten bereitgestellt werden [32].

Dieses Kapitel ist in zwei Abschnitte unterteilt: Zuerst werden die Voraussetzungen und Schritte diskutiert, die notwendig sind, um eine rein identitätsbasierte Grid-Infrastruktur um attributbasierte Autorisierung zu erweitern. Dies wird am Beispiel der D-Grid Kerninfrastruktur dargelegt. Der zweite große Abschnitt beschäftigt sich mit den notwendigen Voraussetzungen für domänenspezifische Grid-Softwareentwickler, die sichere Software für diese Infrastruktur entwickeln. Hierbei werden für Clientsoftware und Gridservices APIs definiert, die es diesen Entwicklern erlauben, ohne tieferes Verständnis der Grid-Infrastrukturen und ihrer Sicherheitsinfrastrukturen sichere Software zu entwickeln.

5.2 Implementierung der attributbasierten Autorisierung

5.2.1 VO-Managementsystem

Das zentrale D-Grid Nutzermanagement bestand aus einem VOMRS Service, der allerdings ausschließlich eine komfortable webbasierte Nutzerverwaltung ermöglicht. Die Nutzerdaten werden in einer Datenbank gehalten. VOMRS bietet aber neben dem Webinterface keinerlei Schnittstellen an, um diese Daten weiter zu verwenden. Das VOMRS System kann mit einem VOMS System gekoppelt werden (vgl. Kapitel 3.5.2), so dass das VOMRS System als führendes System die Verwaltungsfunktionen bietet, während das VOMS System seine Schnittstellen bereitstellt, um mit den Nutzerdaten weitere Funktionen zu ermögliche, wie beispielsweise die Autorisierung auf Grid-Ressourcen anhand von im VO-Management verwalteten Attributen. VOMS bietet hier aber auch nur Schnittstellen zur vertrauenswürdigen Zusicherung von Nutzerattributen, d. h. Nutzer können sich sie beschreibende Attribute ausstellen

lassen.

5.2.2 Überführung in den Regelbetrieb

Übersicht

Das als Ergebnis dieser Arbeit ermittelte Konzept ermöglicht die vertrauenswürdige Übermittlung von Attributen an Gridkomponenten. In Kapitel 3.5 wurde gezeigt, dass keine der betrachteten Grid-Middlewares die Autorisierung mit Unterstützung für Campus- und VO-Attribute momentan unterstützt. In den Entwicklungsteams aller Middlewares gibt es Bestrebungen, SAML-Unterstützung nachzurüsten bzw. die Autorisierungsframeworks derartig zu generalisieren, dass sich beliebige SAML-basierte Anwendungsszenarien durch reine Konfiguration oder Programmierung von Plugins gegen feste Schnittstellen umsetzen lassen. Die Ergebnisse dieser Arbeit sind kompatibel zu diesen geplanten Erweiterungen und schaffen somit die Voraussetzungen, um die erwarteten Funktionen sinnvoll einsetzen zu können.

Die Verfügbarkeit in stabilen Versionen der Middlewares ist allerdings zeitlich nicht vorherzusagen. Daher muss für die Umsetzung der beschriebenen Architektur ein mehrschrittiges Vorgehen gewählt werden. Dieses besteht aus

- Zwischenlösungen, welche nach relativ kurzer Zeit wieder durch neuere Lösungen ersetzt oder um neue Funktionen erweitert werden. Dies ist notwendig, um sofort erste attributbasierte Autorisierungslösungen anbieten zu können und
- eine Kombination von Schemata, Standards und Softwarekomponenten auszuwählen, die eine solide Basis für die zukünftige fertige D-Grid Autorisierungsinfrastruktur legen.

Die Zwischenlösung muss implementiert werden, auch wenn sich nicht vollständig konform mit der geplanten Zielarchitektur ist. Wie in Kapitel 3.5 dargestellt wurde, gibt es keine Lösung, die auf aktuell verfügbaren Komponenten basiert, Campusattribute und VO-Attribute für die Autorisierung auf den im D-Grid eingesetzten Middlewares unterstützt und keine Trustprobleme durch den Einsatz von Trust Proxying oder dem Bereitstellen möglicherweise nicht mehr gültiger Attribute erzeugt.

Es ist allerdings bereits jetzt möglich, VO-Attribute in allen drei Middlewares zu unterstützen, wobei im Globus Toolkit nur die Web Service basierten Komponenten unterstützt werden. Basierend auf den Anforderungen der Communities im D-Grid,

welche in [29] erhoben wurden, kann die Unterstützung von VO-Attributen als kurzfristig wichtiger angesehen werden als die Unterstützung von Campusattributen. Daher kann diese Unterstützung eingeführt werden, bevor auch alle Komponenten die Autorisierung anhand von Campusattributen unterstützen.

Es muss allerdings sichergestellt werden, dass diese VOMS-basierte Erweiterung in soweit zukunftssicher ist, dass die Unterstützung von Campusattributen eingeführt werden kann, wenn die Unterstützung von SAML Attribute Assertions in den Grid Middlewares verfügbar wird. Die folgende Roadmap ist als Leitfaden konzipiert, der es existierenden D-Grid Communities erlaubt, attributbasierte Autorisierung schrittweise auf ihren Ressourcen einzuführen, ohne auf die Verfügbarkeit aller notwendiger Komponenten für die Zielarchitektur warten zu müssen. Weiterhin stellt der Leitfaden sicher, dass die Ressourcen der Communities untereinander kompatibel bleiben, da eine gemeinsame Technologie zugrunde liegt. Dies ist bei parallelen Eigenentwicklungen in den Communities nicht gewährleistet.

Aktuelle Situation: Identitätsbasierte Autorisierung im D-Grid

Abbildung 5.1 stellt den aktuell eingesetzten Ansatz für die Authentifizierung und Autorisierung auf den Komponenten der D-Grid Infrastruktur dar.

Die Autorisierung basiert nicht auf Attributen, die den Nutzer beschreiben, sondern rein auf seiner Identität. Diese wird durch den X.509 Distinguished Name aus dem Nutzerzertifikat repräsentiert. Die Grid-Ressourcen bilden anhand der Autorisierungspolicies diesen X.509 DN auf einen lokalen UNIX Useraccount ab. Wenn eine Policy für ein solches Mapping vorliegt, ist der Nutzer autorisiert, andernfalls nicht.

Die Policies werden aus den Daten der VO Managementsysteme gebildet. Jede Ressource unterstützt mindestens eine D-Grid VO. Aus den Daten der VO Managementsysteme werden die Listen generiert, welche die X.509 Distinguished Names aller VO-Mitglieder enthalten. Diesen wird jeweils ein lokaler UNIX Account auf jeder Ressource zugewiesen. Der Datenabgleich zwischen Ressource und VO Managementsystemen findet regelmäßig, i. d. R. einmal am Tag, statt. Das VO Managementsystem beinhaltet bereits einen VOMS Service, dieser wird allerdings nicht verwendet, um Attribute Certificates auszustellen.

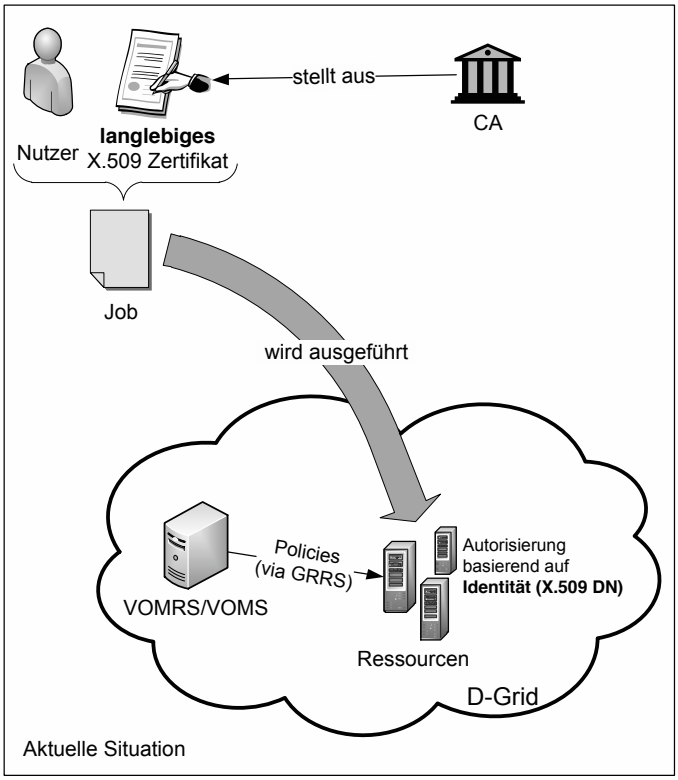

Abbildung 5.1: Aktuelle Situation

Schritt 1: Einsatz von VO-Attributen zur Autorisierung

Der erste Schritt für eine attributbasierte Autorisierung auf den D-Grid Ressourcen ist in Abbildung 5.2 dargestellt. Dieser Schritt fügt die attributbasierte Autorisierung anhand von VO-Attributen ein. Hierzu werden die VO-Managementsysteme VOMRS/VOMS verwendet. Der VOMS Service wird verwendet, um Attribute Certificates und SAML-Assertions mit den Nutzerattributen auszustellen. Welche Kodierung zu verwenden ist, hängt von der Zielmiddleware ab. Es ist auch möglich, beide Kodierungen gleichzeitig zu verwenden, um Middlewareübergreifende Jobs zu ermöglichen. Da die Attribute Assertions den X.509 Distinguished Name des Nutzers enthalten, ist die Bindung an das Nutzerzertifikat explizit gegeben und kann von den Grid-Ressourcen überprüft werden.

Die Policies mit den X.509 Distinguished Names autorisierter Nutzer muss weiterhin regelmäßig vom GRRS an die Grid-Ressource übertragen werden. Dieser Ansatz erlaubt es aber, dass ein Nutzer auf mehrere lokale UNIX Accounts gemappt werden kann, wovon beispielsweise einer höhere Rechte im Dateisystem besitzt. So kann der Nutzer, wenn er sich über ein Attribut als VO Software Administrator ausweist, auf

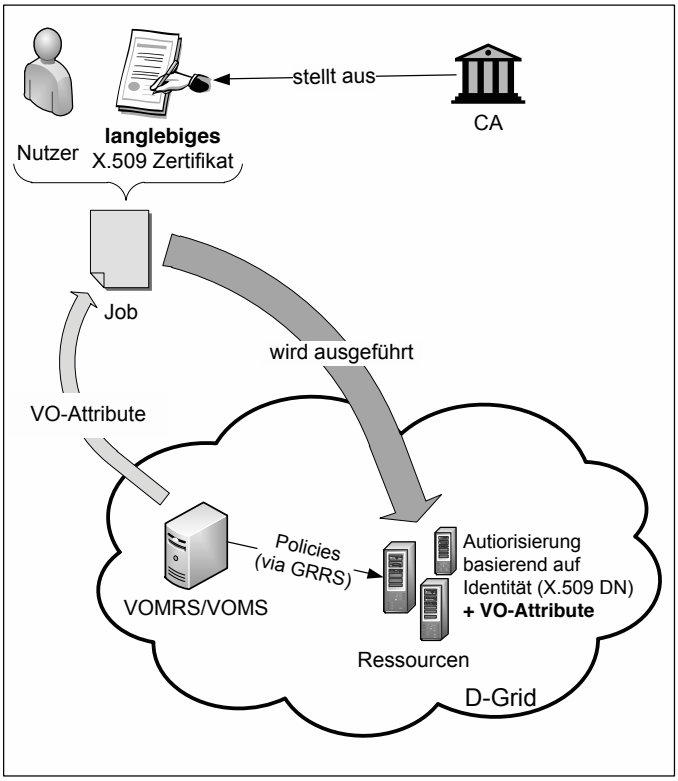

Abbildung 5.2: Schritt 1

diesen Account gemappt werden. Wenn er mit einem Attribut, das ihn als normalen Nutzer ausweist, autorisiert wird, wird er auf den anderen Account gemappt, der weniger Rechte hat. Somit werden bereits weitergehende Usecases unterstützt, aber das Skalierungsproblem mit der wachsenden Größe der Grid-Infrastruktur noch nicht behoben.

Die im Vergleich zur aktuellen Situation notwendigen Komponenten sind:

- SAML-fähiger VOMRS/VOMS Service: Dieser Service wird bereits im D-Grid betrieben, aber nicht genutzt.
- gLite: gLite unterstützt VOMS Attribute Certificates. Die Konfiguration muss entsprechend angepasst werden.
- Globus Toolkit: Der VOMS PDP für die Globus Web Service Komponenten muss auf den Grid-Ressourcen installiert und konfiguriert werden. Die prä-Web Service Komponenten werden nicht unterstützt.
- UNICORE5: Der Attribute Certificate PDP/PIP aus dem IVOM Projekt muss

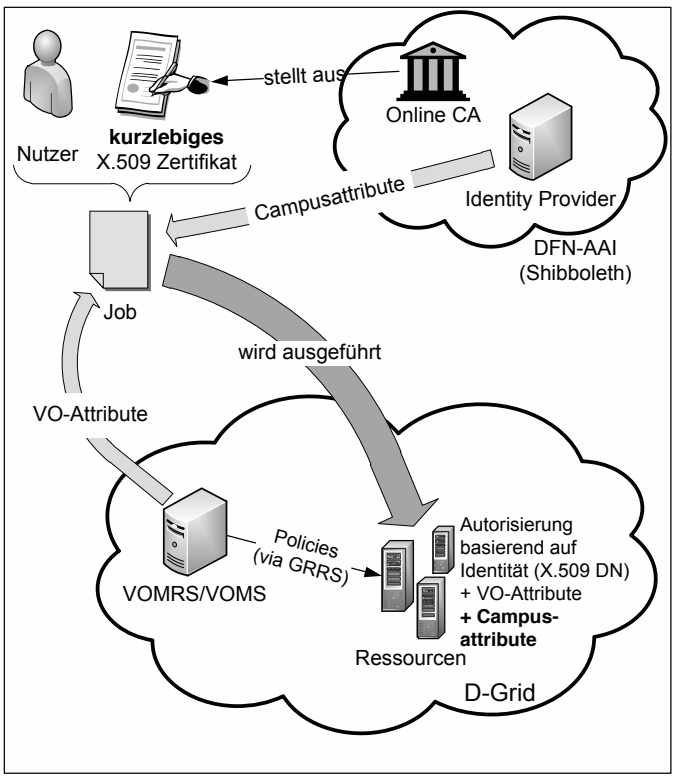

Abbildung 5.3: Schritt 2

installiert und konfiguriert werden. UNICORE6 wird durch den SAML-fähigen VOMS unterstützt und setzt bereits in diesem Schritt auf SAML Attribute Assertions.

Schritt 2: Einsatz von Campusattributen zur Autorisierung

Um neben den VO-Attributen auch Campusattribute zu unterstützen, müssen die Grid Middlewares SAML Attribute Assertions unterstützen. Der Transport von SAML Attribute Assertions zu den Grid-Ressourcen ist durch den Einsatz von Online CAs in den entsprechenden Campusföderationen möglich. Auf den Grid-Ressourcen müssen entsprechende PDPs bzw. PIPs eingesetzt werden. Der Ablauf eines Gridjobs mit Autorisierung basierend auf Campus- und VO-Attributen ist in Abbildung 5.3 dargestellt.

Wenn alle eingesetzten Grid Middlewarekomponenten SAML Attribute Assertions unterstützen, kann auch die Verwendung von VO-Attributen auf SAML umgestellt werden. Aktuelle Versionen des VOMS Service unterstützen die Ausgabe der

VO-Attribute als SAML Assertion. Auch diese enthält, wie die Attribute Certificates, den X.509 DN des VO-Mitgliedes. Das Binding an die Identität des Nutzers ist also weiterhin gewährleistet. Es ist sinnvoll, die gesamte Autorisierung auf SAML Assertions umzustellen, da die VOMS PDPs/PIPs für UNICORE und das Globus Toolkit nicht aktiv weiterentwickelt und unterstützt werden. SAML ist dagegen ein anerkannter Standard geworden, der von allen Middlewareentwicklern aktiv unterstützt wird. Somit verringert sich die Komplexität der Installation, da keine Attribute Certificate spezifischen Erweiterungen der Middlewares mehr notwendig sind und auch die Konfiguration einheitlicher ist.

Um diesen Schritt umsetzen zu können, sind folgende Komponenten notwendig:

- DFN-AAI: Alle alle Heimatorganisationen müssen beteiligt sein, deren Mitarbeiter in D-Grid VOs organisiert sind.
- Online CA des DFN-Vereins: Sie muss die signierten SAML Attribute Assertions von den Identity Providern der Heimatorganisationen der Nutzer empfangen und in die ausgestellten Zertifikate einbetten.
- Identity Provider der DFN-AAI: Sie müssen alle auf den Grid-Ressourcen als vertrauenswürdige Instanzen bekannt sein. Dies umfasst insbesondere ihre Zertifikate, die benötigt werden, um die SAML Attribute Assertions validieren zu können.
- gLite: Die Unterstützung für SAML-Assertions muss verfügbar sein.
- Globus Toolkit: GridShib for Globus muss installiert und konfiguriert werden. Die prä-Web Service Komponenten werden nicht unterstützt.
- UNICORE5: SAML-PDP/PIP aus IVOM muss installiert und konfiguriert werden. UNICORE6 bietet eine eingebaute SAML-Unterstützung

Dieser Ansatz skaliert für große Nutzermengen. Die Heimatorganisationen der Nutzer müssen keine gridspezifische Registration Authorities der Grid CAs mehr betreiben, um ihre Nutzer zu identifizieren. Es genügt, Mitglied der DFN-AAI zu sein. Die DFN-AAI ist nicht gridspezifisch, sondern kann für viele Anwendungsfälle genutzt werden. Der Nutzen für eine Organisation, in der DFN-AAI organisiert zu sein, ist somit deutlich höher, als der Betrieb einer reinen Grid Registration Authority.

5.3 Implementierung des myGridAPI

5.3.1 Existierende high-level Grid-APIs

Es existieren mehrere high-level APIs für die Implementierung gridfähiger Software. Dies sind zum einen diejenigen APIs, die von den Grid Middlewares zur Verfügung gestellt werden. Diese sind middlewarespezifisch und damit entwickelte Software kann nur im Kontext der jeweiligen Grid Middleware betrieben werden. Da im nach den Anforderungen dieser Arbeit drei Grid Middlewares unterstützt werden müssen, ist es für Entwickler von Diensten für das D-Grid nicht sinnvoll, diese APIs einzusetzen, da sie sich in alle APIs getrennt einarbeiten müssten.

Darüber hinaus gibt es zwei Ansätze für middlewareunabhängige Grid APIs.

- Das *Java Commodity Grid Kit* (CoG Kit) [43] ist ein high-level Grid API, das aus dem Kontext des Globus Toolkit hervorgegangen ist. Es beinhaltet das jGlobus Java API, das den Zugriff auf Globus Toolkit-basierte Grid-Ressourcen ermöglicht. Weitere Adaptoren existieren, allerdings bisher nicht für gLite und UNICORE.

- Das *Grid Application Toolkit* (GAT) [1] ist eine programmiersprachenunabhängige Definition eines generischen Grid API. Implementierungen dieser Definition sind in C, Python und Java verfügbar. Neben Adaptoren für das Globus Toolkit, gLite und UNICORE sind auch nicht gridspezifische enthalten, z. B. für SSH und PBS. GAT ist inzwischen Bestandteil der *Simple API for Grid Applications, SAGA* [37].

Diese APIs sind für den direkten Einsatz von Entwicklern, die keine umfassende Erfahrung mit der Programmierung von Gridkomponenten und dem genauen inneren Aufbau der von ihnen genutzten Grid-Infrastruktur haben, nicht geeignet, da sie diesbezüglich eine zu hohe Einstiegshürde mitbringen. In dieser Arbeit werden diese APIs eingesetzt, um ein minimales myGridAPI zu entwickeln, das die Gridfunktionen stark abstrahiert und notwendige Funktionen, wie den Einsatz der Sicherheitsinfrastruktur, so weit wie möglich verbirgt. Außerdem werden in dem API alle Eigenschaften, die von der konkreten Ausprägung der Grid-Infrastruktur abhängen, mit Defaultwerten versehen. Die Entwickler benötigen somit keine genauen Kenntnisse über den inneren Aufbau der Grid-Infrastruktur.

5.3.2 Credential-Management

In einem Grid Workflow müssen alle Komponenten, von der Clientsoftware bis zur letzten Datensenke für die Ergebnisse, die Authentifizierungs- und Autorisierungsinfrastruktur unterstützten. Selbst wenn eine Ressource frei zugänglich ist und keine Autorisierung durchführt, muss sie in der Lage sein, von der anfragenden Ressource eine Delegation eines Grid-Credentials zu empfangen und diese Delegation an nachgelagerte Gridkomponenten weiter zu delegieren. Um dies für die Entwickler domänenspezifischer Software zu erleichtern, enthält das myGridAPI hierfür entsprechende Funktionen in dem myCredentialAPI.

Ein Einsatzbereich des myCredentialAPI ist Clientsoftware, in der Credentials auch zwischen Jobs verwaltet werden müssen und in Szenarien mit attributbasierter Autorisierung entsprechende Attribute Authorities kontaktiert werden müssen. Diese Clients stellen den Startpunkt eines Gridjobs und somit auch der Delegationen von Grid-Credentials mit eingebetteten Attributen dar.

Ein weiterer Einsatzbereich des myCredentialAPI ist die Entwicklung von domänenspezifischen Grid-Diensten. Hiervon gibt es zwei Untertypen, von denen einer die Delegation der Grid-Credentials direkt vom anfragenden System empfängt und der andere, der von nicht gridfähigen Clients verwendet wird und somit kein delegiertes Credential vom anfragenden Client erhält. Dieser Typ bezieht das notwendige Credential für die Authentifizierung und Autorisierung im Grid über einen MyProxy Dienst (siehe Kapitel 2.2.2).

Credential Management für Clientsoftware

Das Credentialmanagement auf Clientsystemen verteilt sich auf drei Hauptfunktionen: Erstens das Management von X.509 Zertifikaten und den zugehörigen privaten Schlüsseln. Zweitens das Ableiten und Delegieren von Proxycredentials aus diesen Zertifikaten und Schlüsseln und drittens das Abrufen von Attribute Assertions von Attribute Authorities und das Einbinden dieser Attribute in die zu delegierenden Proxycredentials.

Hierfür sind die in folgendem Interface definierten Methoden notwendig:

void addCredentialFromFile(File file): Diese Methode fügt dem Credentialstore ein neues Credential hinzu, welches als PKCS12 Datei im Dateisystem zugreifbar sein muss.

void acquireCredentialFromOnlineCA(URL onlineCA): Diese Methode bezieht ein kurzlebiges Gridcredenital von einer Online CA. Dieses kann optional Campusattribute als eingebettete SAML Attribute Assertion enthalten.

void acquireVOAttributes(URL vomsService): Diese Methode greift auf einen VOMS Service zu und bezieht ein Attribute Certificate oder eine SAML-Assertion mit den VO-Attributen des Nutzers. Diese wird in das zu delegierende Proxycredential eingebaut.

void delegateToServce(URL gridService): Diese Methode erstellt aus einem Credential aus dem Credentialstore ein Proxycredential und delegiert dieses an den Delegation Service eines Grid-Dienstes. Dieser liefert eine Ressourcen-ID zurück, die anschließend mit dem eigentlichen Zugriff auf die Fachmethode des Dienstes verwendet werden kann.

Der Zugriff auf die eigentliche Fachmethode eines domänenspezifischen Grid-Dienstes kann in dieser API nicht abstrahiert werden, da hierzu unterschiedliche Funktionen vorhanden sein können. Diese Methoden müssen beispielsweise aus den WSDL-Beschreibungen der Dienste generiert werden. Die sicherheitsrelevanten Funktionen werden allerdings von der hier vorgestellten API abgedeckt. Der eigentliche Zugriff auf den Grid-Dienst findet dann üblicherweise über einen mit TLS gesicherten Kanal (Transport Level Security) oder über verschlüsselte XML-Nachrichten (Message Layer Security) statt. Diese Funktionen entsprechen denen regulärer Web Services und werden von den entsprechenden Frameworks bereits hinreichend abstrahiert.

Credential Management für Dienste

Grid-Dienste müssen Delegationen von Proxycredentials erhalten. Dies kann durch eine Delegation direkt vom aufrufenden System, bspw. einer Clientsoftware, geschehen, wenn Client und Dienst dies unterstützen. Wenn dies nicht gegeben ist, muss das Credential von einem MyProxy Service abgerufen werden, auf dem der Nutzer es vorher hinterlegt hat. Das Beziehen eines Nutzercredentials über eine Online CA ist hier nicht möglich, da hierfür die Interaktion des Nutzers notwendig wäre. Für das Abrufen eines Credentials von einem MyProxy System ist nur eine Methode notwendig:

GSSCredential acquireCredentialFromMyProxy(String user, String pass): Diese Methode bezieht ein Credential, das mit dem angegebenen Benutzernamen und Passwort auf dem voreingestellten MyProxy Credential Store hinterlegt wurde. Es wird in Java als Objekt des Typs GSSCredential zurückgegeben und kann beispielsweise den Properties eines Compute Jobs hinzugefügt werden (siehe unten). Ein solches Credential kann eingebettete Campus- und/oder VO-Attribute enthalten. Dies spielt an dieser Stelle keine Rolle, da das Credential bereits mit oder ohne diese Erweiterungen auf dem MyProxy Service hinterlegt wurde.

5.3.3 Zugriff auf Grid Computedienste

Zur sicheren Verwendung von Computediensten dient der myJobSubmissionAPI-Teil des myGridAPI. Computedienste sind im wissenschaftlichen Hochleistungsrechnen die wichtigsten generischen Grid-Dienste. Sie stellen allerdings keine domänenspezifischen spezialisierten Dienste mit genau definierten fachspezifischen Funktionen zur Verfügung, sondern bieten ein Interface für den Zugriff auf Hochleistungscluster über Gridtechnologien. Die eigentliche Logik muss auf der Ressource vorinstalliert sein oder als Executable mit dem Job und den Eingabedaten gemeinsam auf die Hochleistungsrechenressource kopiert werden. Die notwendigen Methoden dienen im Einzelnen folgenden Zwecken:

void cancelJob(): Mit dieser Methode kann ein laufender Job abgebrochen werden, falls seine Ergebnisse nicht mehr benötigt werden oder ein Fehler erkannt wurde.

JobProperties getProps(): Diese Methode liefert das Properties-Objekt zurück, in welchem die Optionen des Jobs hinterlegt sind.

JobStatus getStatus(): Diese Methode liefert den aktuellen Status des Jobs zurück. Da verschiedene Gridmiddlewares unterschiedliche Stati verwenden, werden diese auf einen gemeinsamen Satz an möglichen Stati abgebildet. Diese sind: FAILED, FINISHED, RUNNING, STAGEIN, STAGEOUT, UNSUBMITTED. Die möglichen Übergänge sind in Abb. 5.4 dargestellt. Zusätzlich ist es möglich, als Obeserver automatisch über Statusänderungen informiert zu werden.

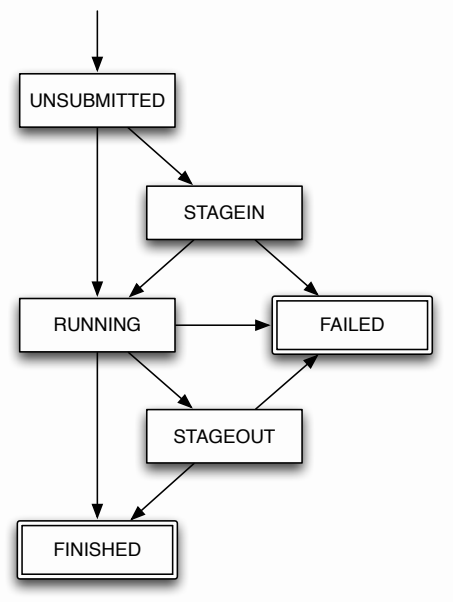

Abbildung 5.4: Jobstati und mögliche Übergänge

void setProps(jobProperties props): Diese Methode dient dem Setzen der Job Properties.

void submit(): Wenn ein Job fertig konfiguriert ist, wird seine Ausführung mit dieser Methode ausgelöst. Der Job wird an die Computeressource submittiert.

Die in Tabelle 5.1 dargestellten Properties dienen der Konfiguration einer Jobsubmission. Diese Properties bilden die minimale Anzahl von Optionen ab, die ein Entwickler benötigt, um erfolgreich einen Computejob im Grid ausführen zu können. Unter Verwendung der Defaultwerte kann ein einfacher Grid-Computejob mit nur einem Parameter definiert werden. Es muss nur das auszuführende Executable angegeben werden. Die anderen Parameter sind entweder optional oder werden durch die Defaultwerte abgedeckt. Bei direkter Verwendung der jeweiligen APIs der Grid Middlewares müsste der Entwickler ein Vielfaches dieser wenigen Optionen einsetzen. Die Properties bedeuten im Einzelnen:

job.targethostname: Dieses Property spezifiziert den Hostnamen des Gridfrontends der Computeressource. Er kann statisch eingesetzt werden oder zur Laufzeit beispielsweise über ein Metascheduling bestimmt werden. Defaultwert: URL des Globus GRAM der Grid Computeressource am Regionalen Rechenzentrum für Niedersachsen.

Name des Properties	Defaultwert
job.targethostname	"gramd1.d-grid.uni-hannover.de"
job.queue	"dgiseq"
job.factory	"PBS"
job.environment	keiner (optional)
job.executable	keiner (muss gesetzt werden)
job.arguments	keiner (optional)
job.stdout	"stderr"
job.stderr	"stdout"
job.stdin	"stdin"
job.filestagein	kein Stage-In
job.filestageout	kein Stage-Out
job.credential	Credential in Dateisystem suchen
job.sandboxdir	wird automatisch generiert
job.deletesandbox	true

Tabelle 5.1: Properties des myGridAPI

job.queue: Computecluster haben meistens getrennte Queues für verschiedene Arten von Jobs, zumindest jeweils eine für parallele und eine für sequenzielle Jobs. Diese muss entsprechend gewählt werden. Defaultwert: „dgiseq“ (Queue für nicht parallele Computejobs der Grid Computeressource am Regionalen Rechenzentrum für Niedersachsen).

job.factory: Die Factory bestimmt, welches Batchsystem verwendet werden soll. Meist gibt es nur eines pro Clusterressource (z. B. PBS). In diesem Fall muss der Defaultwert vom Entwickler nicht geändert werden. Defaultwert: „PBS“ (Name des Batchsystems der Grid Computeressource am Regionalen Rechenzentrum für Niedersachsen)

job.environment: Hier können Umgebungsvariablen hinterlegt werden, die für den Betrieb des Programms auf der Grid-Ressource notwendig sind. Dies können beispielsweise die Pfade für die Ein- und Ausgabedateien sein. Defaultwert: keiner, Parameter ist optional.

job.executable: Das Executable bezeichnet den Pfad und den Namen des Pro-

gramms, das auf dem Cluster ausgeführt werden soll. Es kann entweder im VO-Softwareverzeichnis vorinstalliert sein oder per File Stage In mit dem Job auf die Ressource übertragen werden. Defaultwert: keiner, Parameter muss gesetzt werden.

job.arguments: Diese Argumente werden dem Executable als Kommandozeilenparameter übergeben. Defaultwert: keiner, Parameter ist optional.

job.stdout, job.stderr, job.stdin: Diese Properties geben die Namen der Dateien an, in welche die Standardausgabe, -fehler und -eingabekanäle des Executables des Jobs umgelenkt werden. Defaultwerte: „stdout“, „stderr“ und „stdin“ respektive.

job.filestagein: Dieses Property enthält eine Liste von Dateien mit Pfadangabe, die vor Ausführung des Jobs auf die Computeressource kopiert werden. Die Quelle kann das lokale Dateisystem der Ressource sein, auf der das API eingesetzt wird, oder eine dritte Ressource mit gridFTP-Schnittstelle. Defaultwert: keiner, Parameter ist optional.

job.filestageout: Analog zum File Stage In werden diese Dateien nach Beendigung des Jobs von der Computeressource herunterkopiert. Das Ziel kann entweder das lokale Dateisystem der Ressource sein, auf der das API eingesetzt wird, oder eine dritte Ressource mit gridFTP-Schnittstelle. Defaultwert: keiner, Parameter ist optional.

job.credential: Dieses Property enthält das Grid-Credential als Objekt des Typs `GSSCredential`. Es kann mit dem myCredentialAPI erzeugt bzw. bezogen werden. Die attributbasierte Autorisierung wird hiervon unmittelbar unterstützt, da es an dieser Stelle unerheblich ist, ob ein Credential eingebettete Attribute enthält oder nicht. Defaultwert: wenn der Parameter nicht gesetzt wird, wird das Credential in der Datei `/tmp/x509_up{userid}` gesucht, was dem Standard in Globus Toolkit- und gLite-basierten Gridclients entspricht.

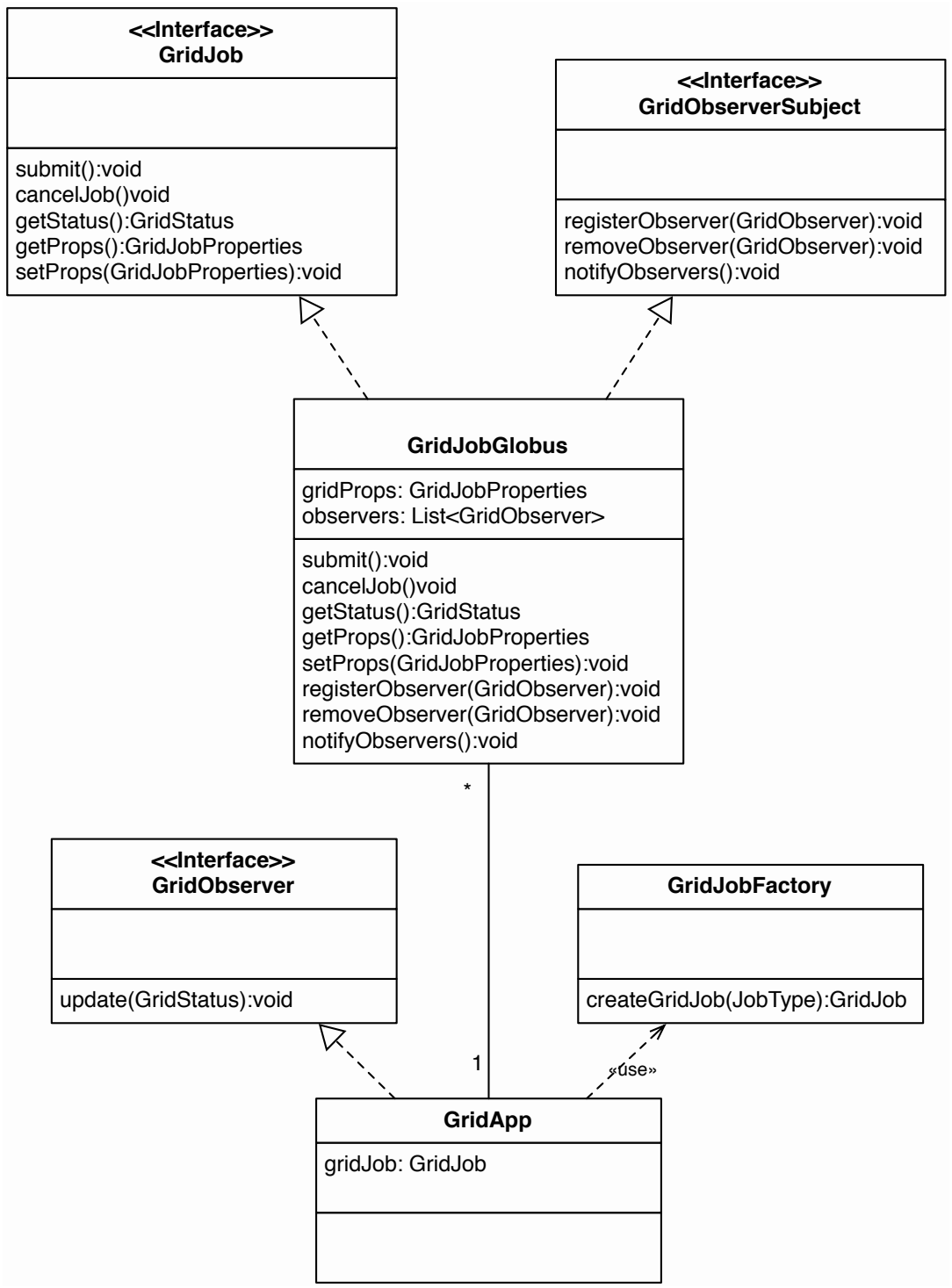

Abbildung 5.5: Klassendiagramm des myJobSubmissionAPI

job.sandboxdir: Um die Ergebnisse verschiedener Jobs, die parallel auf der gleichen Computeressource ausgeführt werden, sauber trennen zu können, werden Jobs in unterschiedlichen temporären Sandboxverzeichnissen ausgeführt. So wird verhindert, dass sie beispielsweise ihre Standardausgabedateien gegenseitig überschreiben. Das API kann automatisch ein Sandboxverzeichnis basierend auf einer Zufallszahl anlegen oder es kann explizit gesetzt werden, beispielsweise falls ein zweiter Job die Ausgaben eines vorherigen Jobs direkt übernehmen soll. Defaultwert: Wenn der Parameter nicht gesetzt wird, wird ein zufälliger Verzeichnisname als Sandbox verwendet und im Arbeitsverzeichnis des Jobs auf der Grid-Ressource angelegt.

job.deletesandbox: Wenn dieses Propery auf `true` gesetzt ist, wird die Sandbox nach Beendigung des Jobs und des File Stage Outs vollständig gelöscht. Defaultwert: „true“.

Unterstützung mehrerer Grid-Middlewares

In Abb. 5.5 ist dargestellt, wie die Unterstützung mehrerer Grid Middlewares mittels einer statischen Methode in der GridJobFactory-Klasse realisiert ist. Die GridApp ist dabei die vom Entwickler mit Hilfe des myJobSubmissionAPI realisierte gridfähige Anwendung. Neben der beispielhaft dargestellten GridJobGlobus-Klasse können durch weitere Implementierungen des GridJob Interfaces und Bereitstellung dieser Implementierungen über die Factoryklasse einfach zusätzliche Middlewares unterstützt werden, ohne dass Änderungen am Code der GridApp notwendig sind. Die oben definierten Properties sind so gewählt, dass sie für die Verwendung der drei berücksichtigten Grid Middlewares hinreichen und keine Änderungen an ihnen notwendig sind, um einen Computejob über eine andere Middleware auszuführen. Weiterhin ist zu erkennen, wie die Überwachung des Status eines Gridjobs durch das Observerpattern realisiert ist.

Kapitel 6

Evaluierung der vertrauenswürdigen attributbasierten Autorisierung im Grid Computing

6.1 Erweiterung der Grid-Security um attributbasierte Autorisierung

6.1.1 Funktionsumfang

Die identitätsbasierte Autorisierung im Grid ist auf lange Sicht nicht ausreichend skalierbar, da jeder Nutzer auf jeder Ressource, die er potentiell nutzen könnte, bekannt sein muss. Dies ist unabhängig davon, ob er diese Ressource auch tatsächlich einmal anfragt. Es ist daher unabdingbar, attributbasierte Autorisierung auf Grid-Ressourcen einzuführen, wenn die Grid-Infrastrukturen weiterhin wachsen. Dies betrifft die Anzahl der Ressourcen, der Virtuellen Organisationen und der Nutzer.

Das zweite Problem der identitätsbasierten Autorisierung im Grid ist, dass ein Nutzer nicht auf derselben Ressource je nach Anwendungsfall mit unterschiedlichen Rechten zugreifen kann. Ein Nutzer kann jedoch mehreren Virtuellen Organisationen angehören oder innerhalb einer Virtuellen Organisation verschiedene Rollen mit unterschiedlichen Rechten besitzen. Es ist ebenfalls nicht möglich für Nutzer, die in mehreren Virtuellen Organisationen Mitglied sind, die Rechte mehrerer VOs in einem Gridjob zu vereinen, beispielsweise wenn Ergebnisdaten aus einer VO für eine andere bereitgestellt werden sollen.

In der Roadmap, die in dieser Arbeit erstellt wurde, wird im ersten Schritt die Autorisierung anhand von VO-Attributen ermöglicht. Dies ermöglicht neue Anwendungsfälle im D-Grid, die bisher nicht umsetzbar waren, wie die Mitgliedschaft in mehreren Virtuellen Organisationen oder die Ausübung mehrerer Rollen in einer Virtuellen Organisation.

Im zweiten Schritt werden zusätzlich Campusattribute und eine Online CA unterstützt, was den Zugang zum Grid deutlich vereinfacht, da das Zertifikatshandling wegfällt und Nutzer ihre Stammdaten für die Nutzung im Grid nicht neu erheben lassen müssen. Dies senkt die Einstiegshürde für die Gridnutzung durch breite Nutzerschaften aus allen Communities erheblich. Weiterhin stehen die Campusattribute auf den Grid-Ressourcen für die Autorisierung zur Verfügung. Dies erlaubt die einfache Zulassung von Nutzern, beispielsweise anhand ihres Status an ihrer Heimateinrichtung. So können zu Lehrzwecken alle Studierenden der Informatik einer Universität auf einer Grid-Ressource mit einer einzigen Regel autorisiert werden, ohne dass ein VO-Management System benötigt wird. Dies steigert die Flexibilität

im Umgang mit weniger stark schützenswerten Ressourcen erheblich und senkt den administrativen Aufwand. Nur so kann eine Grid-Infrastruktur mit steigender Größe skalieren und hunderttausende oder mehr Nutzer und Ressourcen unterstützen. Das ist mit dem aktuellen identitätsbasierten Ansatz nicht möglich.

6.1.2 Sicherheitsaspekte

In Kapitel 3.4 werden Probleme und mögliche Ursachen für Vertrauensverlust bei der Übermittlung von Attributen von verschiedenen Attribute Authorities an zugriffsbeschränkte Ressourcen modelliert. Im Folgenden wird die vorgestellte Lösung für attributbasierte Autorisierung aus dieser Arbeit im Hinblick auf diese Aspekte evaluiert.

Discovery Probleme

Die Discovery Probleme stellen zwar kein Vertrauensproblem dar, müssen aber gelöst werden, um in lose gekoppelten Systemen sicher zu stellen, dass ausreichend Attribute auf Ressourcen vorhanden sind, um eine erfolgreiche Autorisierung zu ermöglichen. Das Attribute Authority Discovery Problem ist durch die Verwendung von Attribute Push gelöst.

Das Service Provider Discovery Problem wird durch die Definition eines einheitlichen Satzes von Attributen gelöst, auf das sich die beteiligten Ressourcenprovider einigen. Hierbei wird sichergestellt, dass die Attribute Authorities diese Attribute auch für alle potentiellen Nutzer pflegen und ausstellen. In der Shibboleth Föderation ist durch die Verwendung des eduPerson Schemas einheitlich geregelt, welche Attribute von den Identity Providern verwaltet werden. Die VO-Management Dienste pflegen innerhalb einer Grid-Infrastruktur wie dem D-Grid ebenso einen einheitlichen Satz von Attributen. Aus diesen Attributen wird von den Ressourcenprovidern ein Subset definiert, welches durch die Online CA bzw. den VO-Management Dienst ausgestellt und von der Clientanwendung in das Gridcredential integriert wird.

Die Discovery Probleme sind somit für den in dieser Arbeit beschriebenen Ansatz für die attributbasierte Autorisierung gelöst.

Binden von X.509 Identitäten an Shibboleth Identitäten

Eine Attribute Assertion stellt für sich gesehen kein kryptographisches Geheimnis dar. Datenschutzgründe sprechen zwar für eine gewisse Vertraulichkeit, aber es kann nicht ausgeschlossen werden, dass eine korrekt signierte Attribute Assertion absichtlich oder unabsichtlich in die Hände eines dritten Nutzers oder Angreifers gerät. Wenn keine Bindung von der X.509 Identität, die für die Authentifizierung im Grid verwendet wird, zur Identität aus der Attribute Assertion hergestellt werden kann, ist es nicht möglich, zu überprüfen, ob ein Missbrauch vorliegt. Daher ist eine überprüfbare Bindung für die hier vorgestellte Lösung notwendig.

Diese Anforderung erfüllt die in dieser Arbeit ausgearbeitete Lösung für Nutzer, die kurzlebige Zertifikate bei der Online CA beziehen. Die Unterstützung von Campusattributen für Nutzer mit langlebigen Zertifikaten ist nicht möglich, da dies voraussetzen würde, dass die X.509 Identität der Nutzer Bestandteil der Attribute Assertion vom Identity Provider seiner Heimateinrichtung wäre. Die VO-Attribute können allerdings auch von Nutzern mit langlebigen Zertifikaten genutzt werden, da diese bei Verwendung von VOMRS/VOMS als VO-Managementsystem automatisch den X.509 Distinguished Name des Nutzers enthalten. Grid-Ressourcen, die also weiterhin identitätsbasiert autorisieren oder die nur VO-Attribute benötigen, werden weiterhin für Nutzer mit langlebigen Zertifikaten unterstützt.

Trust Proxying

Die in dieser Arbeit berücksichtigten Anforderungen für die Erweiterung der attributbasierten Autorisierung im Grid Computing schließen den Einsatz von Trust Proxying aus, da es die Möglichkeit eröffnet, dass aus Versehen oder absichtlich falsche Attribute im Grid für die Autorisierung eingesetzt werden könnten. Mit Trust Proxying kann durch einen kompromittierten Trust Proxy die gesamte Grid-Infrastruktur kompromittiert werden. Die eigentliche Attribute Authority, aus deren Zuständigkeitsbereich ein Attribut gefälscht wurde, hat keine Möglichkeit, dagegen vorzugehen und Grid-Ressourcen haben keine Möglichkeit, einen solchen Missbrauch zu verhindern. Ohne Trust Proxying wird immer nur eine Heimateinrichtung bzw. eine Virtuelle Organisation kompromittiert, deren Identity Provider respektive VO-Managementsystem durch Angreifer kompromittiert sind.

Da die in dieser Arbeit definierte Lösung kein Trust Proxying verwendet und

nach vollständiger Umsetzung trotzdem Support für alle betrachteten Middlewares liefert, ist dieser Angriffsvektor nicht möglich und kann somit in Einklang mit den Policies der Gridcommunities, Ressourcenprovider und Teilnehmern der Shibboleth Föderation umgesetzt werden.

Multiple Attribute Authority Based Access Control

Durch die Bereitstellung von Nutzerattributen aus mehreren Quellen stellen sich neuartige Anforderungen an die eigentlichen Autorisierungsfunktionen auf den Ressourcen. Aktuell eingesetzte Konzepte, wie die Attribute Based Access Control (ABAC) oder die um die Unterstützung mehrerer Policies erweiterte Variante Attribute Based Multipolicy Access Control (ABMAC), sind zwar kompatibel mit Attributen aus mehreren Quellen, können aber nicht alle möglichen Szenarien umsetzen, die durch diese Erweiterung möglich werden. Um die erweiterten Möglichkeiten umzusetzen, wurde in dieser Arbeit die Multiple Attribute Authority Based Access Control (MAABAC) entwickelt, welche ABMAC dahingehend erweitert.

Durch die vertrauenswürdige Bereitstellung von Attributen von mehreren Attribute Authorities und dem MAABAC Konzept wurden in dieser Arbeit die Voraussetzungen geschaffen, die Autorisierung auf Grid-Ressourcen vertrauenswürdig und mit erweiterten Möglichkeiten bei der Formulierung der Policies umzusetzen. Die Umsetzung der Autorisierung auf den Grid-Ressourcen ist im Detail nicht Gegenstand dieser Arbeit und sollte in Zukunft weiter verfolgt werden (siehe Kapitel 7.2). Da die Softwarekomponenten, die in der hier erarbeiteten Infrastruktur eingesetzt werden, sämtlich als Open Source Software zur Verfügung stehen, können sie bei Bedarf entsprechend erweitert werden. Bis dahin muss das implizite Vertrauen zwischen PDP und PIP ausreichen, indem ausschließlich Attribute mit Scope verwendet werden. Dieser kann von PIP geprüft werden und ein Attribut im Konfliktfall verworfen werden. Die MAABAC-Lösung macht diesen Ansatz flexibler, indem auch Attribute ohne Scope unterstützt werden und in besonderen Anwendungsfällen explizit erlaubt werden kann, dass eine Attribute Authority Attribute ausstellt, für die sie eigentlich nicht zuständig ist. Jede Ressource kann in MAABAC so konfiguriert werden, dass sie dies für ein ganz spezielles Attribut zulässt, während dies mit dem herkömmlichen Ansatz über Scopes nur für alle Attribute mit dem entsprechenden Scope vorgesehen ist.

6.1.3 Leistungsfähigkeit

Die beschriebenen Vorgänge auf den Clientsystemen (Beziehen eines kurzlebigen Zertifikates mit Campus-Attributen sowie Abrufen der VO-Attribute vom VOMS) lassen sich in einer Clientapplikation zusammenfassen. Aus Nutzersicht muss nur die Legitimation (bspw. Benutzername/Passwort) für den Identity Provider seiner Heimateinrichtung eingegeben werden. Die restlichen Schritte können automatisch ablaufen.

Dieser Vorgang dauert in der Regel weniger als eine Minute und muss auch nicht für jeden Gridjob durchgeführt werden sondern, wenn die maximale Laufzeit von kurzlebigen Zertifikaten ($\approx 11\frac{1}{2}$) genutzt wird. Es muss allerdings darauf geachtet werden, dass alle Jobs beendet sind bevor das Credential ausläuft. Es ist somit kein signifikanter Einfluss auf die Laufzeiten typischer Grid Jobs vorhanden und auch die Nutzerakzeptanz ist sehr hoch durch die vergleichsweise seltene Ausführung des Vorgangs und der geringen Nutzerinteraktion.

Der zusätzliche Overhead durch die attributbasierte Autorisierung auf den Grid-Ressourcen ist im Vergleich zu den erwarteten Joblaufzeiten von üblicherweise mehreren Stunden auf Hochleistungsrechenressourcen vernachlässigbar. Auch bei High Throughput-Anwendungen mit vielen kurz laufenden Grid Jobs ist der zusätzliche Overhead nicht signifikant. Die Zeit, die für die erweiterte Autorisierung notwendig ist, ist um mehrere Größenordnungen kleiner als die Laufzeit von Computejobs bzw. Datenübertragungen im Grid und somit nicht signifikant.

6.2 Evaluierung des myGridAPI am Beispiel der GDI-Grid Middleware

Das myGridAPI zur vereinfachten Erstellung von Gridanwendungen durch domänenspezifische Entwickler wurde erfolgreich im Projekt GDI-Grid (Geodateninfrastruktur-Grid) eingesetzt. Im Folgenden wird die Nützlichkeit des API durch die Erfolge, die in diesem Projekt erzielt wurden, evaluiert.

Im Bereich der Geodateninfrastrukturen gibt es bereits umfangreiche Web Service-basierte Infrastrukturen zur Verwendung in Geoinformationssystemen. Diese sind vom Open Geospatial Consortium (OGC) in Form von OpenGIS Web Services (OWS) [49] spezifiziert. Viele dieser Dienste bieten Funktionen an, die große Da-

tenbestände verarbeiten. Hierzu sind skalierbare Compute- und Datenmanagementdienste notwendig, die einzelne Server nicht anbieten können. Da dies durch Gridtechnologien ermöglicht wird, ist das Forschungsziel die Anbindung von OWS-Diensten an das Grid, so dass die Clients der OWS-Infrastruktur weiter verwendet werden und gleichzeitig die mächtigen Funktionen des Grids genutzt werden können. Um dieses Ziel zu erreichen, wird das myJobSubmissionAPI in der im GDI-Grid Projekt erstellten Software eingesetzt.

6.2.1 Die existierende OGC-WS-basierte GDI-Infrastruktur

Eine typische nicht gridifizierte Geodateninfrastruktur basiert auf den vom Open Geospatial Consortium spezifizierten Komponenten:

- Ein *Client* für den Zugriff auf OGC Web Services. Dieser unterstützt den Zugriff auf gridifizierte Services nicht.
- *Computeressourcen*, z. B. ein Web Processing Service (WPS), werden als OWS-konforme Web Services bereitgestellt. Sie unterstützen gemäß den OGC Standards keinerlei Zugriffskontrolle aus dem Service heraus. Existierende Lösungen verwenden hier meist eine vorgelagerte Autorisierung z. B. basierend auf der IP-Adresse des Clients.
- *Speicherressourcen* werden auch als OWS-konforme Dienste angeboten. Dies kann beispielsweise ein Web Map Service (WMS) sein. Aus Sicht der Sicherheit gelten dieselben Einschränkungen wie bei den Computeressourcen.

Eine Trennung von Schnittstelle (d. h. dem OGC Web Service) und der speicher- oder rechenintensiven Logik ist innerhalb der OGC Standards zwar möglich, wird aber selten umgesetzt. Hier kann das Grid helfen, Speicher- und Computeressourcen bereitzustellen, die von den OGC Web Services verwendet werden.

Eine weitere wichtige Anforderung, die bisher nicht durch OGC Web Services umgesetzt werden kann, ist die Orchestrierung der Geodienste mittels einer Workflow Engine.

6.2.2 Anforderungen an die GDI-Grid Infrastruktur

Die drei Middlewares der Computeressourcen der D-Grid Infrastruktur sollen flexibel unterstützt werden. Da die Entwickler der Geodienste hierbei nicht die APIs

sämtlicher Middlewarekomponenten lernen sollen, wird das myGridAPI eingesetzt. Dieses ermöglicht das middlewareunabhängige Programmieren von fachspezifischen Diensten für eine Grid-Infrastruktur.

Die spezifischen Anforderungen für diese Abstraktionsschicht sind:

Ausführung von Geoprozessen auf den D-Grid Compute Ressourcen.

File Stage In und File Stage Out: Vor Ausführung müssen Eingabedaten und eventuell das ausführbare Executable auf die Ressource übertragen werden. Nach der Ausführung müssen die Ergebnisse der Berechnung ausgelesen werden können.

Zugriff auf die Standardeingabe-, Standardausgabe und Standardfehlerkanäle des Executables.

Bereitstellung und Verwaltung eines Grid-Credentials für die Jobabgabe. Das Credential kann entweder von einem MyProxy Service bezogen werden oder von der anfragenden Ressource per Delegation bereitgestellt werden, falls diese die Grid Security Infrastructure unterstützt.

Unterstützung des Web Services Resource Framework: Die gridifizierten Geoservices basieren auf dem Framework des Globus Toolkit 4 und unterstützen damit das Web Service Resource Framework (WSRF) um Gridjobs als zustandsbehaftete Web Services zugreifbar zu machen. Das WSRF wird im Folgenden erläutert.

Das Web Services Resource Framework

Im GDI-Grid Projekt wird die Grid Middleware Globus Toolkit 4 eingesetzt, um gridifizierte Geodienste zu erstellen. Da im D-Grid bisher die Grid-Ressourcen der Kerninfrastruktur über alle drei eingesetzten Middlewares erreichbar sind, wird auch hierfür auf das Globus Toolkit zurückgegriffen. Dies wird allerdings durch das myGridAPI verborgen und kann somit ohne weitere Änderungen an den Grid-Geodiensten an andere Grid Middlewares angepasst werden.

Im Globus Toolkit 4 wird für Web Service-basierte Komponenten das *Web Services Resource Framework* (WSRF) [30] eingesetzt. Dieses erweitert die bekannten SOAP-basierten statuslosen Web Services um Funktionen, die den expliziten Zugriff

auf statusbehaftete Ressourcen über Web Service Technologien ermöglichen. Statusbehaftete Web Services sind im Grid Computing besonders sinnvoll, da häufig lang existierende Sessions benötigt werden, die asynchron von den jeweiligen Clients kontaktiert werden. Ein Beispiel hierfür ist die Ausführung eines Computejobs auf einem Globus Toolkit GRAM, dem Frontend für Rechenressourcen. Hier wird ein potentiell Stunden bis Tage laufender Job initiiert und sein Status regelmäßig abgerufen. Durch den Einsatz von WSRF ist es möglich, den Job als Ressource direkt zu adressieren anstatt einen statuslosen Web Service aufzurufen und über eine Job ID oder ähnliche Mechanismen das gewünschte Ergebnis zu erhalten.

Dieser Ansatz erleichtert auch die Autorisierung von Zugriffen. So kann beispielsweise das Anlegen von neuen Ressourcen (Compute Jobs) durch eine Policy gesteuert werden, die anhand von Nutzeridentitäten oder Nutzerattributen autorisiert, während auf die Ressource selber nur von ihrem Ersteller zugegriffen werden darf. Somit ist sichergestellt, dass niemand die Jobs eines anderen beeinflussen kann.

Ein weiterer Vorteil ist, dass auf dem Globus Toolkit 4 basierende Web Services einfach in die Grid Security Infrastructure eingebunden werden können. In jedem Globus Web Service Container ist ein Delegation Service vorhanden, der Delegationen von Proxycredentials entgegen nimmt. Es wird eine Ressource erstellt, die direkt adressiert werden kann. Wenn anschließend ein Geodienst im selben Globus Container kontaktiert wird, kann ihm eine Referenz auf die Ressource mit dem Proxycredential übergeben und somit ein Grid-Securitykontext erzeugt werden. Der Entwickler des Geodienstes muss sich somit nur minimal mit der Delegation von Grid-Credentials befassen. Das Grid-Credential aus der Ressource des Delegation Service kann dann mittels des myGridAPI verwendet werden, um aus dem Geodienst heraus weitere Grid-Dienste aufzurufen. Des Weiteren kann in Zukunft die in dieser Arbeit konzipierte attributbasierte Autorisierung direkt übernommen werden, da sie nur einmal für den gesamten Globus Container installiert werden muss und dann von allen Diensten über Konfigurationsparameter ohne Codeänderungen genutzt werden kann.

6.2.3 Architektur

Komponenten der GDI D-Grid Infrastruktur

Im GDI-Grid Projekt werden Grid-Infrastrukturen für drei Beispielszenarios implementiert. Diese sind Hochwassersimulation, Routing im Katastrophenfall sowie Lärmimmissionssimulation in urbanen Umgebungen. Diese Szenarios haben eine wichtige Gemeinsamkeit: Sie haben immer wieder auftretende Workflows die spezifische Datenressourcen abfragen und mittels dieser Daten einen oder mehrere Geoprozesse ausführen. Basierend auf diesen Gemeinsamkeiten ist es das Ziel, eine Grid-Infrastruktur zu entwerfen und aufzubauen, die diese Workflows effizient, skalierend und sicher ausführt. Dazu werden Komponenten für die Orchestrierung, die Sicherheit, den Datenzugriff und den Zugriff auf Rechenressourcen benötigt.

Die Komponenten der gridifizierten OGC-Infrastruktur sind:

- *Gridclients* sind Desktopapplikationen, welche den Zugriff auf gridbasierte Geodienste erlauben. Der Client ermöglicht den Zugriff auf WSRF-basierte Web Services unter Verwendung der Grid Security Infrastructure für die Authentifizierung und Autorisierung.

- *Non-Grid Clients* sind Clients für Geoinformationssysteme, die auf OGC-WS konforme Dienste zugreifen können. Für den Zugriff auf gridbasierte Geodienste sind sie nicht geeignet.

- *WSRF-basierte Geodienste* bieten die Verarbeitung von Daten an. Auf sie kann entweder direkt von einem Gridclient zugegriffen werden oder über die Workflow-Engine. Die eigentlichen Executables, welche die Algorithmen für das Geoprocessing enthalten, sind entweder im Geodienst gespeichert und werden gemeinsam mit den Daten auf eine Grid Rechenressource übertragen oder liegen bereits vorinstalliert auf den Grid Rechenressourcen vor. Zusätzlich ist es auch möglich, die Berechnungen direkt in dem WSRF Geodienst auszuführen. Dies ist aber nur sinnvoll, wenn die Berechnung selten aufgerufen wird und sehr wenig Rechenzeit beansprucht. In diesem Fall kann der Overhead durch die Submittierung des Jobs an eine Grid-Ressource vermieden werden, falls dies zu einer schnelleren Ausführung führt.

- *OGC-WS basierte Fassaden* für die WSRF-basierten Geodienste dienen dazu, den Zugriff auf WSRF-basierte Geodienste auch für OWS Clients zu ermög-

lichen. Sie bieten dem Client eine herkömmliche OGC-konforme Schnittstelle an und wandeln die Anfrage in eine WSRF-basierte Anfrage um, die an einen WSRF-basierten Geodienst weitergeleitet wird. Hierfür muss ein Grid-Credential bezogen werden, das nicht vom Client delegiert wird.

- Grid Computeressourcen werden über einen *Globus Toolkit GRAM* bereitgestellt. Diese Dienste sind nicht Bestandteil der GDI-Grid Infrastruktur sondern werden von den D-Grid Ressourcenprovidern bereitgestellt. Somit müssen lediglich die Executables mit den Geoprocessingalgorithmen auf diesen Ressourcen vorinstalliert werden oder mit dem Job an diese übertragen werden. Die Schnittstelle, d. h. der WSRF Geodienst, ist von der Ausführung der eigentlichen Berechnung getrennt.

- Eine gridfähige *Workflowengine* wird in der GDI-Grid Infrastruktur eingesetzt. Diese basiert auf BPEL [16]. Um nicht gridifizierte OWS Clients zu unterstützen, kann eine OWS-konforme Fassade verwendet werden, die den Workflow als OWS-Dienst bereitstellt. Um den Sicherheitsanforderungen des Grid gerecht zu werden, muss hierfür ein Grid-Credential bereitgestellt werden, da die OWS Clients dies nicht unterstützen. Dies geschieht durch einen MyProxy-Dienst, der ebenfalls Bestandteil der D-Grid Infrastruktur ist und somit nicht innerhalb der GDI-Grid Infrastruktur realisiert werden muss. Die Einbindung der BPEL Workflow Engine in die GDI-Grid Infrastruktur ist in [21] im Detail beschrieben.

- *Gridbasierte Storageressourcen* stellen über die Grid Security Infrastructure den gesicherten Zugriff auf verschiedene Geodaten bereit. Dies können Relationale Datenbanken, XML-Strukturen oder dateibasierte Ressourcen sein. Ein gemeinsames Frontend für diese Datentypen bietet OGSA-DAI [3] [38] und wird somit in der GDI-Grid Infrastruktur eingesetzt. Details sind in [34] beschrieben.

Das myJobSubmissionAPI ermöglicht die Abgabe von generischen Berechnungsjobs an einen Globus Toolkit GRAM. Es unterstützt den Domänenentwickler allerdings nicht dabei, die Logik eines Geoprozesses zu extrahieren und als Executable in das Grid zu schicken. Es bleibt Aufgabe des Entwicklers, aus der möglicherweise monolithisch in den OGC Web Service integrierten Logik ein auf den Grid-Ressourcen

ausführbares Executable zu erzeugen. Dies ist in aller Regel ein Linux Kommandozeilenprogramm oder interpretierbarer Bytecode, z. B. für die Java VM. Für den produktiven Betrieb sollte ein derartiges Executable vom VO-Software Administrator auf den Grid-Ressourcen vorinstalliert und getestet werden. In diesem Fall ist sichergestellt, dass das Executable mit den jeweiligen Grid-Ressourcen kompatibel ist.

Bei dynamischer Nutzung nicht vorkonfigurierter Grid-Ressourcen ist es möglich, dass das mit dem Job gemeinsam submittierte Executable nicht ausführbar ist. In diesem Fall bietet es sich an, über Pilotjobs die Eignung einer Grid-Ressource zu testen und in einer Datenbank das Ergebnis zu speichern. In diesem Fall wird nach und nach eine Positiv- bzw. Negativliste von Grid-Ressourcen aufgebaut. Wenn die Grid-Infrastruktur einen Metascheduler nutzt, d. h. der submittierende Dienst nicht weiß, auf welcher Grid-Ressource der Job schließlich ausgeführt wird, ist dies unter Umständen nicht möglich. Eine solche Infrastruktur wird daher nur sinnvoll einsetzbar sein, wenn die Grid-Ressourcen hinreichend genaue Selbstbeschreibungen anbieten, anhand derer automatisch geprüft werden kann, ob die genauen Anforderungen des Jobs erfüllt werden. Bis derartige Funktionen verfügbar sind, werden von Hand eingerichtete und getestete Grid-Ressourcen vorkonfiguriert und beispielsweise im Round Robin Verfahren für die Ausführung des Jobs kontaktiert.

Zusammenspiel der Komponenten

Die WSRF-basierten Geodienste verarbeiten alle Anfragen unabhängig davon, ob sie von einem Grid Client wie der Workflow Engine, einer Endbenutzerapplikation oder einer OWS Fassade aufgerufen werden. Der WSRF Geoservice entscheidet, ob eine Anfrage lokal berechnet werden kann oder ob Rechen- oder Speicherkapazität aus dem Grid notwendig ist. Wenn eine Grid-Ressource mit der eigentlichen Bearbeitung des Jobs beauftragt wird, geschieht dies über das myJobSubmissionAPI, die Bestandteil des myGridAPI ist.

Die Komponenten und deren Interaktionen sind in Abb. 6.1 dargestellt, die zeitliche Abfolge von Aufrufen eines Grid Clients in Abb. 6.2. Eine Anfrage von einem GSI-fähigen Client ermöglicht das direkte Delegieren des Proxycredentials und die Verwendung von MyProxy ist somit nicht notwendig. Das myCredentialAPI wird verwendet, um aus einem lokalen Zertifikat ein Proxycredential zu erzeugen oder

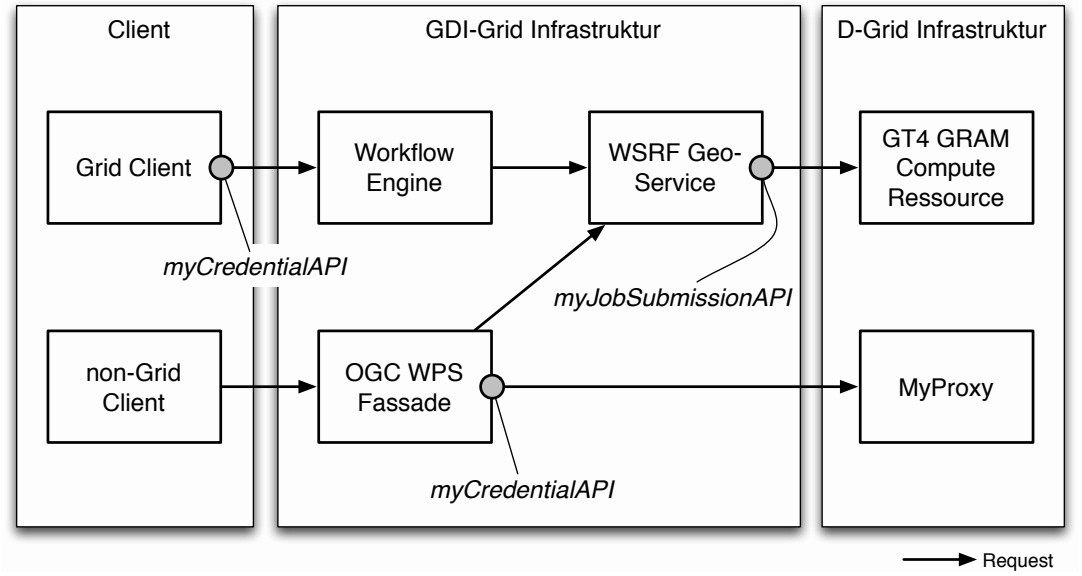

Abbildung 6.1: Architektur der GDI Computedienste

um ein Credential von einer Online CA zu beziehen. Wenn ein Zugriff von einer OWS Fassade die Ausführung auf einer Grid-Ressource erfordert, muss ein GSI Proxycredential verfügbar gemacht werden, da der OWS Dienst und der non-Grid Client keine Delegation von Proxycredentials unterstützen. Dieses Credential muss von einem MyProxy Dienst unter Verwendung des myCredentialAPI bezogen werden. Details hierzu werden in Kapitel 4.4.1 diskutiert.

6.2.4 Leistungsfähigkeit der GDI-Grid Infrastruktur

Im GDI-Grid Projekt wurde die hier vorgestellte Infrastruktur erfolgreich für das Überflutungsszenario implementiert. Grundlage ist ein digitales Geländemodell der Elbmündung über 218 km^2 mit 5m Auflösung (8,7 Millionen Punkte). Über die Workflowengine wird das Terrain in 67 überlappende Kacheln unterteilt. Die im Projekt gridifizierten, domänenspezifischen Dienste für Bruchkantenerkennung verwenden Computeressourcen des D-Grid über die Globus Toolkit Middleware, um die rechenaufwändigen Operationen auszuführen. Jede Kachel wird von einem 2,66 GHz Intel Xeon Core mit zwei Gigabyte Arbeitsspeicher berechnet. Da so alle Kacheln parallel auf Grid-Ressourcen berechnet werden können, benötigt diese Operation weniger als 30 Minuten. Die lokale Berechnung auf einem Desktopcomputer mit einem 2 GHz Intel Core 2 Prozessor und zwei Gigabyte Arbeitsspeicher benötigt dagegen

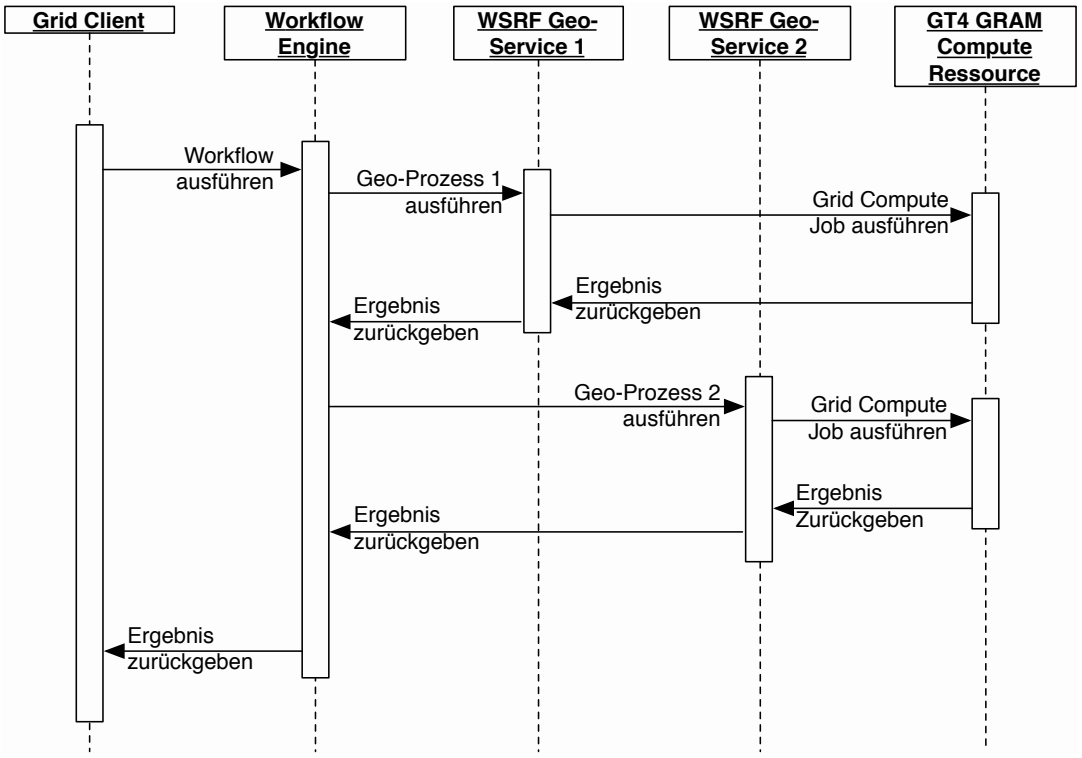

Abbildung 6.2: Sequenzdiagramm eines Geodaten-Workflows

mehr als vier Stunden. Der zusätzliche Overhead durch die Gridkomponenten, die Übertragung der Daten ins Grid und das wieder zusammensetzen der Kacheln ist also deutlich niedriger als der durch die Verwendung des Grids erzielte Speedup durch Parallelisierung der Ausführung. Für größere Gebiete wird der Speedup noch größer ausfallen, solange ausreichend viele freie Computeressourcen im Grid zur Verfügung stehen.

Diese GDI-Grid Infrastruktur wäre ohne ein solches angepasstes Grid-API nicht möglich gewesen, da einige Entwickler aus der Geodateninfrastruktur-Domäne auch nach Schulung durch die Grid Security Experten Schwierigkeiten hatten, ihre Software erfolgreich in die Grid-Infrastruktur einzubinden.

Kapitel 7

Zusammenfassung und Ausblick

7.1 Zusammenfassung

Ausgangspunkt dieser Arbeit waren die neuartigen Anforderungen, die sich für Autorisierungsinfrastrukturen in föderierten Infrastrukturen wie dem Grid Computing ergeben. Aktuelle Grid-Infrastrukturen verwenden vorwiegend die wenig flexible und für große Nutzerzahlen nicht skalierende identitätsbasierte Autorisierung, da die Voraussetzungen für die vertrauenswürdige attributbasierte Autorisierung noch nicht gegeben sind. Die besondere Herausforderung liegt darin begründet, dass die Attribute aus mehreren Quellen stammen und diese Quellen keine direkte Beziehung zu den Gridressourcen haben.

In dieser Arbeit wurden Probleme identifiziert, die in einer solchen Infrastruktur zu Vertrauensverlust in die Korrektheit von auf Grid-Ressourcen zur Verfügung stehenden Attribute führen. Dies sind Discovery Probleme, Bindungsprobleme der Attribute an Entitäten, Trust Proxying und die Behandlung von Attributen aus mehreren Quellen durch die Autorisierungsfunktionen in den Grid-Komponenten.

Die Discovery Probleme werden durch den Einsatz von Attribute Push und die Forderung nach einem Standardsatz von Attributen gelöst. Bindungsprobleme werden für VO-Attribute durch den Einsatz von X.509-basiertem VO-Management gelöst. Hierbei wird die Bindung der Identität des Nutzers an die übermittelten Attribute sichergestellt, indem die Attribute vom VO-Managementsystem den X.509 Distinguished Name des Nutzers enthalten, der auch durch die Authentifizierungsfunktionen des Grid Security Infrastructure aus dem Proxycredential gewonnen wird. Campusattribute werden durch die Verwendung einer Online CA an das ausgestellte Zertifikat gebunden, da die Ausstellung des Zertifikats auf den gleichen Daten beruht wie die durch den Identity Provider der Heimatorganisation des Nutzers ausgestellte Attribute Assertion. Um die Attribute aus unterschiedlichen Quellen auf den Grid-Ressourcen auch für erweiterte Autorisierungsfunktionen verwenden zu können, wird die Multiple Attribute Authority Based Access Control (MAABAC) als Erweiterung der Attribute Based Multipolicy Authorization Access Control (ABMAC) entwickelt.

Das vorgestellte Design und die Implementierung für eine attributbasierte Autorisierungsinfrastruktur für das Grid zeigt Technologien auf, welche die Anforderungen an die attributbasierte Autorisierung in Grid-Infrastrukturen erfüllen. Herbei wurde besonderes Augenmerk auf die Vertrauenswürdigkeit der vorgestellten Lösung

gelegt. Weiterhin wurde sichergestellt, dass bereits erfolgreich eingesetzte Lösungen weiterverwendet werden können und somit bereits erfolgreich eingeführte und betriebene Komponenten weiter genutzt werden.

Durch die Ergebnisse dieser Arbeit werden die Voraussetzungen für die attributbasierte Autorisierung auf Grid-Ressourcen geschaffen. Hierfür werden Campusattribute aus einer Shibboleth Föderation durch eine Online CA in kurzlebige Nutzerzertifikate integriert, VO-Attribute von einem VO Management Service (VOMS) ausgestellt und vom Nutzer in ein Proxycredential integriert. Dies führt dazu, dass diese Attribute auf den Grid-Ressourcen zur Verfügung stehen und für Autorisierungsentscheidungen genutzt werden können. Durch die in dieser Arbeit vorgestellten Forschungsergebnisse wird sichergestellt, dass dieses Vorgehen für alle beteiligten Stakeholder vertrauenswürdig ist und somit im Einklang mit den jeweiligen Policies umgesetzt werden kann.

Durch das in dieser Arbeit vorgestellte Verfahren können Attribute aus mehreren Quellen vertrauenswürdig zu den Grid-Ressourcen übertragen werden. Da noch nicht alle Komponenten der Grid Middlewares derart erweitert wurden, dass sie die gewünschten Funktionen für die attributbasierten Autorisierung bereitstellen können, wurde eine zweistufige Roadmap definiert, um die neuen Funktionen der attributbasierten Autorisierung schrittweise einzuführen. Im ersten Schritt wird, basierend auf verfügbarer Software, die Nutzung von Attribute Certificates bzw. SAML Assertions mit VO-Attributen für die Autorisierung im Grid ermöglicht. Bei Verfügbarkeit der entsprechenden Komponenten wird im zweiten Schritt die Unterstützung für SAML Assertions mit Campusattributen hinzufügt und die Übermittlung der VO-Attribute für alle Middlewares auf SAML Assertions umgestellt. Dadurch, dass im zweiten Schritt ausschließlich auf den auch außerhalb der Grid-Welt akzeptierten Standard SAML gesetzt wird, ist die Voraussetzung für künftige Interoperabilität mit anderen Grid-Infrastrukturen oder beliebigen SAML-basierten Authentifizierungs- und Autorisierungsinfrastrukturen geschaffen.

Da die Anpassung existierender Software an die Schnittstellen des Grids und insbesondere an die Grid Security Infrastructure hohe Anforderungen an die Entwickler stellt, ist eine Abstraktion dieser Schnittstellen notwendig. Eine solche Abstraktion wird in dieser Arbeit definiert und eine Java-basierte Implementierung vorgestellt. Hierbei steht die einfache Anwendung und niedrige Lernkurve für den Einsatz der Abstraktion im Vordergrund. Dieses Ziel wurde erreicht und die Implementierung

erfolgreich von mehreren Entwicklern aus der Geodateninfrastrukturdomäne eingesetzt. Hierdurch wurde eine Infrastruktur geschaffen, die gleichzeitig die Weiterverwendung existierender Clients ermöglicht und die massive Rechenleistung, die durch das Grid bereitgestellt wird, nutzt.

7.2 Ausblick

Die in dieser Arbeit vorgestellten Ergebnisse liefern die notwendige Grundlage für die Einführung einer attributbasierten Autorisierungsinfrastruktur für Grids, indem Attribute aus mehreren Quellen vertrauenswürdig an die Ressourcen übermittelt werden können. Direkt hierauf aufbauende Arbeiten sind notwendig, die diese Attribute auf den Ressourcen für Autorisierungsentscheidungen verwenden. Hierfür ist ein middlewareübergreifendes Autorisierungsframework sinnvoll, das auf den Schnittstellen der Grid Middlewares aufsetzt, aber so viel Funktionalität wie möglich derart bereitstellt, dass Mehrfachimplementierungen und -konfiguration vermieden werden.

Das erwartbare zukünftige Wachstum der Grid-Infrastrukturen und das internationale Zusammenwachsen der nationalen Grid-Inseln zu einem globalen Grid erfordern es, noch mehr als bisher gemeinsame standardisierte Schnittstellen einzusetzen. Diese Arbeit basiert bereits auf entsprechenden Standards wie der Security Assertion Markup Language und kann somit einen Beitrag leisten, das Ziel eines globalen Grids zu erreichen. Die zukünftige Herausforderung wird sein, die gewachsenen, auf unterschiedlichen Middlewares basierenden Grid-Infrastrukturen derart mit allgemein akzeptierten Schnittstellen zu erweitern, dass die eingesetzte Middleware keine Rolle mehr spielt.

Durch das Paradigma des Cloud Computings, das viele Eigenschaften des Grid Computings teilt, werden derzeit neue Herangehensweisen entwickelt, um die Notwendigkeit lokaler Ressourcen weiter zu minimieren. Hierbei werden viele weiterführende Konzepte aus der Sicherheit im Grid Computing wie die kryptographisch abgesicherte Delegation von Nutzerrechten bisher nicht eingesetzt. Wenn beide Ansätze in der Zukunft verschmelzen, muss daran geforscht werden, wie man Errungenschaften beider Ansätze so kombiniert, dass das Ergebnis eine mächtigere Technologie darstellt als die beiden Teile getrennt betrachtet. Da die Ergebnisse dieser Arbeit zwar am Beispiel des Grid Computings erforscht wurden, aber generell auf föderierte organisationsübergreifende IT-Infrastrukturen übertragen werden können, ist die

Anwendung der Ergebnisse auch im Cloud Computing oder in einer gemeinsamen Cloud/Grid-Infrastruktur möglich.

Anhang A

Lastenheft des myGridAPI

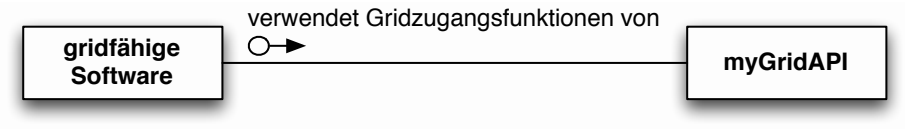

Abbildung A.1: Umweltdiagramm des myGridAPI

A.1 Zielbestimmung

Mit dem Produkt myGridAPI soll es Softwareentwicklern ohne erweiterte Kenntnisse in der Programmierung von Gridsoftware erleichtert werden, sichere und funktionstüchtige Software für den Einsatz in Grid-Infrastrukturen zu entwickeln.

Das Unterziel der Erstellung funktionstüchtiger Software soll die Entwickler dahingehend entlasten, dass sie zum einen ein API verwenden können, welches eine möglichst niedrige Einstiegshürde in der Anwendung besitzt. Zum anderen sollen sie sich nicht mit dem internen Aufbau der Grid-Infrastruktur auskennen müssen, für die sie Software entwickeln.

Das Unterziel der Erstellung sicherer Software soll es den Entwicklern ermöglichen, ohne genaue Kenntnisse über die Funktionsweise der Sicherheitsfunktionen in Grid-Infrastrukturen zu haben, Software so zu implementieren, dass sie in der Zielinfrastruktur sicher betrieben werden kann.

A.2 Produkteinsatz

Das myGridAPI soll von allen Entwicklern domänenspezifischer Software für eine Grid-Infrastruktur genutzt werden können.

Die Anwendungsbereiche sind vielfältig und durch die Communities bestimmt, die Interesse am Einsatz ihrer Software in einer Grid-Infrastruktur haben. Dies betrifft in erster Linie wissenschaftliche Communities.

A.3 Produktübersicht

Die Produktumgebung ist als Umweltdiagramm in Abb. A.1 dargestellt.

A.4 Produktfunktionen

Typische Geschäftsprozesse, die mit Hilfe des myGridAPI implementiert werden sollen, sind folgende:

/**LF10**/ Erstellen eines Credentials

Akteur: Gridfähiges Programm

Beschreibung: Ein Grid-Credential wird aus den lokal gespeicherten Nutzercredentials erstellt.

/**LF20**/ Abrufen eines Credentials

Akteur: Gridfähiges Programm

Beschreibung: Ein Grid-Credential wird von einem entfernten Credential Repository abgerufen.

/**LF30**/ Gridjob vorbereiten

Akteur: Gridfähiges Programm

Beschreibung: Eingabedaten und Software werden im Grid verfügbar gemacht.

/**LF40**/ Initiierung eines Gridjobs

Akteur: Gridfähiges Programm

Beschreibung: Ein Gridjob wird zur Ausführung submittiert.

/**LF50**/ Überwachung des Status eines Gridjobs.

Akteur: Gridfähiges Programm

Beschreibung: Der aktuelle Status eines Gridjobs wird abgefragt.

/**LF60**/ Abrufen der Ergebnisse eines Gridjobs

Akteur: Gridfähiges Programm

Beschreibung: Nach Beendigung werden die Ausgabedaten eines Gridjobs abgerufen.

Alle diese Arbeitsabläufe sollen implementiert werden können, ohne Rücksicht auf die verwendete Grid-Middleware nehmen zu müssen.

A.5 Produktdaten

Langfristig zu speichernde Hauptdaten fallen für das myGridAPI nicht an.

A.6 Produktleistungen

An das Produkt werden keine besonderen Leistungsanforderungen gestellt. Die meisten Funktionen erfordern eine Interaktion mit Grid-Diensten. Die Antwortzeiten der Funktionen des APIs werden durch deren Antwortzeiten dominiert.

A.7 Qualitätsanforderungen

Die Qualitätsanforderungen sind in Tabelle A.1 dargestellt und im Folgenden erläutert.

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität	x			
Zuverlässigkeit			x	
Benutzbarkeit	x		x	
Effizienz			x	
Änderbarkeit	x			
Sicherheit	x			

Tabelle A.1: Qualitätsanfoderungen an das myGridAPI

Funktionalität: Das API soll den Zweck erfüllen, die Funktionalität zuverlässig zu gewährleisten, sodass die Entwickler sich nicht in die Bedienung von Grid APIs einarbeiten müssen. Daher ist diese Anforderung sehr hoch.

Zuverlässigkeit: Das API soll eine normale Zuverlässigkeit gewährleisten. Besondere Anforderungen sind nicht vorhanden.

Benutzbarkeit: An die Benutzbarkeit werden hohe Anforderungen gestellt. Das API soll minimale Anforderungen an den benutzenden Entwickler stellen.

Effizienz: Das API soll keinen signifikanten zeitlichen Overhead in der auf ihm aufgebauten Software erzeugen.

Änderbarkeit: Das API soll ohne Änderungen am Code auf andere Gridinfrastrkturen übertragen werden können, indem erfahrene Gridentwickler neue Adaptoren hinzufügen können.

Sicherheit: Das API muss in den Sicherheitsfunktionen hohe Anforderungen erfüllen, da ansonsten ein Zugriff Dritter auf die Grid-Credentials der Nutzer der Software nicht vermieden werden kann.

Anhang B

Pflichtenheft des myGridAPI

B.1 Zielbestimmung

Musskriterien

Funktionen, die vom myGridAPI bereitgestellt werden müssen sind:

- Definieren und Submittieren von Grid Compute Jobs
- Verwaltung und Bereitstellung von Grid Credentials

Wunschkriterien

Keine.

Abgrenzungskriterien

Das API soll keine weiterführenden Funktionen bieten, die über das Minimum dessen, was für einen erfolgreich ausgeführten Gridjob notwendig ist, hinausgehen. Falls weitergehende Funktionen benötigt werden, muss ein vollwertiges Grid-API eingesetzt werden und es muss sichergestellt sein, dass der Entwickler über die notwendigen Kenntnisse verfügt, diese korrekt anzuwenden.

B.2 Produkteinsatz

Das myGridAPI dient der Entwicklung sicherer domänenspezifischer Gridsoftware.

Anwendungsbereiche

Der Anwendungsbereich ist die Softwareentwicklung in Communities, die Gridinfrastrukturen nutzen möchten.

Zielgruppen

Die Zielgruppe sind Entwickler von domänenspezifischer Gridsoftware, die keine tiefergehenden Kenntnisse über den inneren Aufbau der Grid-Infrastruktur und der Programmierung von Grid-Diensten haben.

Betriebsbedingungen

Es gelten die üblichen Betriebsbedingungen für den Softwareentwicklungsbereich.

B.3 Produktübersicht

Siehe Übersichtsdiagramm über die Geschäftsprozesse des myGridAPI (Abb. B.1).

B.4 Produktfunktionen

Geschäftsprozesse

/**F10**/ (/**LF10**/) Erstellen eines Credentials ohne Attribute aus lokalem Zertifikat

Ziel: Aus dem Nutzerzertifikat und dem privaten Schlüssel ein Grid-Credential erstellen

Kategorie: primär

Vorbedingung: Nutzerzertifikat und privater Schlüssel müssen im lokalen Dateisystem vorliegen

Nachbedingung Erfolg: Ein Proxycredential für die Delegation im Grid liegt vor

Nachbedingung Fehlschlag: Ein Gridjob kann nicht initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- Zertifikat und privaten Schlüssel aus lokalem Dateisystem laden
- Passphrase vom Nutzer abfragen
- Proxycredential generieren

/**F11**/ Erstellen eines Credentials von Online CA

Ziel: Kurzlebiges Credential von einer Online CA beziehen

Kategorie: primär

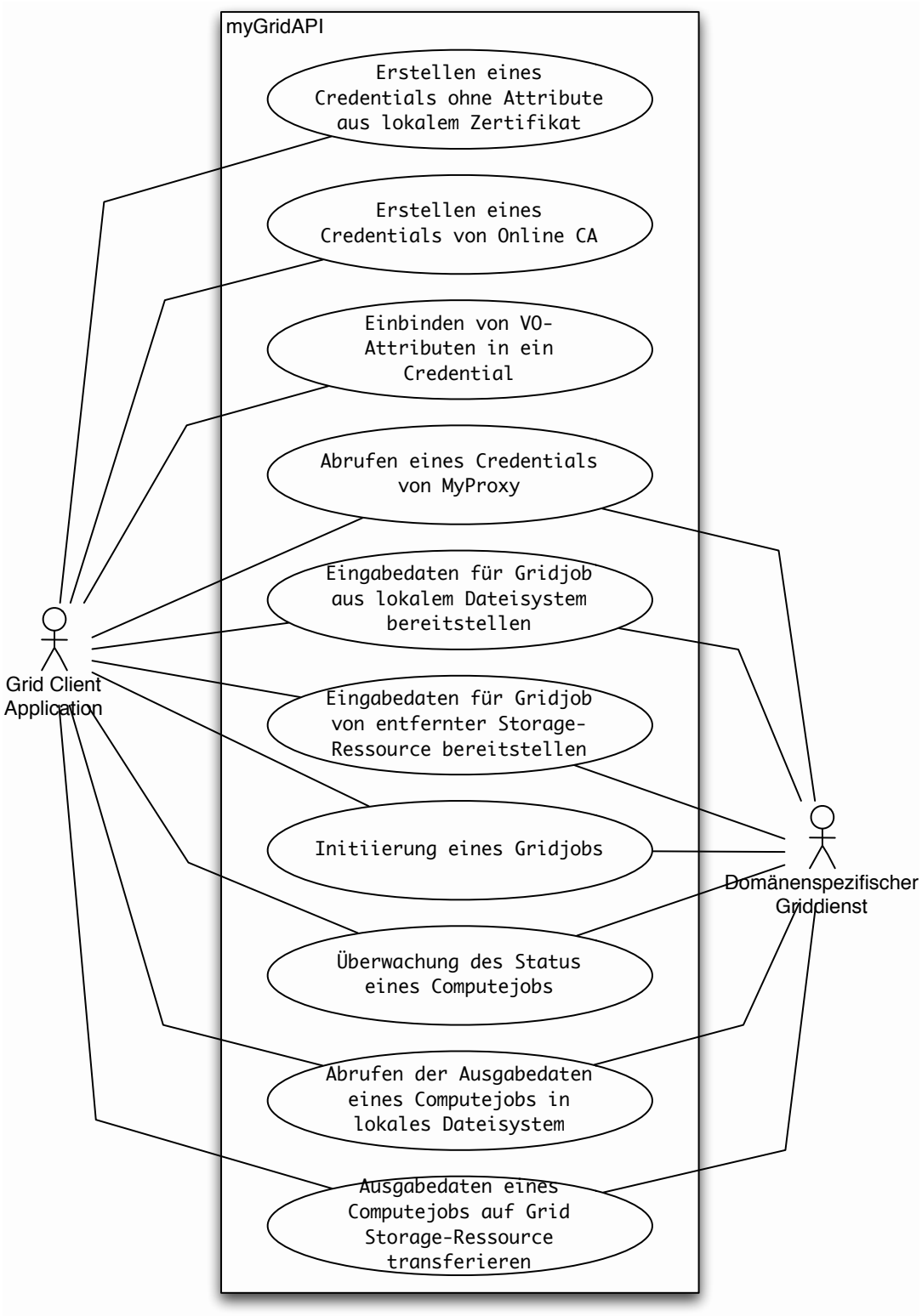

Abbildung B.1: Geschäftsprozesse des myGridAPI

Vorbedingung: Nutzer muss über Login auf Online CA über den Identity Provider seiner Heimateinrichtung verfügen

Nachbedingung Erfolg: Ein Proxycredential für die Delegation im Grid liegt vor

Nachbedingung Fehlschlag: Ein Gridjob kann nicht initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- Online CA kontaktieren und Zertifikat abfragen (dieses kann optional Campusattribute enthalten)
- Benutzername und Passwort vom Nutzer abfragen
- Aus kurzlebigem Zertifikat und privaten Schlüssel ein Proxycredential generieren

/**F12**/ Einbinden von VO-Attributen in ein Credential

Ziel: Aus dem Nutzerzertifikat und dem privaten Schlüssel ein Grid-Credential erstellen

Kategorie: primär

Vorbedingung: Nutzerzertifikat und privater Schlüssel müssen im lokalen Dateisystem vorliegen

Nachbedingung Erfolg: Ein Proxycredential für die Delegation im Grid liegt vor

Nachbedingung Fehlschlag: Ein Gridjob kann nicht initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- Zertifikat und privaten Schlüssel aus lokalem Dateisystem laden
- Passphrase vom Nutzer abfragen
- Proxycredential generieren

/**F20**/ /**LF20**/ Abrufen eines Credentials von MyProxy

Ziel: Ein Grid-Credential wird von einem MyProxy Credential Repository abgerufen

Kategorie: primär

Vorbedingung: Der Nutzer muss über ein MyProxy Upload Tool ein Credential im MyProxy hinterlegt haben

Nachbedingung Erfolg: Ein Proxycredential für die Delegation im Grid liegt vor

Nachbedingung Fehlschlag: Ein Gridjob kann nicht initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- MyProxy Credential Repository kontaktieren
- Benutzername und Passwort vom Nutzer abfragen
- Proxycredential von MyProxy empfangen

/**F30**/ /**LF30**/ Eingabedaten für Gridjob aus lokalem Dateisystem bereitstellen

Ziel: Die Eingabedaten und ggf. das Executable müssen auf eine Grid Computeressource übertragen werden

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und die Datenquelle muss existieren

Nachbedingung Erfolg: Die angegebenen Daten befinden sich auf der Computeressource

Nachbedingung Fehlschlag: Der Gridjob kann nicht erfolgreich initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- Daten werden über geeignetes Protokoll übertragen
- Endstatus der Übertragung wird zurückgemeldet

/**F31**/ Eingabedaten für Gridjob von entfernter Storage-Ressource bereitstellen (Third-Party Transfer)

Ziel: Die Eingabedaten und ggf. das Executable müssen auf eine Grid Computeressource übertragen werden

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und die Datenquelle muss existieren

Nachbedingung Erfolg: Die angegebenen Daten befinden sich auf der Computeressource

Nachbedingung Fehlschlag: Der Gridjob kann nicht erfolgreich initiiert werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm bereitet Gridjob vor

Beschreibung: Folgende Schritte sind notwendig:

- Third-Party File Transfer-Dienst wird beauftragt, Daten zu übertragen
- Endstatus der Übertragung wird zurückgegeben

/**F40**/ /**LF40**/ Initiierung eines Gridjobs

Ziel: Der Computejob wird auf einer Grid-Ressource ausgeführt

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und die Eingabedaten müssen übertragen sein

Nachbedingung Erfolg: Die Berechnung wurde erfolgreich durchgeführt

Nachbedingung Fehlschlag: Der Computejob muss noch einmal ausgeführt werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm initiiert Berechnung des Gridjobs

Beschreibung: Folgende Schritte sind notwendig:

- Grid Computeressource wird kontaktiert und Jobparameter werden übertragen
- Endstatus der Berechnung wird zurückgegeben

/F50/ /LF50/ Überwachung des Status eines Computejobs

Ziel: Der Status eines laufenden Computejobs ist bekannt

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und ein Computejob muss gerade ausgeführt werden

Nachbedingung Erfolg: Der Status des Computejobs ist bekannt

Nachbedingung Fehlschlag: Der Status des Computejobs ist nicht bekannt, ein Fehlschlag muss angenommen werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm benötigt Status eines Computejobs

Beschreibung: Folgende Schritte sind notwendig:

- Grid Computeressource wird kontaktiert und Jobstatus wird abgefragt
- Aktueller Status des Computejobs wird zurückgegeben

/F60/ /LF60/ Abrufen der Ausgabedaten eines Computejobs in lokales Dateisystem

Ziel: Die Ausgabedaten werden auf den lokalen Rechner übertragen

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und ein Computejob muss erfolgreich ausgeführt worden sein

Nachbedingung Erfolg: Die Ausgabedaten liegen lokal vor

Nachbedingung Fehlschlag: Die Ausgabedaten liegen nicht lokal vor, der Job muss noch einmal ausgeführt werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm benötigt Ausgabedaten eines Computejobs

Beschreibung: Folgende Schritte sind notwendig:

- Daten werden über geeignetes Protokoll übertragen
- Endstatus der Übertragung wird zurückgemeldet

/**F61**/ Ausgabedaten eines Computejobs auf Grid Storage-Ressource transferieren (Third-Party Transfer)

Ziel: Die Ausgabedaten werden auf die entfernte Storage-Ressource übertragen

Kategorie: primär

Vorbedingung: Ein gültiges Grid-Credential muss vorliegen und ein Computejob muss erfolgreich ausgeführt worden sein

Nachbedingung Erfolg: Die Ausgabedaten liegen auf entfernter Storage Ressource vor

Nachbedingung Fehlschlag: Die Ausgabedaten liegen nicht auf entfernter Storage Ressource vor, der Job muss noch einmal ausgeführt werden

Akteur: Gridfähiges Programm

Auslösendes Ereignis: Gridfähiges Programm benötigt Ausgabedaten eines Computejobs für weitere Verarbeitung

Beschreibung: Folgende Schritte sind notwendig:

- Third-Party File Transfer-Dienst wird beauftragt, Daten zu übertragen
- Endstatus der Übertragung wird zurückgegeben

B.5 Produktdaten

Es fallen keine myGridAPI-spezifischen Produktdaten an.

B.6 Produktleistungen

L10 Alle Aufrufe dürfen maximal 2 Sekunden länger benötigen als die Zeit, die gebraucht wird, um die Antwort von den entfernten Diensten zu erhalten

B.7 Qualitätsanforderungen

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität				
Angemessenheit			x	
Richtigkeit	x		x	
Interoperabilität	x			
Ordnungsmäßigkeit	x			
Sicherheit	x			
Zuverlässigkeit				
Reife			x	
Fehlertoleranz			x	
Wiederherstelltbarkeit			x	x
Benutzbarkeit				
Verständlichkeit	x			
Erlernbarkeit	x			
Bedienbarkeit	x			
Effizienz				
Zeitverhalten		x		
Verbrauchsverhalten				x
Änderbarkeit				
Analysierbarkeit			x	
Modifizierbarkeit	x			
Stabilität			x	
Prüfbarkeit			x	
Übertragbarkeit				
Anpassbarkeit		x		
Installierbarkeit				x
Konformität		x		
Austauschbarkeit		x		

Tabelle B.1: Qualitätsanfoderungen an das myGridAPI

B.8 Benutzungsoberfläche

Das myGridAPI benötigt keine Benutzungsoberfläche

B.9 Nichtfunktionale Anforderungen

Die wichtigste nichtfunktionale Anforderung ist die Minimierung von Sicherheitslücken durch Fehlbedienung, sowohl durch den domänenspezifischen Entwickler bei der Erstellung von gridfähiger Software als auch durch den Nutzer bei der Benutzung des Grids.

B.10 Technische Produktumgebung

Das myGridAPI muss in javabasierten Software-Entwicklungsumgebungen eingesetzt werden. Die fertige Software wird auf Clientsystemen und Grid-Diensten eingesetzt.

Software

Für die Entwicklung von Grid-Diensten: Jedes Betriebssystem mit lauffähigem Java Development Kit. Für die Ausführung der fertigen Software: Jedes Betriebssystem mit lauffähigem Java Runtime Environment

Hardware

Alle Komponenten laufen problemlos auf aktueller Desktop- und Serverhardware.

B.11 Spezielle Anforderungen an die Entwicklungsumgebung

Es werden keine abweichenden Anforderungen an die Entwicklungsumgebung gegenüber der Produktumgebung gestellt.

B.12 Gliederung in Teilprodukte

Das myGridAPI wird in zwei Teilprodukte geteilt. Das eine Teilprodukt (myJobSubmissionAPI) dient der Abgabe und Überwachung eines Gridjobs an ein Computeelement der Grid-Infrastruktur. Das andere Teilprodukt (myCredentialAPI) dient der Verwaltung von Grid-Credentials und der Sammlung und Einbettung von Attributen in das Grid-Credential.

Funktion	myJobSubmissionAPI	myCredentialAPI
F10		x
F11		x
F12		x
F20		x
F30	x	
F31	x	
F40	x	
F50	x	
F60	x	
F61	x	

B.13 Ergänzungen

Keine.

Literaturverzeichnis

[1] Allen, G. ; Davis, K. ; Godale, T. ; Hutanu, A. ; Kaiser, H. ; Kielmann, T. ; Merzky, A. ; Nieuwpoort, R. van ; Reinefeld, A. ; Schintke, F. ; Scott, T. ; Seidel, E. ; Ullmer, B.: The Grid Application Toolkit: Toward Generic and Easy Application Programming Interfaces for the Grid. In: *Proceedings of the IEEE* Bd. 93, Issue 3, 2005, S. 534–550

[2] Ananthakrishnan, R. ; Freeman, T. ; Ceccanti, A. ; Ciaschini, V. ; Hesselroth, T.: *Incubator/VOMS*. Online: `http://dev.globus.org/wiki/Incubator/VOMS`, Zuletzt zugegrifen am 26.03.2011

[3] Antonioletti, M. ; Atkinson, M.P. ; Baxter, R. ; Borley, A. ; Hong, N.P. C. ; Collins, B. ; Hardman, N. ; Hume, A. ; Knox, A. ; Jackson, M. ; Krause, A. ; Laws, S. ; Magowan, J. ; Paton, N.W. ; Pearson, D. ; Sugden, T. ; Watson, P. ; Westhead, M.: The Design and Implementation of Grid Database Services in OGSA-DAI. In: *Concurrency and Computation: Practice and Experience* Bd. 17, Issue 2-4, 2005, S. 357–376

[4] Balzert, H.: *Lehrbuch der Softwaretechnik: Software-Entwicklung*. 2. Auflage. Heidelberg : Spektrum, Akad. Verl., 2008. – ISBN 3827404800

[5] Barton, T. ; Basney, J. ; Freeman, T. ; Scavo, T. ; Siebenlist, F. ; Welch, V. ; Ananthakrishnan, R. ; Baker, B. ; Goode, M. ; Keahey, K.: Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, GridShib, and MyProxy. In: *Proceedings (preliminary) of the 5th Annual PKI R&D Workshop* (2006)

[6] Barton, T. ; Basney, J. ; Freeman, T. ; Scavo, T. ; Siebenlist, F. ; Welch, V. ; R.Ananthakrishnan ; Baker, B. ; Goode, M. ; Keahey,

K.: Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, Gridshib, and MyProxy. In: *5th Annual PKI R&D Workshop* (2006), April. http://www.globus.org/toolkit/presentations/gridshib-pki06-final.pdf

[7] Benedyczak, K. ; Lewandowski, M. ; Nowinski, A. ; Bala, P.: UNICORE Virtual Organizations System. In: Wyrzykowski, Roman (Hrsg.) ; Dongarra, Jack (Hrsg.) ; Karczewski, Konrad (Hrsg.) ; Wasniewski, Jerzy (Hrsg.): *Parallel Processing and Applied Mathematics* Bd. 6068, Springer Berlin / Heidelberg, 2010 (Lecture Notes in Computer Science), 155-164

[8] Cantor, S. ; Scavo, T.: *myVocs*. Online: https://spaces.internet2.edu/display/GS/MyVocs, Zuletzt zugegriffen am 26.03.2011

[9] Chadwick, D.: Authorisation in Grid computing. In: *Information Security Technical Report* 10 (2005), Nr. 1, 33 - 40. http://www.sciencedirect.com/science/article/B6VJC-4FY3DNK-5/2/352fd6ccff17a59ceeb27fb40eb62e59

[10] Chadwick, D. ; Otenko, A.: The PERMIS X.509 Role Based Privilege Management Infrastructure. In: *Future Generation Computer Systems* 19 (2003), February, Nr. 2, 277-289. http://www.cs.kent.ac.uk/pubs/2003/2109

[11] CILogon: *GridShib - CILogon*. Online: http://www.cilogon.org/gridshib, Zuletzt zugegriffen am 26.03.2011

[12] Demchenko, Y.: *gLite Java Authorisation Framework (gJAF) and Authorisation Policy coordination*. Online: http://staff.science.uva.nl/~demch/presentations/EGEE06-mwsg-gjaf-yd.pdf, Zuletzt zugegriffen am 26.03.2011

[13] DFN-Verein: *Certificate Policy and Certification Practice Statement of the Public Key Infrastructure in the Deutsche Forschungsnetz - SLCS - DFN*. Online: https://www.pki.dfn.de/fileadmin/PKI/DFN-PKI_SLCS-CPCPS_v11.pdf, Mai 2009

[14] DFN-Verein: *DFN-AAI: DFN-AAI*. Online: https://www.aai.dfn.de/, Februar Zuletzt zugegriffen am 25.03.2011

[15] Dierks, T. ; Allen, C.: *RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2*. IETF Network Working Group, August 2008

[16] Dörnemann, T. ; Friese, T. ; Herdt, S. ; Juhnke, E. ; Freisleben, B.F.: Grid Workflow Modelling Using Grid-Specific BPEL Extensions. In: *Proceedings of German e-Science Conference*, 2007, S. 1–9

[17] Education, Directory Working G. f.: *Certificate Policy and Certification Practice Statement of the Public Key Infrastructure in the Deutsche Forschungsnetz - SLCS - DFN*. Online: `http://middleware.internet2.edu/eduperson/docs/internet2-mace-dir-eduperson-200806.html`, June 2008

[18] EUGridPMA: *European Policy Management Authority for Grid Authentication*. Online: `http://www.eugridpma.org/`, Zuletzt zugegriffen am 26.03.2011

[19] Farrell, S. ; Housley, R.: *RFC 3281: An Internet Attribute Certificate Profile for Authorization*. IETF Network Working Group, April 2002

[20] Fermilab: *The Virtual Organization Management Registration Service*. Online: `http://computing.fnal.gov/docs/products/vomrs/`, zuletzt zugegriffen am 20.03.2011

[21] Fleuren, T. ; Muller, P.: BPEL Workflows Combining Standard OGC Web Services and Grid-enabled OGC Web Services. In: *Software Engineering and Advanced Applications, 2008. SEAA '08. 34th Euromicro Conference*, 2008. – ISSN 1089–6503, S. 337–344

[22] Flury, P. ; Tschopp, V. ; Lenggenhager, T. ; Witzig, C.: *Shibboleth Interoperability with Attribute Retrieval through VOMS*. Online: `https://edms.cern.ch/cedar/plsql/doc.info?document_id=807849`, Januar 2007

[23] Foster, I.: What is the Grid? - a three point checklist. In: *GRIDtoday* 1 (2002), July, Nr. 6. `http://www-fp.mcs.anl.gov/~{}foster/Articles/WhatIsTheGrid.pdf`

[24] Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: *Journal of Computer Science and Technology* 21 (2006), 513-520. `http://dx.doi.org/10.1007/s11390-006-0513-y`. – ISSN 1000–9000

[25] Foster, I. ; Kesselman, C.: *The Grid 2: Blueprint for a New Computing Infrastructure*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2003. – ISBN 1558609334

[26] Foster, I. ; Kesselman, C. ; Tsudik, G. ; Tuecke, S.: A security architecture for computational grids. In: *CCS '98: Proceedings of the 5th ACM conference on Computer and communications security*. New York, NY, USA : ACM Press, 1998. – ISBN 1–58113–007–4, S. 83–92

[27] Foster, I. ; Kesselman, C. ; Tsudik, G. ; Tuecke, S.: A Security Architecture for Computational Grids. In: *Proceedings of the 5th ACM Conference on Computer and Communications Security*. New York, NY : ACM Press, 1998, S. 83–91

[28] Garzoglio, G. ; Alderman, I. ; Altunay, M. ; Ananthakrishnan, R. ; Bester, J. ; Chadwick, K. ; Ciaschini, V. ; Demchenko, Y. ; Ferraro, A. ; Forti, A. ; Groep, D. ; Hesselroth, T. ; Hover, J. ; Koeroo, O. ; La Joie, C. ; Levshina, T. ; Miller, Z. ; Packard, J. ; Sagehaug, H. ; Sergeev, V. ; Sfiligoi, I. ; Sharma, N. ; Siebenlist, F. ; Venturi, V. ; Weigand, J.: Definition and Implementation of a SAML-XACML Profile for Authorization Interoperability Across Grid Middleware in OSG and EGEE. In: *Journal of Grid Computing* 7 (2009), 297-307. `http://dx.doi.org/10.1007/s10723-009-9117-4`. – ISSN 1570–7873. – 10.1007/s10723-009-9117-4

[29] Gietz, P. ; Haase, M. ; Pfeiffenberger, H. ; Schiffers, M.: *IVOM Work Package 2 Report: VO-Management Requirements from A Community Perspective*. Online: `http://www.d-grid.de/fileadmin/user_upload/documents/DGI-FG1-IVOM/requirements-v1.0.pdf`, 2007

[30] Graham, S. ; Karmarkar, A. ; Mischkinsky, J. ; Robinson, I. ; Sedukhin, I.: *Web Services Resource 1.2 (WS-Resource*. Online: `http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf`, April 2006

[31] Grimm, C. ; Groeper, R. ; Makedanz, S. ; Pfeiffenberger, H. ; Gietz, P. ; Haase, M. ; Schiffers, M. ; Ziegler, D.W.: Trust Issues in Shibboleth-Enabled Federated Grid Authentication and Authorization Infrastructures Supporting Multiple Grid Middleware. In: *Proceedings of the IEEE International Conference on e-Science and Grid Computing*, 2007, S. 569 –576

[32] Groeper, R. ; Grimm, C. ; Makedanz, S. ; P., Hans ; Z., Wolfgang ; Gietz, P. ; Schiffers, M.: A concept for attribute-based authorization on D-Grid resources. In: *Future Generation Computer Systems* 25 (2009), Nr. 3, 275 - 280. http://dx.doi.org/10.1016/j.future.2008.05.008. – DOI 10.1016/j.future.2008.05.008. – ISSN 0167–739X

[33] Groeper, R. ; Grimm, C. ; Piger, S. ; Wiebelitz, J.: An Architecture for Authorization in Grids using Shibboleth and VOMS. In: *Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on*, 2007. – ISSN 1089–6503, S. 367 –374

[34] Groeper, R. ; Kunz, C. ; Grimm, C.: Connecting OGC web services and the Grid using Globus Toolkit 4 and OGSA-DAI. In: *Grid Computing, 2009 10th IEEE/ACM International Conference on*, 2009, S. 66 –73

[35] Internet2: *Shibboleth Project - Internet 2 Middleware*. Online: http://shibboleth.internet2.edu/, Zuletzt zugegriffen am 26.03.2011

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

[37] Jha, S. ; Kaiser, H. ; Merzky, A. ; Weidner, O.: Grid Interoperability at the Application Level Using SAGA. In: *e-Science and Grid Computing, IEEE International Conference on*, 2007, S. 584 –591

[38] Karasavvas, K. ; Antonioletti, M. ; Atkinson, M.P. ; Hong, N.P. C. ; Sugden, T. ; Hume, A.C. ; Jackson, M. ; Krause, A. ; Palansuriya, C.: Introduction to OGSA-DAI Services. In: *Lecture Notes in Computer Science* Bd. 3458, 2005, S. 1–12

[39] Kesselman, C. ; Foster, I.: *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1998 http://www.amazon.co.uk/exec/obidos/ASIN/1558604758/citeulike-21. – ISBN 1558604758

[40] Kunz, C. ; Wiebelitz, J. ; Smith, M.: An attack-resilient Grid auditing infrastructure. In: *Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on*, 2010, S. 635 –639

[41] La Joie, C.: *Argus: Introduction.* Online: `https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZIntro`, Dezember Zuletzt zugegriffen am 26.03.2011

[42] Lang, B. ; Foster, I. ; Siebenlist, F. ; Ananthakrishnan, R. ; Freeman, T.: A Multipolicy Authorization Framework for Grid Security. In: *Network Computing and Applications, 2006. NCA 2006. Fifth IEEE International Symposium on*, 2006, S. 269 –272

[43] Laszewski, G. von ; Foster, I. ; Gawor, J. ; Lane, P.: A Java Commodity Grid Toolkit. In: *Concurrency: Practice and Experience* Bd. 13, 2001, S. 645–662

[44] Lorch, M. ; Proctor, S. ; Lepro, R. ; Kafura, D. ; Shah, S.: First experiences using XACML for access control in distributed systems. In: *Proceedings of the 2003 ACM workshop on XML security.* New York, NY, USA : ACM, 2003 (XMLSEC '03). – ISBN 1–58113–777–X, 25–37

[45] Meisner, C.: *Erstellung eines grafischen MyProxy-Clients.* Bachelorarbeit, Leibniz Universität Hannover, März 2009

[46] Novotny, J. ; Tuecke, S. ; Welch, V.: An online credential repository for the Grid: MyProxy. In: *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, 2001, S. 104 –111

[47] OASIS Open: *OASIS.* Online: `http://www.oasis-open.org/home/index.php`, Zuletzt zugegriffen am 26.03.2011

[48] OASIS Open: *OASIS Security Services (SAML) TC.* Online: `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security`, Zuletzt zugegriffen am 26.03.2011

[49] Open Geospatial Consortium, Inc.: *OGC(R) Standards and Specifications | OGC(R).* Online: `http://www.opengeospatial.org/standards/`, Zuletzt zugegriffen am 26.03.2011

[50] Pordes, R. ; Petravick, D. ; Kramer, B. ; Olson, D. ; Livny, M. ; Roy, A. ; Avery, P. ; Blackburn, K. ; Wenaus, T. ; Würthwein, F. ; Foster, I. ; Gardner, R. ; Wilde, M. ; Blatecky, A. ; McGee, J. ; Quick, R.: The

open science grid. In: *Journal of Physics: Conference Series* 78 (2007), Nr. 1, 012057. http://stacks.iop.org/1742-6596/78/i=1/a=012057

[51] Sandhu, R. S. ; Coyne, E. J. ; Feinstein, H. L. ; Youman, C. E.: Role-Based Access Control Models. In: *IEEE Computer* 29 (1996), Nr. 2, 38-47. citeseer.ist.psu.edu/sandhu96rolebased.html

[52] Schiffers, M.: *Management dynamischer Virtueller Organisationen in Grids*, LMU München, Diss., Juli 2007

[53] Schneider, K.: *Abenteuer Softwarequalität. Grundlagen und Verfahren für Qualitätssicherung und Qualitätsmanagement.* Dpunkt Verlag, 2007

[54] The EGEE Project: *gLite - Ligthweight Middleware for Grid Computing.* Online: http://glite.web.cern.ch/glite/, Zuletzt zugegriffen am 26.03.2011

[55] The European Organization for Nuclear Research: *Virtual Organization Membership Service.* Online: http://edg-wp2.web.cern.ch/edg-wp2/security/voms/voms.html, Zuletzt zugegrffen: 26.03.2011

[56] Tuecke, S. ; Welch, V. ; Engert, D. ; Pearlman, L. ; Thompson, M.: *RFC 3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile.* IETF Network Working Group, June 2004

[57] Venturi, V. ; Riedel, M. ; Memon, S. ; Memon, S. ; Stagni, F. ; Schuller, B. ; Mallmann, D. ; Tweddell, B. ; Gianoli, A. ; Berghe, S. van d. ; Snelling, D. ; Streit, A.: Using SAML-Based VOMS for Authorization within Web Services-Based UNICORE Grids. In: *Euro-Par 2007 Workshops: Parallel Processing* Bd. 4854, Springer Berlin / Heidelberg, 2008 (Lecture Notes in Computer Science), 112-120

[58] Vollbrecht, J. ; Calhoun, P. ; Farrell, S. ; Gommans, L. ; Gross, G. ; Bruijn, B. de ; Laat, C. de ; Holdrege, M. ; Spence, D.: *RFC 2904: AAA Authorization Framework.* IETF Network Working Group, August 2000

[59] Vyssotsky, V. A. ; Corbató, F. J. ; Graham, R. M.: Structure of the multics supervisor. In: *Proceedings of the November 30–December 1, 1965, fall*

joint computer conference, part I. New York, NY, USA : ACM, 1965 (AFIPS '65 (Fall, part I)), 203–212

[60] Yuan, E. ; Tong, J.: Attributed based access control (ABAC) for Web services. In: *Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on*, 2005, S. 569

Akademischer Lebenslauf

Persönliche Daten	
Name	Ralf Gröper
Geburtsdatum	30.06.1978
Geburtsort	Hamburg
Ausbildung	
1984-1988	Grund- und Hauptschule Oststeinbek
1988-1997	Gymnasium im Schulzentrum Glinde
1998-2000	Technische Universität Hamburg-Harburg; Studium der Allgemeinen Ingenieurwissenschaften
2000-2004	Leibniz Universität Hannover, Bachelorstudium der Angewandten Informatik, Abschluss mit Bachelor of Science in Angewandter Informatik
2004-2006	Leibniz Universität Hannover, Masterstudium der Informatik, Abschluss mit Master of Science in Informatik
Wissenschaftliche Berufstätigkeit	
2006-2011	Wissenschaftlicher Mitarbeiter an der Leibniz Universität Hannover, Fakultät für Elektrotechnik und Informatik, Institut für verteilte Systeme. Mitarbeit in den Forschungsprojekten „IVOM – Interoperabilität und Integration von VO-Management Technologien in D-Grid“ und „GDI-Grid – Geodateninfrastruktur-Grid“

Wissenschaftliche Veröffentlichungen

R. Groeper, C. Grimm, S. Makedanz, H. Pfeiffenberger, W. Ziegler, P. Gietz, M. Schiffers: A Concept for Attribute-Based Authorization on D-Grid Resources, The International Journal of Grid Computing: Theory, Methods and Applications, Vol. 25, Nr. 3, S. 275-280, März 2009, Elsevier

R. Groeper, C. Kunz, C. Grimm: Connecting OGC Web Services and the Grid using Globus Toolkit 4 and OGSA-DAI, Proc. of the 10th IEEE / ACM International Conference on Grid Computing (Grid2009) 13. - 15. August 2009

S. Piger, C.Kunz, C. Grimm, R. Groeper: Enhancing Security in Grids through Self-Restricted Delegation of Rights with User-based Policies, Proc. of the Parallel and Distributed Computing and Networks (PDCN 2008), Innsbruck Februar 2008

S. Piger, C. Grimm, R. Groeper, C. Kunz: A Comprehensive Approach to Self-Restricted Delegation of Rights in Grids, Proc. Eigth IEEE International Symposium on Cluster Computing and the Grid 2008 (CCGrid2008), Lyon Mai 2008

C. Grimm, R. Groeper, S. Makedanz, H. Pfeiffenberger, P. Gietz, M. Haase, M. Schiffers, W. Ziegler: Trust Issues in Shibboleth-Enabled Federated Grid Authentication and Authorization Infrastructures Supporting Multiple Grid Middleware, Proceedings of the Third IEEE International Conference on e-Science and Grid Computing, Bangalore, India 10-13 December 2007

R. Groeper, C. Grimm, S. Piger, J. Wiebelitz: An Architecture for Authorization in Grids using Shibboleth and VOMS, Proc. of the 33rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Lübeck, Germany, August 2007.

S. Piger, C. Grimm, J. Wiebelitz, R. Groeper: An Approach to Restricted Delegation of User Rights based on the gLite Middleware, Proc. Cracow Grid Workshop 06, Krakau, Polen, Oktober 2006